ReclAIm: A multi-agent framework for degradation-aware performance tuning of medical imaging AI

Research paper

Eleftherios Tzanis, PhD^{1,*}, Michail E. Klontzas, MD, PhD^{1,2,3,*}

- 1. Artificial Intelligence and Translational Imaging (ATI) Lab, Department of Radiology, School of Medicine, University of Crete, Heraklion, Crete, Greece
- 2. Computational Biomedicine Laboratory, Institute of Computer Science Foundation for Research and Technology Hellas (ICS FORTH), Heraklion, Crete, Greece
- 3. Division of Radiology, Department of Clinical Science, Intervention and Technology (CLINTEC), Karolinska Institute, Huddinge, Sweden

*Corresponding authors

Addresses for correspondence

Eleftherios Tzanis, PhD

Postdoctoral Researcher
Artificial Intelligence and Translational Imaging (ATI) Lab
Department of Radiology, School of Medicine,
University of Crete, Voutes, 71003, Heraklion, Crete, Greece
E-mail: tzaniseleftherios@gmail.com; etzanis@uoc.gr
ORCID: 0000-0003-0353-481X

Michail E. Klontzas, MD, PhD

Assistant Professor of Radiology
Artificial Intelligence and Translational Imaging (ATI) Lab
Department of Radiology, School of Medicine,
University of Crete, Voutes, 71003, Heraklion, Crete, Greece
Tel: +30 2811391351 E-mail: miklontzas@gmail.com; miklontzas@uoc.gr
ORCID: 0000-0003-2731-933X

Abstract

Ensuring the long-term reliability of AI models in clinical practice requires continuous performance monitoring and corrective actions when degradation occurs. Addressing this need, this manuscript presents ReclAIm, a multi-agent framework capable of autonomously monitoring, evaluating, and fine-tuning medical image classification models. The system, built on a large language model core, operates entirely through natural language interaction, eliminating the need for programming expertise. ReclAIm successfully trains, evaluates, and maintains consistent performance of models across MRI, CT, and X-ray datasets. Once ReclAIm detects significant performance degradation, it autonomously executes state-of-theart fine-tuning procedures that substantially reduce the performance gap. In cases with performance drops of up to -41.1% (MRI InceptionV3), ReclAIm managed to readjust performance metrics within ±1.5% of the initial model results. ReclAIm enables automated, continuous maintenance of medical imaging AI models in a user-friendly and adaptable manner that facilitates broader adoption in both research and clinical environments.

1. Introduction

In recent years, the development and adoption of AI models in clinical practice has accelerated [1–3]. An important challenge that arises is ensuring that these systems maintain their performance over time and across test cases. A model that performs well during its development and evaluation phases may degrade in the field, due to factors such as data distribution shifts, variations in imaging quality, modifications in imaging equipment, or changes in clinical protocols [4–6].

Monitoring the continuing performance of AI models, alerting users when degradation is detected, and implementing corrective interventions are essential for patient safety and for supporting the long-term reliability of AI systems in routine clinical settings. Continuous performance monitoring is also emphasized in the EU AI Act (Article 72 on post-market monitoring), which highlights the importance of post-market surveillance for high-risk AI systems to ensure their ongoing safety and compliance [7].

Agentic systems have recently emerged as AI frameworks capable of automating complex tasks in medical AI, ranging from data preprocessing and radiomic feature extraction to model development, evaluation, and deployment [8]. These systems embed a large language model (LLM) as the core reasoning engine and interface mechanism with the user, while invoking tools to interact with the computational environment or external resources.

The aim of this study was to develop and benchmark ReclAIm, a multi-agent framework that monitors the performance of medical image classification models, detects degradation, and when necessary, autonomously executes fine-tuning to mitigate performance loss. The system was designed to operate solely through natural language interaction with the user, making its use straightforward for both technical and non-technical scientific personnel.

