axioms

MDPI

Article

Explainable Fuzzy AI Challenge 2022: Winner's Approach to a Computationally Efficient and Explainable Solution

Sunny Mishra [1], Amit K. Shukla [2,*] and Pranab K. Muhuri [1]

[1] Department of Computer Science, South Asian University, New Delhi 110021, India
[2] Faculty of Information Technology, University of Jyvaskyla, P.O. Box 35 (Agora), 40014 Jyvaskyla, Finland
* Correspondence: amit.k.shukla@jyu.fi or amitkshukla@live.com

Abstract: An explainable artificial intelligence (XAI) agent is an autonomous agent that uses a fundamental XAI model at its core to perceive its environment and suggests actions to be performed. One of the significant challenges for these XAI agents is performing their operation efficiently, which is governed by the underlying inference and optimization system. Along similar lines, an Explainable Fuzzy AI Challenge (XFC 2022) competition was launched, whose principal objective was to develop a fully autonomous and optimized XAI algorithm that could play the Python arcade game "Asteroid Smasher". This research first investigates inference models to implement an efficient (XAI) agent using rule-based fuzzy systems. We also discuss the proposed approach (which won the competition) to attain efficiency in the XAI algorithm. We have explored the potential of the widely used Mamdani- and TSK-based fuzzy inference systems and investigated which model might have a more optimized implementation. Even though the TSK-based model outperforms Mamdani in several applications, no empirical evidence suggests this will also be applicable in implementing an XAI agent. The experimentations are then performed to find a better-performing inference system in a fast-paced environment. The thorough analysis recommends more robust and efficient TSK-based XAI agents than Mamdani-based fuzzy inference systems.

Keywords: explainable AI; fuzzy systems; AI agents; Mamdani inference system; TSK; 03B52

MSC: 03B52

Citation: Mishra, S.; Shukla, A.K.; Muhuri, P.K. Explainable Fuzzy AI Challenge 2022: Winner's Approach to a Computationally Efficient and Explainable Solution. *Axioms* **2022**, *11*, 489. https://doi.org/10.3390/axioms11100489

Academic Editor: Humberto Bustince

Received: 8 August 2022
Accepted: 14 September 2022
Published: 20 September 2022

1. Introduction

An explainable artificial intelligence [1] (or XAI) is an artificial intelligence (AI) where humans can easily understand and analyze the actions performed by the AI. The idea behind using XAI is that the results given by AI models would be more acceptable to the end-user if the results have an explanation in layman's terms associated with them [2]. The recent advances in machine learning algorithms can produce an AI that can learn, decide, and act independently without any supervision. However, these AI systems cannot explain their output and actions to human users [1]. It is challenging to apply these models to high-stakes or government-regulated domains such as insurance, loans, mortgages, portfolio rebalancing, power plant operations, etc. XAI provides much-needed explanations for AI applications in such domains [3]. It tries to improve by providing reasoning behind the decisions made by an AI system.

Recent research has shown that rule-based fuzzy systems work well in implementing XAI models because fuzzy rules consist of words and IF–ELSE statements that contribute to the overall explainability of the AI model [3,4]. Autonomous XAI agents can be built using rule-based fuzzy systems that can work in a fast-paced environment. However, one of the major concerns while implementing XAI agents using rule-based fuzzy systems is the associated performance penalty. Along similar lines, an Explainable Fuzzy AI Challenge (XFC 2022) competition was launched, whose principal objective was to develop

a fully autonomous and optimized XAI algorithm that could play the Python arcade game "Asteroid Smasher" [5]. In the game, a 2-dimensional spacecraft moves around in an asteroid field, and the goal of the XAI agent is to terminate all the asteroids while avoiding collision with them.

This paper introduces a computationally efficient and most explainable solution to the competition, where we implement XAI agents using rule-based fuzzy systems. This recent work motivated us to explore the feasibility and performance of such XAI implementations, which use rule-based fuzzy inference systems at their core in a fast-paced environment [6,7]. Further, we wanted to explore which type-1 fuzzy inference system would be most efficient and perform better for such explainable artificial intelligence implementations.

In this research work, we tried to answer the following questions

- Can an AI agent be implemented using the fuzzy inference system and later be explained logically?
- Which type-1 fuzzy inference system is most computationally efficient and performs better in such an implementation?
- How does such an inference system help to attain higher efficiency?
- Are the generated explanations sensible and understandable to the end-user?

To answer these research questions, we took the implementation of an explainable AI system provided by the competition [5]. The game environment changes at 60 frames per second; hence, the XAI agent needs to perform its calculation within that time. Simply, the XAI agent code needs to complete its execution in less than 0.016 s to control the spacecraft efficiently. Further, we compared and experimented with the two most popular type-1 fuzzy inference systems, Mamdani and Takagi–Sugeno–Kang (TSK). We implemented the same XAI agent using both Mamdani and TSK inference systems to perform a fair comparison. We then benchmarked both implementations to determine which fuzzy inference system performs better and is computationally efficient.

The major contribution of the manuscript is as follows:

1. An XAI agent to work in a fast-paced environment using fuzzy inference systems is designed.
2. The objective is to find a computationally efficient and feasible way to implement these XAI agents using rule-based fuzzy systems.
3. The actions of the agents are explained in IF–ELSE statements, which are easier for an average user to understand without any technical expertise.
4. The proposed fuzzy-inference-systems-based XAI agent achieved an accuracy of 85.45%.

This paper is organized in the following way. Section 2 provides quick background material for the related methodologies used in this competition. Section 3 provides a step-by-step flow of the proposed approach, which includes the components of the competition and the utilized methodology. The results are also compiled in this section. Section 4 concludes the paper by presenting a detailed discussion of the outcomes and future works.

2. Literature Survey

There have been no past works in this area apart from the XFC competition in 2021 [8]. However, we provide a short literature study for the ready reference of the readers, which will also help them understand the methodologies behind the approaches used in the competition.

To start with, Mendel and Bonissone [3] have discussed why rule-based fuzzy inference systems are suitable for XAI implementation. The authors further presented a way to explain the output of rule-based fuzzy inference systems by taking an association of antecedents of the rule-base. They stated that triangular or trapezoidal membership functions for rule antecedents were more suitable for XAI implementation than Gaussian membership functions because the former partitioned the state space more unambiguously than the latter. Hagras [2] discussed the need for XAI and the efforts that are currently being researched to realize XAI or Whitebox/Transparent models. He also discussed the

feasibility of using Type-1 and Type-2 fuzzy inference systems to implement XAI because a layman user can easily understand the output of such fuzzy inference systems.

