Group Decision Support System Simulator Based on Information Fusion and AI Methods

Summary of Responses and Answers

Vahid Moeinifar

Group Decision Support System Simulator Based on Information Fusion and AI Methods

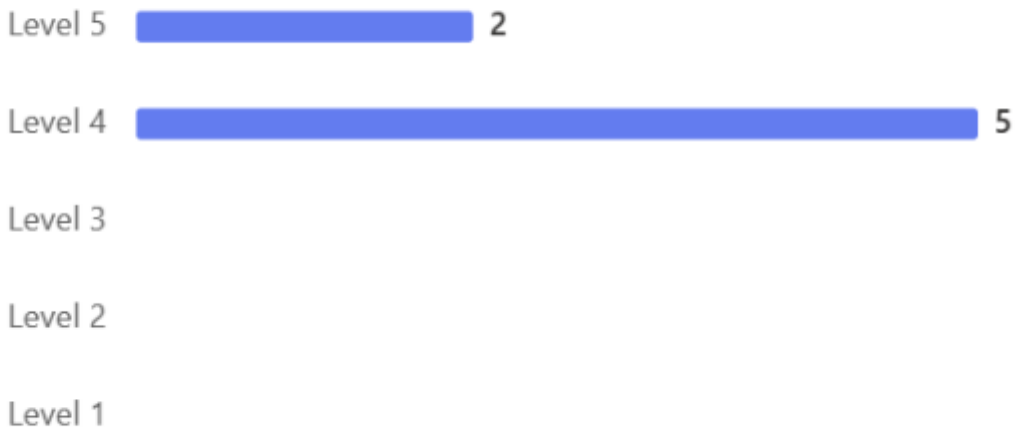
1. How clear is your understanding of what a Decision Support System (DSS) is?

Group Decision Support System Simulator Based on Information Fusion and AI Methods

2. How well do you understand the main goal of the GDSS Simulator project?

4.29

Average Rating



Group Decision Support System Simulator Based on Information Fusion and AI Methods

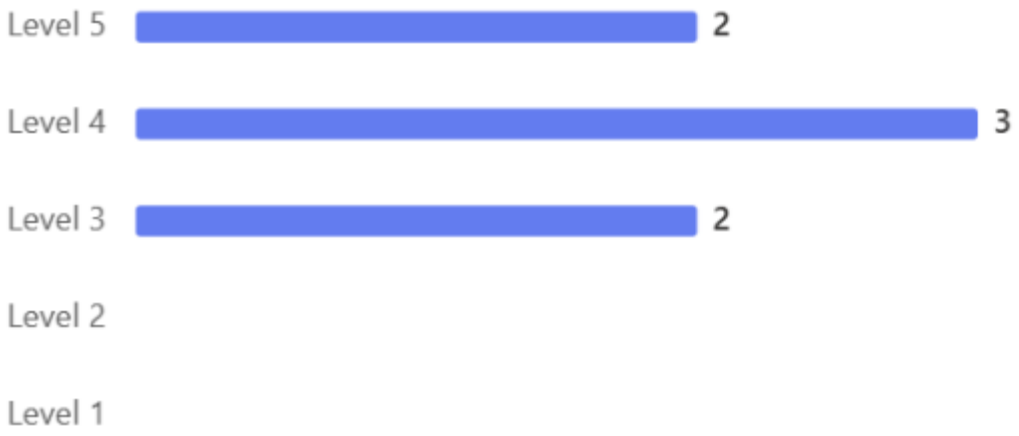
3. How easy is it to understand the overall system architecture (Python + C++ + Qt Quick + MATLAB)?

Group Decision Support System Simulator Based on Information Fusion and AI Methods

4. How useful do you think such a simulator could be for supporting group decision-making in real applications?

4.00

Average Rating



Which part of the project is not clear for you?

Your Question: You should talk more about various AI models you're using and the fusion process. Are you using multiple instances of the same model simultaneously, or are you using many different models at the same time?

My Answer: Good question. The system utilizes several different heterogeneous models rather than just repeated instances of the same one. Each decision-making agent is able to use a different AI method (for example, Bayesian inference, weighted neural fusion, or fuzzy logic) depending on the type of its data and the level of its uncertainty. After that, the fusion layer combines their results with an information fusion framework — merging probabilistic and AI-based weighting to come up with a collective group decision.

... Finally I updated my Description MD file based on your suggestion! Thank you!

Which part of the project is not clear for you?

Your Question: I think it is not easy how to use this system in a real world scenario in a real problem.

My Answer: Here are some real-world examples of how this GDSS simulator idea might be utilized:
1- Disaster Response Planning:Multiple agents (for instance, weather models, terrain analysis, and rescue teams) offer uncertain inputs. The system merges these to determine the safest and quickest evacuation routes.
2- Medical Diagnosis Support:Various AI models or experts interpret patient data (imaging, lab results, and symptoms). The system combines their outputs to provide a more trustworthy diagnostic recommendation.
3- Smart Manufacturing: Sensors and predictive AI models keep track of machine health. The GDSS merges the uncertain sensor readings and expert evaluations to decide when to carry out maintenance or to stop the machine from breaking down.

Such instances demonstrate how the simulator can be utilized as a collaborative decision-making system development environment for the real world before actually deploying them in operation.

... Finally I upload a detaild presentation of real-world example and current companies that use DSS! Here is the link.

Which part of the project is not clear for you?

Your Question: What data would you feed to train an AI models? What is your expectations on the output?

My Answer: For this simulator, the training data depends on the decision domain, but generally it includes:

- **Agent inputs:** numerical or categorical data representing each decision-maker's observation (e.g., sensor readings, expert scores, probabilities).
- **Uncertainty measures:** confidence levels or error margins for each agent's input.
- **Ground truth or consensus labels:** past group decisions or validated outcomes to train supervised AI models.

Do you have any suggestion to make the project better? (Optional)

Your Question: I'm not sure why use C++ for the user facing code, when you already do your processing in python. This is inversion of what is usually done, C++ for the heavy lifting and combine it with some scripting language for flexibility. Although if you're already familiar with QML and want to work in it, I understand the decision. I just feel there are too many layers.

My Answer: Designing a user interface can be done in many ways, but **Qt Quick** really stands out by striking a great balance between performance, flexibility, and how quickly you can get a design up and running. With **QML**, the UI layer becomes incredibly dynamic and uses the **GPU** to speed things up, which lets you create smooth, real-time visualizations, something that's super important for interactive decision simulations.
Even though the core of the computations runs in Python, using C++ as the backend through Qt gives solid integration between the languages and makes data sharing between different parts of the system efficient. Basically, C++ keeps things fast and organized, Python takes care of the AI logic, and QML gives you a snappy, modern interface. This setup keeps the whole system neatly separated into modules, making it easy to update and expand.

Vahid Moeinifar

Group Decision Support System Simulator Based on Information Fusion and AI Methods

Feel free to ask more questions...

vmoeinifar@agh.edu.pl