2. Materials and Methods

2.1 Agentic system

A multi-agent system capable of monitoring the performance of medical image classification models and triggering continual learning when necessary was developed using the smolagents library [9]. **Figure 1** provides an overview of the system developed herein. The architecture consists of a master agent that orchestrates a team of task-specific agents, each having at its disposal a series of strictly defined python tools:

- I. The *image classification agent* is responsible for training and deploying a variety of deep learning models, including ResNet [10], InceptionV3 [11], VGG16 [12], and EfficientNet [13]. This agent can be used by users who wish to train custom deep learning models with in-house data. It utilizes dedicated python tool classes that support model training and deployment, provide configurable augmentation pipelines ranging from basic to advanced or custom transformations, and implement multiple strategies for handling class imbalance, including weighted or focal loss functions as well as overor under-sampling.
- II. The *performance comparison agent* evaluates inference outputs against the corresponding test results to assess model accuracy and generalizability. It compares a series of global metrics such as accuracy, balanced accuracy, area under the ROC curve, precision, recall, and F1-score, as well as per-class performance, to identify potential degradation when a model is applied to new data. By quantifying considerable declines and generating recommendations, including fine-tuning or focused retraining on underperforming classes, the agent provides a mechanism for continuous monitoring and maintenance of deployed models.
- III. The *fine-tuning agent* adapts and optimizes existing models when performance degradation is detected. It applies strategies such as full or partial layer retraining, head-

only adaptation, and gradual unfreezing, while supporting differential learning rates and optimizer reinitialization [14, 15]. Moreover, it integrates data augmentation, class imbalance handling and catastrophic forgetting [16–18] prevention to preserve accuracy on previously learned classes.

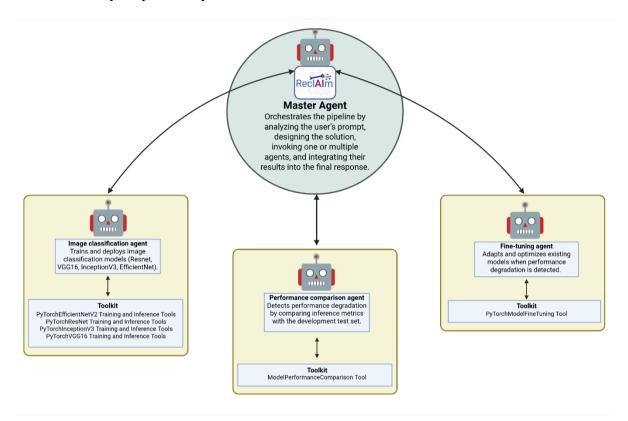


Figure 1. Master agent and task specific agents with their respective toolkits

The operation of the agentic system relies on the system prompt, which serves as the permanent set of instructions guiding its behavior. These instructions define the role of each agent, the tools available to them, and the prompting strategy used. In this work, the ReAct prompting approach [19] was adopted, where the system operates in a continuous cycle of "thinking-acting-observing". **Figure 2** illustrates an example of how a user interacts with the system to complete a task. At its core, the system consists of the system prompt, the set of task-specific agents, and their associated tools. The user communicates with the master agent in natural language, and the input to the LLM consists of the user prompt, the system prompt, and any

information stored in memory. The LLM first analyzes this input and generates a completion. In the initial steps, the output consists of the model's reasoning, outlining a potential solution to the task. Based on this reasoning, the LLM then performs an action, typically invoking a task-specific agent and producing code to call the relevant tool. The results of this action, along with the system's prior outputs, are stored in memory and provided as input for the next iteration. The LLM then evaluates these observations to either return the final result to the user or, if errors occur or the task remains incomplete, to initiate a new cycle of "thinking-acting-observing". The LLM used as the core reasoning engine of the developed system was ChatGPT-4.1.

2.2 Datasets

To cover a broad spectrum of imaging modalities, we selected three publicly available datasets: a brain tumor MRI dataset [20], a chest CT dataset for COVID-19 diagnosis [21], and a chest X-ray dataset [22] for pneumonia detection. The brain tumor MRI dataset [20] contains 7,023 images classified into four categories: glioma, meningioma, pituitary tumor, and no tumor. The chest CT dataset [21] consists of 2,482 scans. Among them, 1,252 scans are positive for SARS-CoV-2 infection, while 1,230 scans correspond to non-infected patients. The PneumoniaMNIST dataset [22] from the MedMNIST collection was used for the X-ray modality. It includes 5,645 chest X-ray images annotated for binary classification into pneumonia (4093 images) and normal cases (1552 images).

Each dataset was divided into three subsets: model development (60%), inference (20%), and fine-tuning (20%). Within the model development subset, images were further partitioned into training (70%), validation (15%), and test (15%) sets, maintaining the original class distribution. The inference subset was stored in a flat folder with an accompanying CSV file

containing the image-to-label mapping, while the fine-tuning subset was organized into classspecific folders to facilitate additional training when needed.