Gunning and Aha [1] discussed the DARPA's XAI program for developing and evaluating a wide variety of new ML techniques. The study included deep learning models, learning explainable features, creating more structured and interpretable models, and model induction techniques that can infer an explainable model from any black-box model. Chimatapu et al. [9] studied several approaches to implementing XAI and compared them in terms of accuracy and interpretability. They also proposed an alternate method of implementing XAI using fuzzy rule-based systems, which use IF–ELSE rules and are easy to understand for an average end-user. This alternate approach allows us to create AI systems that have a good balance between accuracy and interpretability. Ferreyra et al. [10] proposed a concise and incremental framework for developing XAI solutions in the telecommunication workforce allocation domain. They used type-2 fuzzy inference systems [11] to model the decision-making process in the workforce allocation domain, particularly for a goal-driven solution. The framework proposed also contained an intuitive explanatory user interface to depict the developed application's understanding.

Potie et al. [12] performed a case study on predicting lung cancer by implementing XAI using evolutionary fuzzy systems; it combined the good degree of understanding, comprehensibility, and explainability associated with fuzzy rule-based systems with the potential of evolutionary algorithms as the optimization technique for improving fuzzy rule-based systems. Experiments done by the author have shown that evolutionary fuzzy systems can be a very feasible solution as they achieve the highest test accuracy and the best explainable features. Kiani et al. [13] discussed the implementation of an XAI system using a type-2 fuzzy inference system for analyzing the fNIRS dataset to get a better insight into how a developing child's brain responds to different stimuli. The authors chose the type-2 fuzzy inference system because of its unique ability to model uncertainty in the input data, which was paramount in this work. Poli et al. [14] worked on semantic image annotation by modeling the problem as a fuzzy constraint satisfaction problem [15]. The authors also proposed an algorithm for semantic image annotation. The XAI system they developed also gives explanations for each annotation that, according to the survey conducted by them, are easy for an average end-user to understand.

As for the background of the 2021 competition, there was only a single agent and no restriction on the execution time [8]. However, in the present competition [7], we have multiple agents, and a time limit of 0.016 s is imposed. For each execution, if the controller takes more time than this, the controller's output is ignored. Further, in this study, we discuss why rule-based fuzzy inference systems are optimal for implementing explainable AI systems.

3. Proposed Approach, Competition Scenarios, and Outcomes

This section covers the step-by-step details of the competition framework and our (winning) proposed approach to providing an efficient solution. The following procedure is depicted in Figure 1.

Step 1: ***Agent Environment***

At first, we explore and analyze the agent environment to determine what inputs the environment provides and the actions the agent is allowed to perform. We use the Python implementation of a classic arcade game called fuzzy asteroids [16].

A two-dimensional spacecraft moves in the game environment to avoid collisions with numerous asteroids that appear. The asteroids have different shapes, sizes, and velocities. The spacecraft also has a weapon that can shoot straight ahead. If the projectiles emitted reach any target asteroids, they break into smaller pieces. The smallest asteroid pieces disappear after being hit by a bullet. The environment is 800 pixels high by 1000 pixels wide, with object wrapping enabled on the corners, i.e., a ship or asteroid that goes out of the game screen space will come back into it from the opposite side.

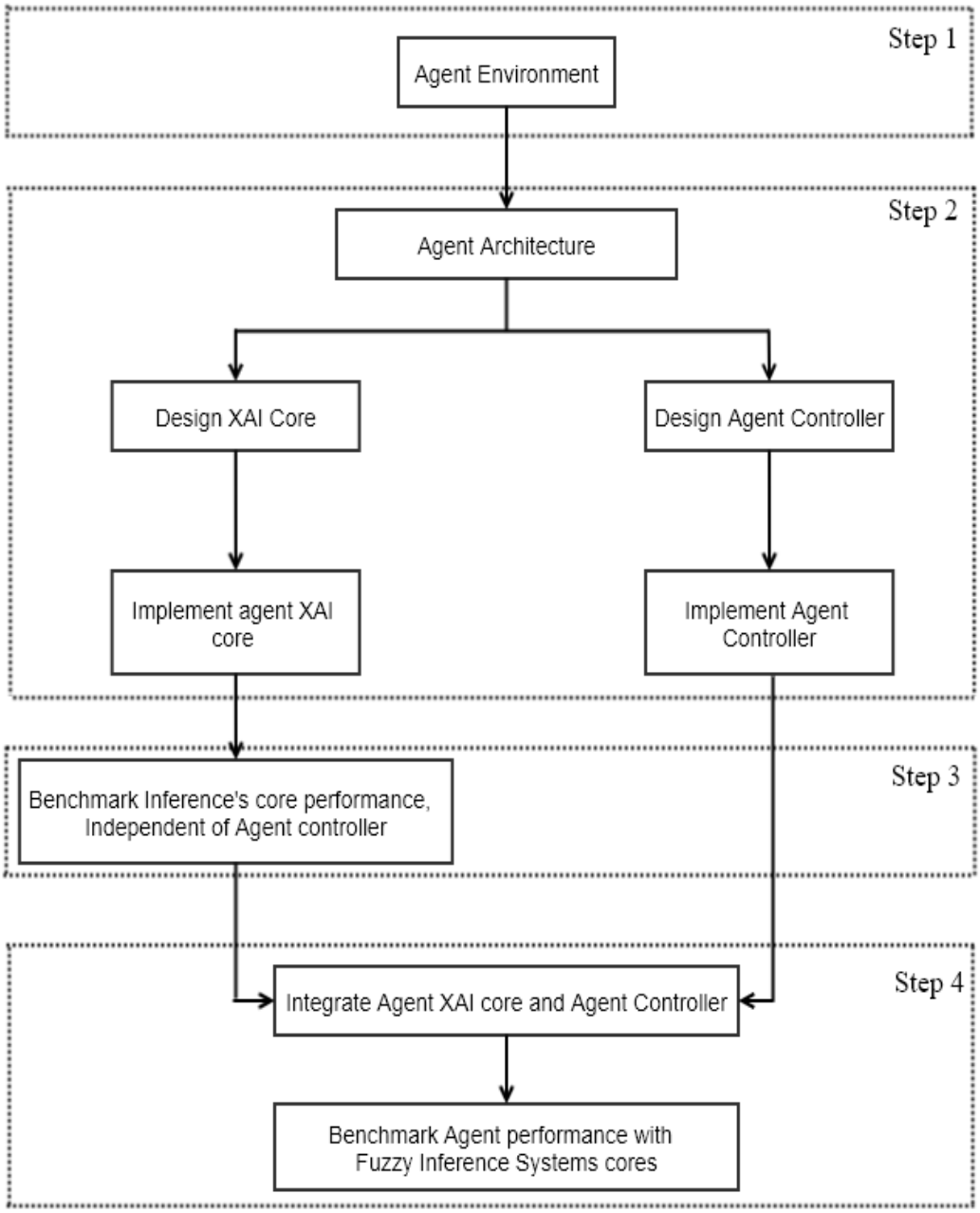

Figure 1. Flowchart of the competition framework and proposed methodology.