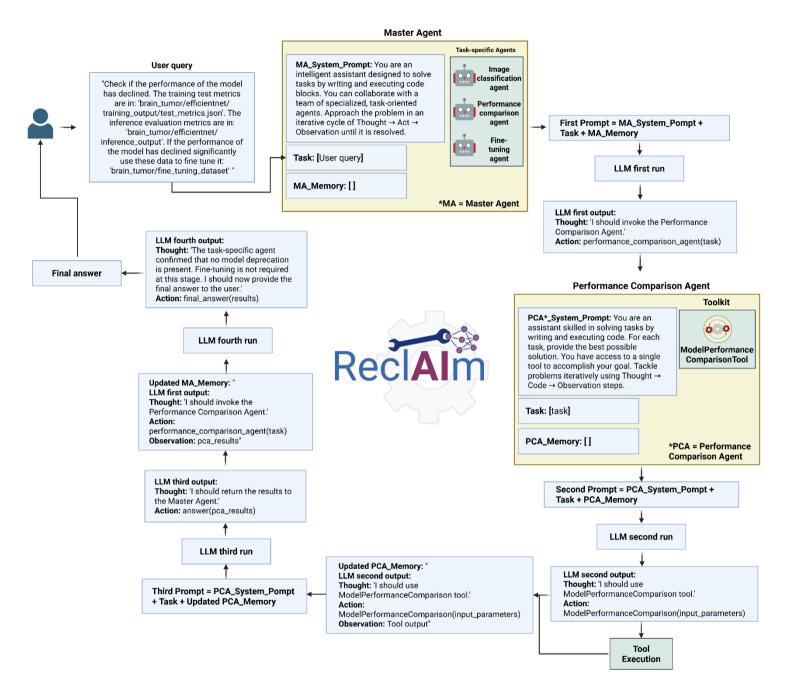


Figure 2 Flowchart demonstrating an example of a complete agentic workflow indicating the interaction between ReclAIm and the user

2.3 Experimental setup

To evaluate the functionality of the developed system and its ability to assess the performance of medical image classification models, the following strategy was employed.

First, queries were created and provided as input to the master agent to initiate model development. Specifically, the system was instructed to use the model development subsets of each imaging modality to train ResNet50, VGG16, EfficientNet, and InceptionV3 models. During this step, the master agent invoked the image classification agent, which in turn employed the respective tool classes to train and evaluate the models. Training was performed using the training and validation partitions of the model development data, and evaluation was carried out on the corresponding test partition. The results obtained on the test data were later compared with inference results on unseen data to assess potential model degradation. Next, prompts were issued to the master agent to:

- I. Use the trained models to classify the inference subsets of each modality and compute evaluation metrics (accuracy, precision, recall, F1-score) using the available ground truth labels.
- II. Detect possible model degradation by comparing performance on the model development test data with that on the inference subset.
- III. Analyze the comparison results and determine whether fine-tuning was required, as well as recommend an appropriate fine-tuning strategy.
- IV. If necessary, initiate fine-tuning using the fine-tuning dataset and retrain the degraded model.
- V. Re-evaluate the fine-tuned model on the inference data and compare the results with the initial test performance to verify improvement.

Representative prompts given to the master agent for model development, inference, and degradation detection with fine-tuning are shown below.

Example prompt for model development:

'Train a classification efficientnet model. The train, validation and test datasets:

"splitted data/brain tumor/model development". Number of classes 4. Set patience to 5 and

number of epochs to 50. Output directory:

"tests/model development/brain tumor/efficientnet/training output"."

Example prompt for inference:

'Use the efficientnet model available here:

"tests/model_development/brain_tumor/efficientnet/training_output", to classify the images in this folder: "splitted_data/brain_tumor/inference_dataset/inference_test". The number of classes is 4. Use ground truth labels to evaluate the

predictions:"splitted_data/brain_tumor/inference_dataset/inference_labels.csv". Save the evaluation output in this directory:

"tests/model development/brain tumor/efficientnet/inference output".'

Example prompt for degradation detection and fine-tuning:

'Check if the performance of the model has declined. The training test metrics are in:

"tests/model_development/brain_tumor/efficientnet/training_output/test_metrics.json".