The agent environment provides the XAI agent with information about the ship and all the asteroids currently in the agent environment. The agent can control the ship's velocity, the ship's turning speed, and the ship's gun to fire bullets. Tables 1–3 summarize the inputs and available actions for the XAI agent.

Table 1. Available inputs for each asteroid on the screen.

Input Name	Description
position	Absolute position of the asteroid on the screen in a tuple (x-coordinate, y-coordinate).
velocity	Current velocity of the asteroids in a tuple (x-velocity, y-velocity) in meters/second.
size	The size of the asteroid.
angle	The angle at which the asteroid is moving.

Table 2. Available inputs for the ship.

Input Name	Description
is_respawning	Boolean variable, which tells if the ship is currently spawning or not.
position	The absolute position of the ship on the screen in a tuple (x-coordinate, y-coordinate).
velocity	Current velocity of the ship in a tuple (x-velocity, y-velocity) in meters/second.
angle	The angle at which the ship is moving.

Table 3. Available actions XAI agent can perform to control the ship.

Action Name	Description and Possible Values
thrust	The XAI agent can set this variable's value to change the ship's acceleration between −480 and 480 to accelerate backward or forward, respectively.
turning_rate	The XAI agent can set this variable's value to change the ship's turning speed between −180.0 and 180.0 to rotate the ship anti-clockwise or clockwise, respectively.
shoot	The XAI agent can call this method to make the ship shoot a bullet.

Step 2: ***Agent Design and Implementation***

The next step is to design the architecture of the XAI agent. The main challenge in designing the agent architecture would be to keep the XAI core separate as an independent module so that we can swap the fuzzy inference system XAI cores easily while performing the required experiments. We will be working independently on designing the XAI core and agent controller to keep these two primary components of our XAI agent independent. We will design the XAI core for a rule-based fuzzy inference system and then implement the XAI core using both Mamdani and TSK fuzzy inference systems. Another crucial component is to design and implement the agent controller such that it can treat the XAI core like an independent module. The agent controller will give specific inputs to the XAI core and then expect a particular output from the XAI core, which it will use to make certain decisions regarding the functioning of the XAI agent.

To simplify, the XAI agent consists of two components (see Figure 2); the first component is the XAI agent controller, and the second component is the XAI core. The XAI agent controller component is responsible for interfacing with the environment and making decisions based on the outputs of the XAI core. The XAI agent controller first gets the inputs from the agent environment and performs the required calculations. It passes the calculated values to the XAI core and returns the output from the XAI core. Then, it decides on the actions to take based on the result of the XAI core. In the end, it conveys the steps to be taken by the ship to the agent environment.

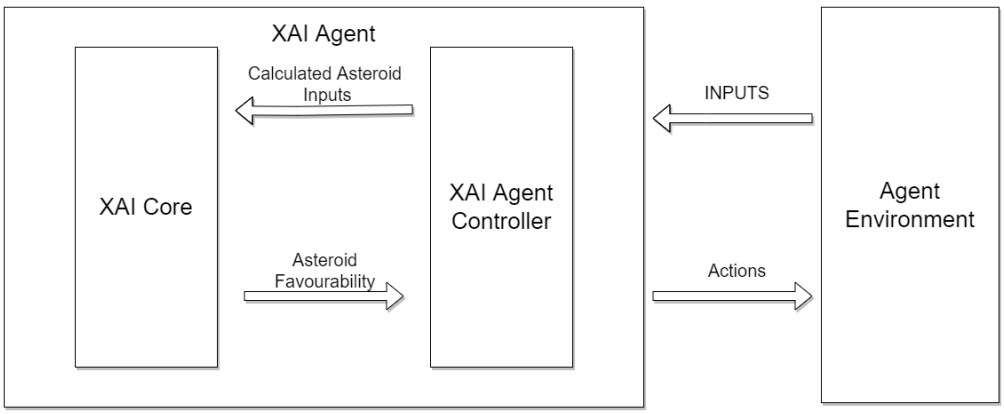

Figure 2. XAI agent block diagram.

The XAI core component is responsible for calculating the asteroid favorability values. Asteroid favorability is a numeric value assigned to each asteroid in the agent environment to signify the importance of targeting that particular asteroid by the XAI agent controller. The higher the asteroid's favorability, the more critical it is to target that asteroid at that moment. The value of asteroid favorability depends on two factors in the following order of importance:

1. How much of a threat does the asteroid currently pose to the ship?
2. How easy is it to target that particular asteroid at this moment?

XAI Agent Controller Design

Firstly, the agent controller uses the output of the XAI core to calculate which asteroid is currently the best target in the agent environment. The agent controller gets all the inputs from the agent environment and calculates the three inputs (see Table 4) for the XAI core to function. After calculating the inputs needed, the agent controller passes the information to the XAI core to obtain asteroid favorability. The controller selects the asteroid with the maximum favorability as the target asteroid.

Table 4. XAI core inputs.

Input	Description
asteroid distance	The distance between the asteroid and the ship.
asteroid size	The size of the asteroid.
asteroid orientation	The asteroid's orientation relative to the ship.

Then, the agent controller divides the agent environment into three zones to decide the agent's behavior (see Figure 3). The first zone (depicted in red) is the threat zone; if an asteroid is in the threat zone, the agent goes into defensive mode and tries to either shoot down the closest asteroid or move away from the closest asteroid. The second zone (depicted in yellow) is the targeting zone; if an asteroid is in the targeting zone, the agent goes into attack mode, selects the best asteroid to target, and shoots it down. The third zone (depicted in green) is the "search and destroy" zone; if an asteroid is in this zone, the agent goes into "search and destroy" mode and moves towards the best asteroid to target and shoots it down.

After finding out which asteroid is best to target and the behavior the agent needs to perform, the agent controller follows Algorithm 1 to decide on what actions to perform when the agent is in defensive mode, attacking mode, or "search and destroy" mode.

XAI Core Design and Implementation using Rule-Based Fuzzy Inference Systems

The XAI core (see Figure 4) takes three inputs, as shown in Table 4. The output of the XAI core is the asteroid favorability value that the agent controller uses. The process of calculating the asteroid favorability is not specified in the XAI core design, and it is decided while implementing the XAI core. The fuzzy inference system (see Figure 5) takes three inputs, as specified in the XAI core design. First, it fuzzifies the input values into their respective fuzzy sets (see Table 5) and uses their respective membership functions (see Figures 6–8).

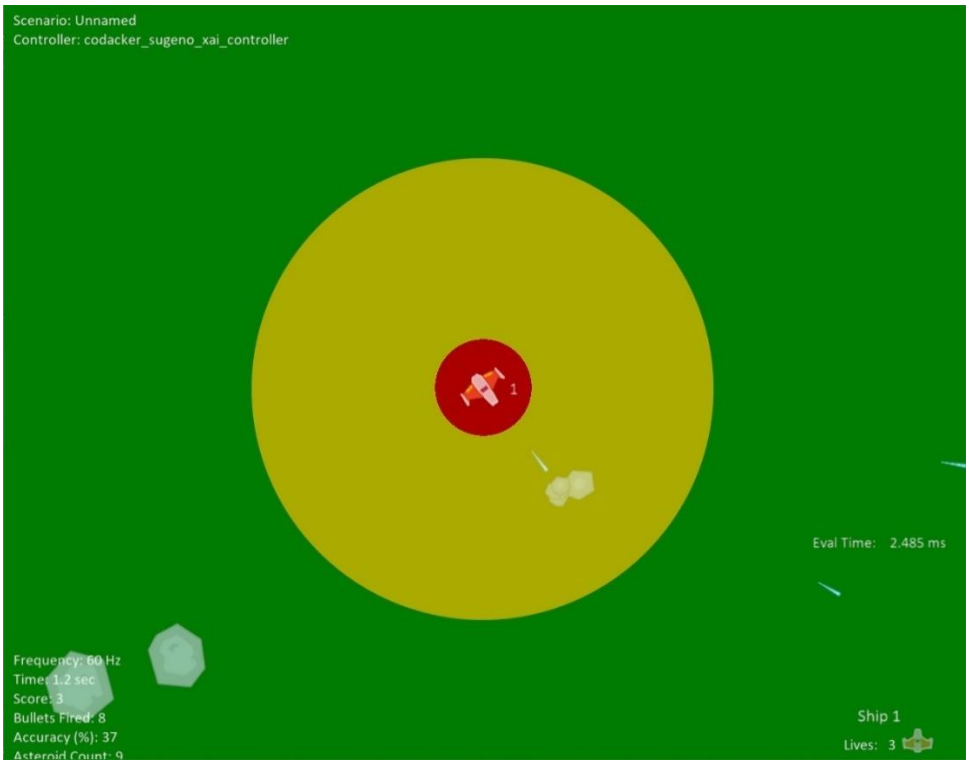

Figure 3. Agent zones and environment.

Algorithm 1: *Decide Agent's Action*

Input: agent_mode, target_asteroid
Output: action

```
if agent_mode = defensive then
    if agent is facing the target_asteroid then
        action ← fire bullet
    end
    else
        action ← move away from the target_asteroid
    end
end
else if agent_mode = attacking then
    if ship is facing the target_asteroid then
        action ← fire bullet
    end
    else
        action ← turn towards the target_asteroid
    end
end
else if agent_mode = search and destroy then
    if ship is facing the target_asteroid then
        action ← move towards the target_asteroid
    end
    else
        action ← turn towards the target_asteroid
    end
end
```

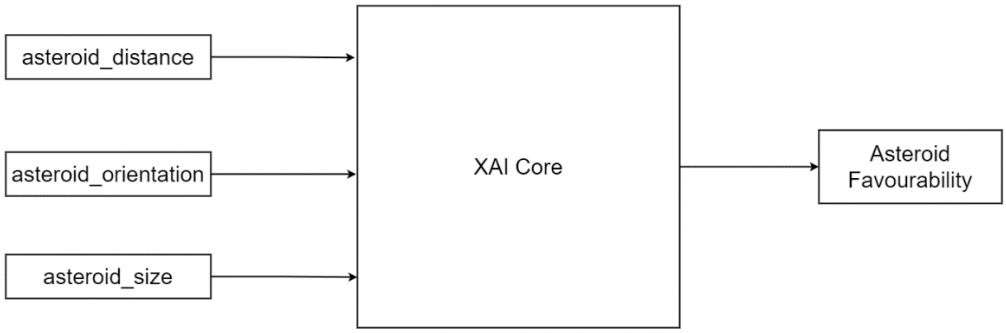

Figure 4. XAI Core block diagram.

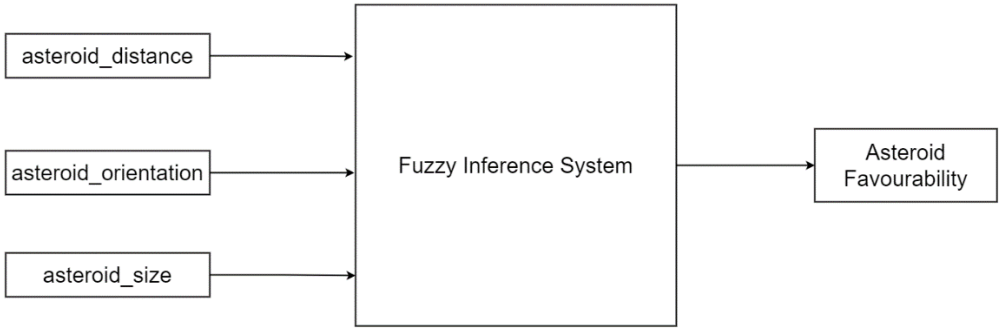

Figure 5. FIS block diagram.

Table 5. Fuzzy inference system inputs.

Input	Fuzzy Sets
asteroid distance	Near, Close, Far
asteroid size	Small, Medium, Large
asteroid orientation	In Sight, Normal, Out of Sight

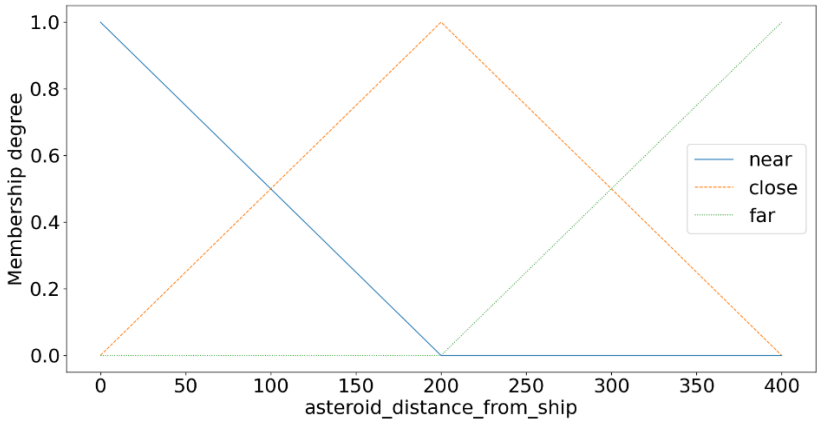

Figure 6. Asteroid distance fuzzy set.