The inference evaluation metrics are in:

"tests/model development/brain tumor/efficientnet/inference output/metrics.json".

Output folder: "tests/compare performance/brain tumor/efficientnet/".

If the performance of the model has declined significantly, use these data to fine tune it:

"splitted data/brain tumor/fine tuning dataset/".

Path to the model:

"tests/model_development/brain_tumor/efficientnet/training_output/best_model.pt".

Path to the config file:

"tests/model development/brain tumor/efficientnet/training output/model config.json".

Save the fine tuned model in: "tests/fine tuned models/brain tumor/efficientnet".'

In each case, the master agent received the inputs, analyzed the task, devised a strategy, and invoked the appropriate task-specific agent. This workflow enabled the evaluation of the

master agent's ability to both invoke individual task-specific agents and to orchestrate more complex, multi-step processes such as invoking the comparison agent, analyzing its outputs, and subsequently activating the fine-tuning agent when required.

2.4 System monitoring

An important component of the system evaluation was the inspection of the execution traces of the master agent and the task-specific agents. This step ensured that the master agent invoked the correct task-specific agents, that each agent selected the appropriate tool for its assigned task, and that the inputs and outputs of every interaction were consistent with the intended workflow. To achieve this, we integrated the OpenTelemetry framework [23] into the system. OpenTelemetry is an open-source observability framework that provides standardized methods for collecting, processing, and exporting telemetry data such as traces, metrics, and logs. In our implementation, OpenTelemetry was used to capture detailed traces of agentic interactions, including the sequence of invocations, the flow of inputs between agents and tools, and the outputs generated at each iteration. The integration was implemented through the tracing utilities provided smolagents library (https://huggingface.co/docs/smolagents/v1.21.2/en/tutorials/inspect runs#inspecting-runswith-opentelemetry). This allowed us to instrument the system with minimal overhead, while providing fine-grained insights into the reasoning and decision-making process of the master agent as well as the operations of the task-specific agents. These traces were exported in a standardized format, enabling both real-time inspection during development and systematic analysis for post-experiment evaluation.

3. Results

3.1 Classification model development

In response to the user queries, the agentic system successfully developed and evaluated 12 classification models: EfficientNet, InceptionV3, ResNet50, and VGG16 for each of the three datasets. In every case, the master agent invoked the appropriate task-specific agent, which analyzed the task and executed the training workflow. The agent first utilized the training tool, inspected the dataset, verified class distribution, and selected suitable configurations. These included the augmentation strategy (ranging from basic to advanced), the need for imbalance handling and the appropriate method (e.g., weighted or focal loss, sampling strategies), the evaluation metric, and hyperparameters such as batch size. Upon completion of training, the agent produced and stored a comprehensive set of outputs. These included the best-performing model, the last-epoch model, a confusion matrix, and training plots. Additionally, three JSON files were generated:

- I. Model configuration file containing model type, training parameters (e.g., augmentation level, patience, evaluation metric, number of epochs, best epoch), imbalance strategy, imbalance ratio, and class distribution.
- II. Training history file recording training and validation loss, evaluation metrics, and learning rates across epochs.
- III. Test metrics file reporting evaluation metrics including accuracy, precision, recall andF1-scores, as calculated on the test partition of the model development dataset.

These test metrics served as the baseline for subsequent comparisons with inference performance on unseen data, enabling the detection of potential model degradation. **Figure 3** illustrates an example of the outputs generated for the InceptionV3 model trained on the brain tumor dataset.

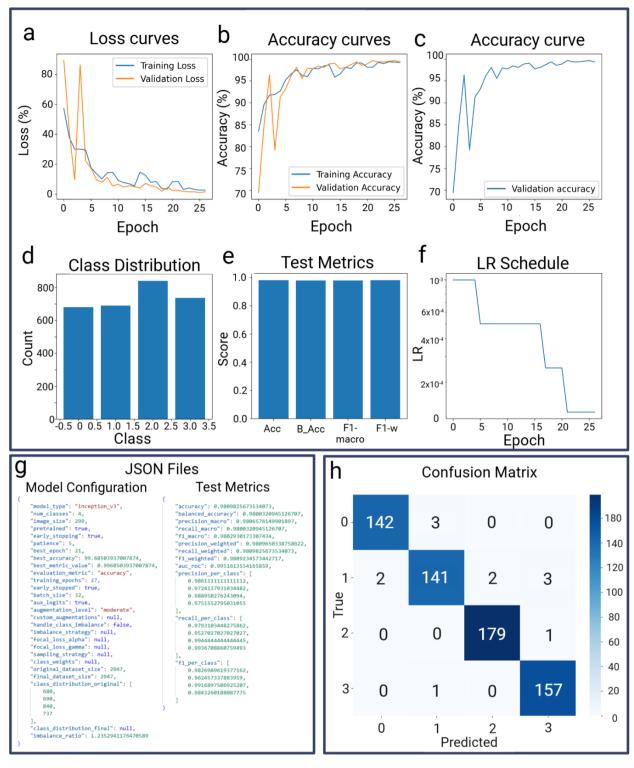


Figure 3. Indicative outputs of ReclAIm for an InceptionV3 model trained on the brain tumor MRI dataset.

3.2 Inference on unseen data

Following model training and internal validation on test partitions, the master agent was prompted to apply the trained models to the inference datasets for each modality. The system executed classification on unseen images and produced a set of outputs. Specifically, it generated a confusion matrix, a JSON file containing the evaluation metrics and a CSV file with detailed per-case results, including filename, case identifier, predicted class, prediction confidence, true label, correctness and the probability distribution across all classes. **Tables 1**, **2** and **3** present the evaluation metrics for test and inference performance of the models trained on the MRI, CT and X-ray datasets, respectively.

3.3 Model degradation detection and fine-tuning

After the independent evaluation procedure was completed, the test and inference metrics were provided to the master agent for each model with prompts to detect potential model degradation. The master agent first invoked the performance comparison agent, which used the corresponding tool to compare the macro and per-class values of accuracy, precision, recall and F1-score between the two evaluation stages. The agent exported plots to visualize the results, including heatmaps showing percentage differences in per-class metrics and bar plots illustrating absolute values and percentage changes in macro-level metrics. In addition, JSON and CSV files containing detailed comparisons and recommendations regarding fine-tuning were generated.

When the comparison revealed a decline greater than 5% in at least one metric (either macro-averaged or per class), the master agent invoked the fine-tuning agent. Using the recommendations from the comparison agent and the fine-tuning dataset, the fine-tuning agent devised an appropriate retraining strategy. After retraining, the updated model was redeployed

to classify the inference data, and its performance was compared with the original baseline to confirm improvement.

Fine-tuning was required in five out of twelve developed models. For the brain tumor dataset (MRI), degradation was observed for EfficientNet (recall for class 0 decreased by 7.6%) and InceptionV3 (with multiple degraded metrics, the largest being recall for class 0, which decreased by 41.1%).

The fine-tuning agent employed two different strategies for these models. For EfficientNet, a full fine-tuning strategy was applied, with no frozen layers, a fine-tuning learning rate of 1×10^{-5} , and a backbone learning rate of 1×10^{-6} . Weighted loss was used to address class imbalance, and catastrophic forgetting was mitigated with a weighting factor of 0.15. For InceptionV3, a partial fine-tuning strategy was applied, freezing the first 150 layers while adapting the higher layers with a fine-tuning learning rate of 2×10^{-5} and a backbone learning rate of 1×10^{-6} . To address imbalance observed during fine-tuning, focal loss (a=0.75, $\gamma=2.0$) was employed. Catastrophic forgetting was penalized with a higher weighting factor of 0.5. For the covid-19 dataset (CT), degradation was also detected for EfficientNet and InceptionV3. For EfficientNet, a full fine-tuning strategy was applied with all layers unfrozen, a fine-tuning learning rate of 1×10^{-5} , and a backbone learning rate of 1×10^{-6} . To better address subtle imbalances, focal loss was introduced with parameters a = 0.25 and $\gamma=2.0$. For InceptionV3, a partial fine-tuning strategy was used, freezing the first 200 layers while adapting the higher layers. The fine-tuning learning rate was set to 1×10^{-4} for the unfrozen layers and 5×10^{-6} for the backbone. Weighted loss was used to handle class balance.

For the pneumonia dataset (X-ray), degradation was detected for the VGG16 model. In this case, a full fine-tuning strategy was employed with all layers unfrozen. Both the fine-tuning and backbone learning rates were set to 1×10^{-5} . No imbalance strategy was applied during retraining.