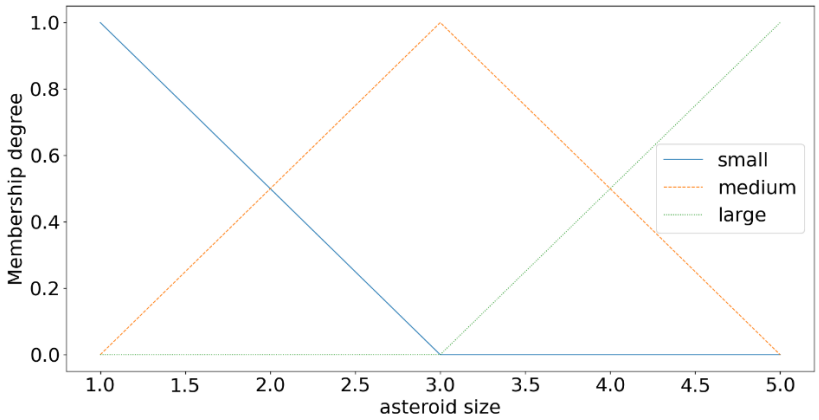

Figure 7. Asteroid size fuzzy set.

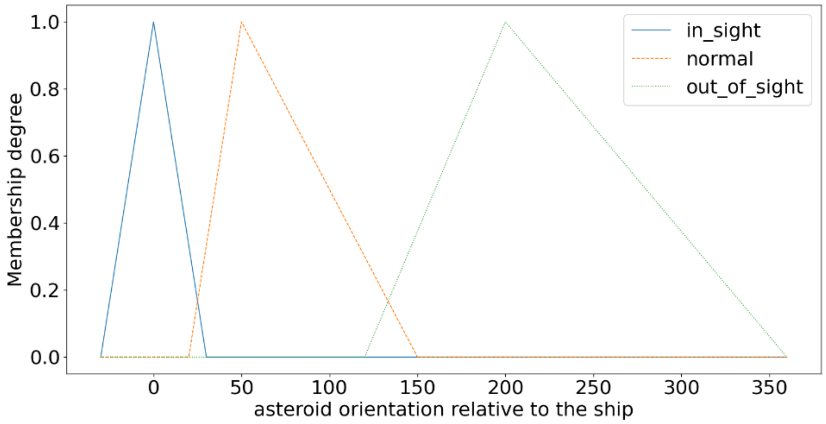

Figure 8. Asteroid orientation fuzzy set.

Trapezoidal and triangular (a special case of trapezoidal membership function) functions have been widely used in the literature for XAI implementation. They partition the state space more unambiguously than any other membership function [6]. Therefore, for our current implementation, we have used triangular membership functions. After conducting several experiments empirically, we identified the following rule-base to be the most efficient for our XAI core implementation using rule-based fuzzy inference systems.

1. IF (asteroid_size IS small) THEN (favorability IS high)
2. IF (asteroid_size IS medium) THEN (favorability IS medium)
3. IF (asteroid_size IS large) THEN (favorability IS low)
4. IF (asteroid_orientation IS insight) THEN (favorability IS high)
5. IF (asteroid_orientation IS normal) THEN (favorability IS medium)
6. IF (asteroid_orientation IS outofsight) THEN (favorability IS low)
7. IF (asteroid_distance IS near) THEN (favorability IS high)
8. IF (asteroid_distance IS close) THEN (favorability IS medium)
9. IF (asteroi_distance IS far) THEN (favorability IS low).

The first three rules are for asteroid size. If we hit a giant asteroid, it will split into multiple asteroids, increasing the number of asteroids in the agent environment. When the number of asteroids increases in the agent environment, the agent takes more time for its execution, reducing our computational performance. Hence, it is better to favor the smaller asteroids first and the large asteroids last.

The following three rules deal with asteroid orientation relative to the ship, i.e., how much we need to rotate our ship to face the ship's gun towards the asteroid. Therefore, it makes sense to favor the asteroid that is in sight (requires less rotation) first and the one that is out of sight (requires more rotation) last.

The last three rules deal with the asteroid's distance from the ship. An asteroid near the ship poses a more significant threat than one that is far away; therefore, it makes sense to favor the asteroid that is near first and the asteroid that is far last.

The fuzzy inference system calculates the asteroid favorability output by applying the fuzzy rule base to the input values. The output values are different for Mamdani and TSK inference systems. In Mamdani, the output is a fuzzy variable with defined membership functions (see Figure 9); in TSK, the output values are real numbers, as defined in Table 6. The Mamdani output calculation requires defuzzification on the fuzzy output value to get a real value. We use the center of area defuzzification method to get a concrete value between 1 and 10 for asteroid favorability. In the case of TSK, we calculate the output by taking a weighted average of the consequent of each rule, which is a real value by default; hence, no defuzzification is required.

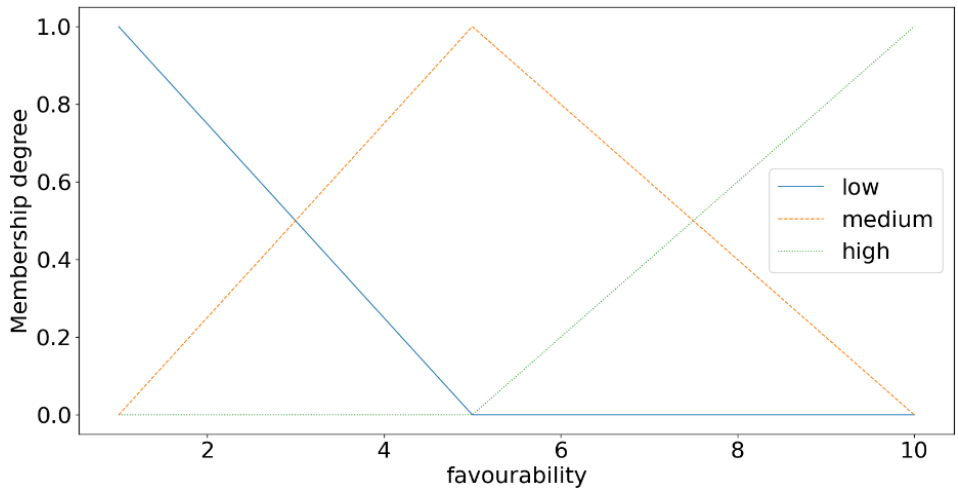

Figure 9. Asteroid favorability fuzzy set.

Table 6. Asteroid favorability output for Mamdani and TSK inference systems.

Asteroid Favorability	Mamdani Output Fuzzy Set	TSK Output Value
Low	Triangular function (1, 1, 5)	1
Medium	Triangular function (1, 5, 10)	5
High	Triangular function (5, 10, 10)	10

Step 3: ***Benchmarking FIS's core performance***

For the next step, we will benchmark the performance of the Mamdani and TSK XAI cores independent of the agent controller to find empirical evidence of which implementation is more computationally efficient for such implementations using Algorithm 2. Figure 10 shows the benchmark result, and clearly, TSK outperforms Mamdani in terms of execution time.

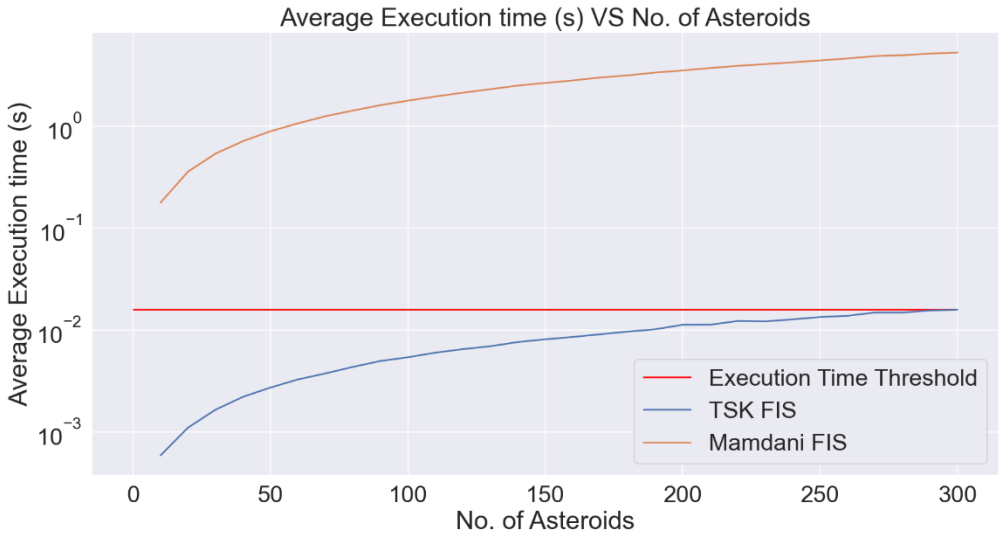

Figure 10. Mamdani vs. TSK execution time benchmark.

Algorithm 2: *XAI Core Execution Time Benchmark*

Input: xai_core, N
Output: execution_time_metrics

```
i ← 1
execution_time_metrics ← new array()
while i < N do
    xai_core_input_vector ← generate_random_inputs(i)
    t1 ← time.now()
    result ← xai_core.evaluate(xai_core_input_vector)
    evaluation_time ← time.now() − t1
    execution_time_metrics.append((i, evaluation_time))
end
return execution_time_metrics
```

Algorithm 3: *XAI Core Noise Benchmark*

Input: xai_core, N
Output: noisy_metrics

```
variances ← [0.00001, 0.0001, 0.001, 0.01, 0.1, 1]
noisy_metrics ← new array()
for variance in variances do
    xai_core_input_vector ← generate_random_inputs(N)
    original_outputs ← xai_core.evaluate(xai_core_input_vector)
    noise ← generate_random_noise(mean = 0, variance = variance, shape =
xai_core_input_vector.shape)
    noisy_input_vector ← xai_core_input_vector + noise
    noisy_outputs ← xai_core.evaluate(noisy_input_vector)
    mean_squared_error ←
calculate_mean_squared_error(original_outputs, noisy_outputs)
    noisy_metrics.append(variance, mean_squared_error)
end
return noisy_metrics
```

Step 4: ***Integrate Agent XAI core and controller and benchmarking***

In the next step, we will integrate the XAI core and agent controller such that the XAI cores are easily swappable for easy experimentation. We will then run benchmarks to evaluate agent performance for Mamdani and TSK implementations of XAI cores using Algorithm 3.

This benchmark was to identify which fuzzy inference system performs better when there is noise in the input data. Figure 11 shows the benchmark results, and here, both Mamdani and TSK fuzzy inference system XAI core implementations have a mean squared error of less than 10^{-1}. However, when the variance reaches 1, TSK output values changes by a significant factor, showing that TSK is sensitive when the input values are out of the noisy range. It depicts an actual change in the input values, which is a good thing for us as the change of 1 in the asteroid size input should be able to create a big difference in favorability value.

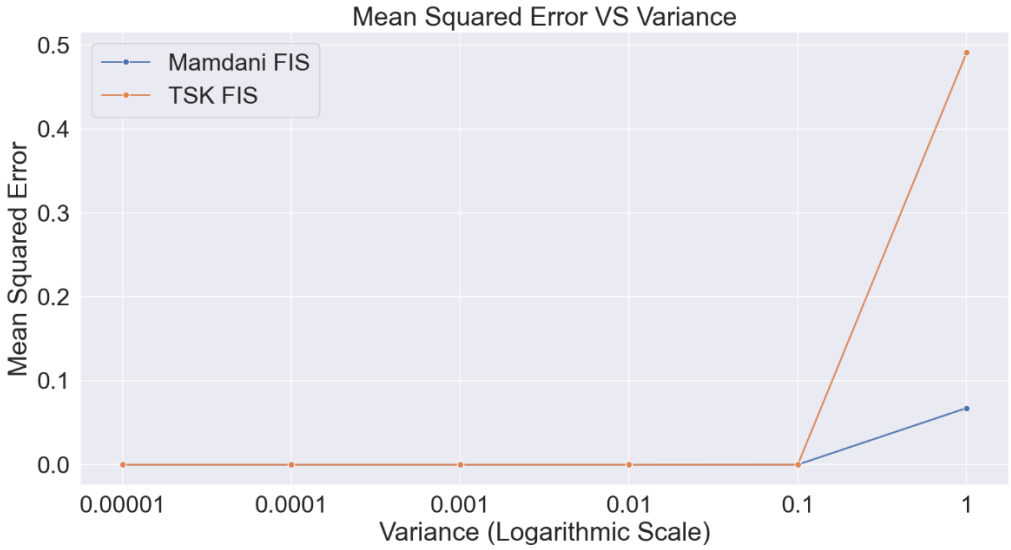

Figure 11. Mamdani vs. TSK noise benchmark.