3.4 Effect of fine-tuning

Per-class percentage differences between test and inference results before and after fine-tuning for the five degraded models can be seen in Figure 4. For MRI EfficientNet, considerable differences before fine-tuning (e.g., recall class 0 = -7.6%, precision class 1 = -5.9%) decreased afterward (recall class 0 = -0.3%; precision class 1 = -2.6%; F1 near 0 across classes). For MRI Inception V3, the largest pre-fine-tuning drops (precision class 2 = -32.9%, recall class 0 = -41.1%, F1 class 0 = -25.5%, F1 class 2 = -19.6%) were largely eliminated, with post-fine-tuning deltas confined to small magnitudes (e.g., precision: [1.1%, 0.0%, 0.6%, -1.4%], recall: [0.9%, -0.1%, -0.7%, 0.4%], F1: [1.0%, -0.1%, 0.0%, -0.5%]). For CT EfficientNet, pre-fine-tuning differences were modest but consistent (precision: [-4.9%, -1.1%], recall: [-0.8%, -5.6%], F1: [-3.0%, -3.4%]) and decreased after fine-tuning (precision: [-1.6%, -0.4%], recall: [-0.4%, -1.7%], F1: [-1.0%, -1.1%]). For CT InceptionV3, the greatest degradation was observed before fine-tuning (precision class 0 = -40.9%, recall class 1 = -40.9%73.0%, F1 class 1 = -57.6%). However, after fine-tuning, all metrics were close to parity (precision: [0.7%, -1.0%], recall: [-1.1%, 0.8%], F1: [-0.2%, -0.1%]). For X-ray VGG16, finetuning reduced but did not fully remove the gap (recall class 0 improved from -7.5% to -5.1%, precision from [1.5%, -2.4%] to [1.0%, -1.6%] and F1 from [-3.2%, -0.8%] to [-2.1%, -0.5%]).

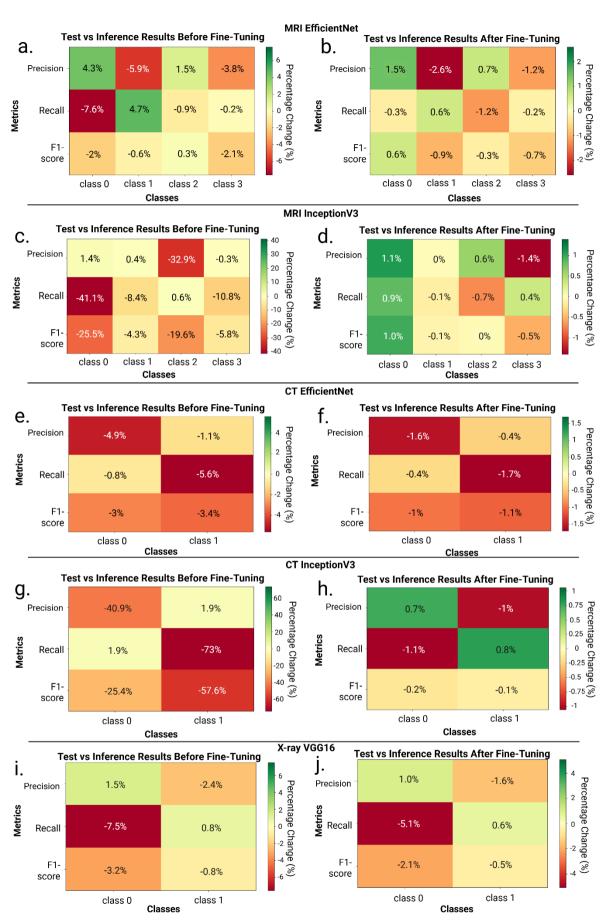


Figure 4. Test and inference results before and after finetuning with ReclAIm for MRI EfficientNet (a,b), MRI InceptionV3 (c,d), CT EfficientNet (e,f), CT InceptionV3 (g,h) and X-ray VGG16 models

Discussion

This manuscript presents ReclAIm, a prototype multi-agent framework capable of detecting performance degradation in medical image classification models. Users can provide natural language prompts to request model evaluation, while the master agent interprets the task, devises an execution strategy, and coordinates the task-specific agents accordingly. Across all tasks, whether model development, degradation detection, or fine-tuning, the system consistently performed as intended without human intervention. A key feature of the proposed framework is its ability to operate in a human-in-the-loop fashion. Owing to its memory capabilities, the system can first be instructed to analyze model performance, then return a detailed degradation assessment along with proposed fine-tuning strategies, and subsequently ask the user how to proceed. When the user provides the next instruction, the system recalls the previous context and results, enabling it to connect the new prompt with prior outputs. Alternatively, it can be prompted from the outset to perform a complete end-to-end process, evaluating the model, detecting degradation, and, if necessary, executing fine-tuning automatically.