We wanted to benchmark the agent's performance in the environment after integrating the XAI core implemented using rule-based fuzzy inference systems with the agent controller. We benchmarked the agent's performance using Algorithm 4 with both TSK and Mamdani XAI core implementations. The metrics obtained by this benchmark are summarized in Table 7. Clearly, in this benchmark, the TSK fuzzy-inference-system-based XAI core implementation outperforms the Mamdani fuzzy-inference-system-based XAI core implementation.

Table 7. Mamdani vs. TSK agent performance.

Performance Metric	Mamdani-Based XAI Agent	TSK-Based XAI Agent
Number of deaths	76	43
Number of asteroids destroyed	227	8427
Number of bullets fired	322	110,313
Average accuracy	57.27%	85.45%
Average execution time	0.0510 s	0.0016 s

Algorithm 4: *XAI Agent Benchmark using a given XAI core*

Input: xai_core
Output: agent_performance_metrics

```
scores ← new array()
scenarios ← get_predefined_scenarios()
N ← scenarios.length
execution_time_metrics ← new array()
xai_agent ← new agent(core = xai_core)
for i ← 0, i < N, i + + do
    environment ← new environment(scenario = scenarios[i], agent = xai_agent)
    agent_score ← environment.run()
    scores.append(agent_score)
end
agent_performance_metrics ← summarize_scores(scores)
return agent_performance_metrics
```

Based on our results and game performance, the proposed approach has been awarded the most computationally efficient and explainable solution in the XFC 2022 competition. The results of the competition were presented at the 2022 North American Fuzzy Information Processing Society Conference [17]. The results are available to the readers online [18].

4. Discussion, Conclusion, and Future Work

In this work, we have designed an XAI agent to work in a fast-paced environment using both Mamdani and TSK fuzzy inference systems at its XAI core to prove the feasibility of using rule-based fuzzy systems for XAI agent implementations. We further conducted experiments to gather empirical evidence to determine which type-1 fuzzy inference system performs better in a fast-paced environment. The Mamdani fuzzy inference system output calculation process contains an extra step for the defuzzification of the output values from the fuzzy output. The defuzzification process includes complex mathematical operations and is very time-consuming even for computers to perform. In the TSK output calculation process, there is no need for the defuzzification step, which gives an edge to TSK in terms of computational efficiency. Due to this extra edge in computational efficiency, the TSK-based XAI core implementation performs marginally better than the Mamdani-based XAI core implementation.

For each execution during the gameplay, the XAI agent outputs timestamped explanations for the action it chooses (see Figure 12). The output of the XAI core is the favorability value governed by the set of rules currently being applied for the calculations. In Figure 12, we can see the applicability or firing rate of each rule currently applicable in calculating the favorability value. For example, consider the output "*Targeting asteroid at (752.34, 486.60) because of rules: ['Rule 1': 1.0, 'Rule 4': 0.48, 'Rule 8': 0.86, 'Rule 9': 0.14]*". Here, this output tells us that Rule 1 was 100% applicable, Rule 4 was 0.48% applicable, Rule 8 was 86% applicable, and Rule 9 was 14% applicable. Now, cross-referring the rule-base in Section 3, we can understand the decision taken by the XAI module to target this asteroid. With the timestamped recording of the gameplay and these timestamped explanations, anyone can analyze the explanations and find out exactly why the agent decided to act at any given time. The explanation behind choosing a target asteroid is explained in terms of the applicability of each rule from the fuzzy rule base, which is just a set of IF–ELSE statements and is easy for an average user to understand without any technical expertise required.

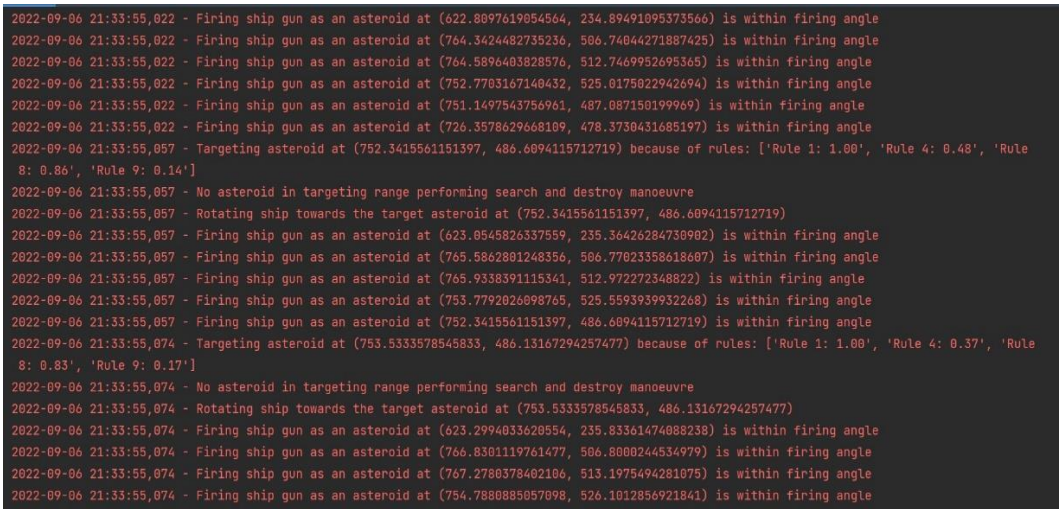

Figure 12. XAI agent explanations.

Further, the experiments showed that the TSK-based XAI agent performed 30 times better than the Mamdani-based XAI agent in computational efficiency. Regarding the agent's performance, the TSK-based XAI agent had an accuracy of 85.45%, while the Mamdani-based XAI agent had an accuracy of 57.27%. The number of deaths for the TSK-based XAI agent was 43. In contrast, the Mamdani-based XAI agent died 75 times. The TSK-based XAI agent was able to destroy 8427 asteroids, while the Mamdani-based XAI agent could only destroy 227 asteroids. The empirical evidence that we have gathered shows that it is better to use TSK fuzzy inference systems for rule-based XAI core implementations whenever the XAI application requires time-critical execution and better performance simultaneously. We further show that the actions taken by the XAI agent using a rule-based fuzzy inference system at its XAI core implementation could easily be explained in plain English using IF–ELSE rules. It is easier for an average user to understand without any technical expertise as a prerequisite.