Recent literature shows that the application of AI agents in the medical domain is expanding [8, 24–27]. Most existing studies focus on agentic or multi-agent systems designed primarily for text-based tasks, such as generating diagnostic reports or clinical summaries [24–26]. However, the effectiveness of agentic systems in more interactive and autonomous settings, where agents can manipulate their environment, process and transform data, or orchestrate end-to-end AI workflows, remains largely unexplored. Towards this end, multi-agent systems such as mAIstro are capable of performing tasks beyond text generation [8]. mAIstro demonstrated end-to-end functionality for medical imaging, supporting exploratory data analysis, radiomic feature extraction, and the development, evaluation, and deployment of AI models. While their study illustrated the feasibility of applying agentic systems to complex AI workflows, it did

not address the important aspect of continual learning. The present study extends this line of research by demonstrating that agentic systems can also be applied to the autonomous monitoring and maintenance of AI models. As the medical community moves toward the widespread adoption of AI-based software solutions, and as regulatory frameworks emphasize ongoing model oversight [7], such capabilities will become essential for ensuring safety, compliance, and sustained clinical performance. The ReclAIm framework can be used either to probe performance degradation when the model is used with a dataset with different characteristics compared to the training dataset, and can also be used to monitor performance degradation over time. Continual learning enables high algorithmic performance even when new out-of-distribution data or unseen data distributions while avoiding catastrophic forgetting which can lead to a significant drop of performance during retraining.

The FDA has foreseen the need for continuous monitoring of machine learning model performance and the need for adaptive retraining of AI algorithms. For this reason, it has issued guidance for Predetermined Change Control Plans (PCCP) based on Good Machine Learning Practice principles [28]. This guidance supports iterative improvement of AI algorithms in the medical domain as long as it adheres to a predefined set of rules. Towards this end, ReclAIm enables automation of the continual improvement of algorithm performance with a predefined set of tools which can be used in PCCP documentation for a marketing submission of AI-enabled device software functions (AI-DSFs). Even though European regulation lags behind the US in allowing AI algorithm retraining, FDA has set an important precedent in promoting continual automated improvement of AI software.

ReclAIm has certain strengths and limitations. The ability to perform automated performance monitoring communicating with the user only with natural language, as well as the ability to employ state-of-the art methods for continual learning are important strengths of this work. Another strength of the method is that even though ReclAIm is based on LLMs, the use of

strictly defined python tools minimizes the possibility of hallucinations. Nonetheless, in our work ReclAIm has been used to assess performance degradation on unseen datasets and not in temporal data. This would be of value in future work so that performance drifts over time are detected in clinically deployed software.

Conclusion

To the best of our knowledge, this is the first multi-agent system specifically developed for automated monitoring and adaptive maintenance of medical image classification models. Through its integrated tools, the system can preprocess medical imaging data, train and deploy classification models, monitor their performance over time, and apply corrective measures when degradation is detected. Its design, relying solely on natural language interaction, makes its implementation straightforward and accessible, thus supporting broader adoption in both research and clinical environments.