This work faced several challenges that are to be considered during implementation. One of the major challenges was writing efficient code so that it would not become a bottleneck for the XAI system. Further, it was essential to design the XAI module flexibly so that we could plug and play different implementations of the XAI and perform benchmarks on them. In addition, it was also challenging to maintain consistency between the interface of the XAI module and the rest of the controller so that XAI implementations would follow the guidelines for the interface.

To conclude, this work helps those new to explainable AI and fuzzy inference systems to understand the concepts in a fun, hands-on learning manner. Further, the experimental approach that we have used can be used as a baseline to compare the efficiency and performance of various XAI agent implementations and find empirical evidence of which XAI implementation performs better.

In this research work (based on competition), we have only explored rule-based fuzzy inference system XAI core implementations. However, the proposed experimental approach and benchmarking algorithms can be used to perform experiments to compare rule-based fuzzy inference system XAI core implementations with any other XAI core implementation to find out which implementation is best in terms of performance and explainability.

In the outcome of this study, the XAI explanations are only text-based. These explanations can be extended further to show visual explanations in real-time for the agent's actions, which will further add to the explainability of the agent as visual explanations are much easier to analyze than textual explanations for an average human being.

Author Contributions: Conceptualization: A.K.S. and P.K.M.; Methodology: S.M. and A.K.S.; Validation: S.M.; Resources: S.M. and A.K.S.; Data curation: S.M. and A.K.S.; Writing—S.M. and A.K.S.;

Writing—review and editing: A.K.S. and P.K.M.; Supervision: A.K.S. and P.K.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data and codes are provided by the organizers of XFC 2022. It is Available online: https://xfuzzycomp.pythonanywhere.com/ (accessed on 28 July 2022). Available online: https://pypi.org/project/fuzzy-asteroids/ (accessed on 28 July 2022). Available online: https://ceas.uc.edu/academics/departments/aerospace-engineering-mechanics/artificial-intelligence-competition/archive.html (accessed on 28 July 2022).

Acknowledgments: We are grateful to the organizers of the Explainable Fuzzy AI Challenge (XFC 2022) and North American Fuzzy Information Processing Society for presenting the problem and related material. The regular tutorials and slack discussions helped in clarifications of the problem.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Gunning, D.; Aha, D. DARPA's explainable artificial intelligence (XAI) program. *AI Mag.* **2019**, *40*, 44–58.
2. Hagras, H. Toward human-understandable, explainable AI. *Computer* **2018**, *51*, 28–36. [CrossRef]
3. Mendel, J.M.; Bonissone, P.P. Critical thinking about explainable ai (XAI) for rule- based fuzzy systems. *IEEE Trans. Fuzzy Syst.* **2021**, *29*, 3579–3593. [CrossRef]
4. Shukla, A.K.; Smits, G.; Pivert, O.; Lesot, M.J. Explaining data regularities and anomalies. In Proceedings of the 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Glasgow, UK, 19–24 July 2020; IEEE: Piscataway, NJ, USA; pp. 1–8.
5. Available online: https://ceas.uc.edu/academics/departments/aerospace-engineering-mechanics/artificial-intelligence-competition.html (accessed on 28 July 2022).
6. Yang, L.H.; Liu, J.; Ye, F.F.; Wang, Y.M.; Nugent, C.; Wang, H.; Martínez, L. Highly explainable cumulative belief rule-based system with effective rule-base modeling and inference scheme. *Knowl. Based Syst.* **2022**, *240*, 107805. [CrossRef]
7. Mamdani, E.H.; Assilian, S. An experiment in linguistic synthesis with a fuzzy logic controller. *Int. J. Hum. Comput. Stud.* **1999**, *51*, 135–147. [CrossRef]
8. Available online: https://ceas.uc.edu/academics/departments/aerospace-engineering-mechanics/artificial-intelligence-competition/archive.html (accessed on 28 July 2022).
9. Chimatapu, R.; Hagras, H.; Starkey, A.; Owusu, G. Explainable AI and fuzzy logic systems. In *Theory and Practice of Natural Computing, Proceedings of the 7th International Conference, TPNC 2018, Dublin, Ireland, 12–14 December 2018*; Springer: Cham, Switzerland, 2018; pp. 3–20.
10. Ferreyra, E.; Hagras, H.; Kern, M.; Owusu, G. Depicting Decision-Making: A Type-2 Fuzzy Logic Based Explainable Artificial Intelligence System for Goal-Driven Simulation in the Workforce Allocation Domain. In Proceedings of the 2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), New Orleans, LA, USA, 23–26 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
11. Shukla, A.K.; Banshal, S.K.; Seth, T.; Basu, A.; John, R.; Muhuri, P.K. A bibliometric overview of the field of type-2 fuzzy sets and systems [discussion forum]. *IEEE Comput. Intell. Mag.* **2020**, *15*, 89–98. [CrossRef]
12. Potie, N.; Giannoukakos, S.; Hackenberg, M.; Fernandez, A. On the need of interpretability for biomedical applications: Using fuzzy models for lung cancer prediction with liquid biopsy. In Proceedings of the 2019 IEEE international conference on fuzzy systems (FUZZ-IEEE), New Orleans, LA, USA, 23–26 June 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1–6.
13. Kiani, M.; Andreu-Perez, J.; Hagras, H.; Filippetti, M.L.; Rigato, S. A Type-2 Fuzzy Logic Based Explainable Artificial Intelligence System for Developmental Neuroscience. In Proceedings of the 2020 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Glasgow, UK, 19–24 July 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–8.
14. Poli, J.P.; Ouerdane, W.; Pierrard, R. Generation of textual explanations in XAI: The case of semantic annotation. In Proceedings of the 2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Luxembourg, Luxembourg, 11–14 July 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 1–6.
15. Lin, K.; Liu, Y.; Lu, P.; Yang, Y.; Fan, H.; Hong, F. Fuzzy constraint-based agent negotiation framework for doctor-patient shared decision-making. *BMC Med. Inform. Decis. Mak.* **2022**, *22*, 218. [CrossRef] [PubMed]
16. Available online: https://pypi.org/project/fuzzy-asteroids/ (accessed on 26 July 2022).
17. Available online: https://nafips2022.cs.smu.ca/ (accessed on 28 July 2022).
18. Available online: https://xfuzzycomp.pythonanywhere.com/ (accessed on 30 July 2022).