References

- 1. Joshi G, Jain A, Araveeti SR, et al (2024) FDA-approved artificial intelligence and machine learning (AI/ML)-enabled medical devices: An updated landscape. Electronics (Basel) 13:498. https://doi.org/10.3390/electronics13030498
- 2. Center for Devices, Radiological Health (2025) Artificial Intelligence-Enabled Medical Devices. In: U.S. Food and Drug Administration. https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-enabled-medical-devices. Accessed 16 Oct 2025
- 3. van Leeuwen KG, Schalekamp S, Rutten MJCM, et al (2021) Artificial intelligence in radiology: 100 commercially available products and their scientific evidence. Eur Radiol 31:3797–3804. https://doi.org/10.1007/s00330-021-07892-z
- 4. Roschewitz M, Mehta R, Jones C, Glocker B (2025) Automatic dataset shift identification to support safe deployment of medical imaging AI. arXiv:2411.07940
- 5. Guan H, Bates D, Zhou L (2025) Keeping medical AI healthy: A review of detection and correction methods for system degradation. arXiv:2506.17442
- 6. Lacson R, Eskian M, Licaros A, et al (2022) Machine learning model drift: Predicting diagnostic imaging follow-up as a case example. J Am Coll Radiol 19:1162–1169. https://doi.org/10.1016/j.jacr.2022.05.030
- 7. EU Artificial Intelligence Act, Article 72: Post-Market Monitoring by Providers and Post-Market Monitoring Plan for High-Risk AI Systems. https://artificialintelligenceact.eu/article/72/. Accessed 17 Oct 2025
- 8. Tzanis E, Klontzas ME (2025) mAIstro: An open-source multi-agent system for automated end-to-end development of radiomics and deep learning models for medical imaging. European Journal of Radiology Artificial Intelligence 4:100044. https://doi.org/10.1016/j.ejrai.2025.100044
- 9. Roucher A, del Moral AV, Wolf T, von Werra L, Kaunismäki E (2025) smolagents: a barebones library for agents that think in code. https://github.com/huggingface/smolagents, Accessed 19 Oct 2025
- 10. He K, Zhang X, Ren S, Sun J (2015) Deep residual learning for image recognition. arXiv:1512.03385
- 11. Szegedy C, Vanhoucke V, Ioffe S, et al (2015) Rethinking the inception architecture for computer vision. arXiv:1512.00567
- 12. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556
- 13. Tan M, Le QV (2019) EfficientNet: Rethinking model scaling for convolutional Neural Networks. arXiv:1905.11946

- 14. Davila A, Colan J, Hasegawa Y (2024) Comparison of fine-tuning strategies for transfer learning in medical image classification. Image Vis Comput abs/2406.10050.: https://doi.org/10.1016/j.imavis.2024.105012
- 15. Parthasarathy VB, Zafar A, Khan A,Shahid A (2024) The ultimate guide to fine-tuning LLMs from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. arXiv:2408.13296
- 16. Perkonigg M, Hofmanninger J, Herold CJ, et al (2021) Dynamic memory to alleviate catastrophic forgetting in continual learning with medical imaging. Nat Commun 12:5678. https://doi.org/10.1038/s41467-021-25858-z
- 17. Kirkpatrick J, Pascanu R, Rabinowitz N, et al (2017) Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci U S A 114:3521–3526. https://doi.org/10.1073/pnas.1611835114
- 18. van de Ven GM, Soures N, Kudithipudi D (2024) Continual learning and catastrophic forgetting. ArXiv abs/2403.05175.: https://doi.org/10.1016/B978-0-443-15754-7.00073-0
- 19. Yao S, Zhao J, Yu D, et al (2022) ReAct: Synergizing reasoning and acting in language models. arXiv:2210.03629
- 20. Nickparvar M (2021) Brain Tumor MRI Dataset. Kaggle doi: 10.34740/KAGGLE/DSV/2645886
- 21. Soares E, Angelov P (2020) SARS-COV-2 Ct-Scan Dataset. Kaggle, doi:10.34740/KAGGLE/DSV/1199870
- 22. Yang, J., Shi, R., Wei, D. et al. (2023) MedMNIST v2 A large-scale lightweight benchmark for 2D and 3D biomedical image classification. Sci Data 10, 41
- 23. OpenTelemetry. In: OpenTelemetry. https://opentelemetry.io/. Accessed 17 Oct 2025
- 24. Chen X, Yi H, You M, et al (2025) Enhancing diagnostic capability with multi-agents conversational large language models. NPJ Digit Med 8:159. https://doi.org/10.1038/s41746-025-01550-0
- 25. Mehandru N, Miao BY, Almaraz ER, et al (2024) Evaluating large language models as agents in the clinic. NPJ Digit Med 7:84. https://doi.org/10.1038/s41746-024-01083-y
- 26. Tang X, Zou A, Zhang Z, et al (2024) MedAgents: Large language models as collaborators for zero-shot medical reasoning. In: Ku L-W, Martins A, Srikumar V (eds) Findings of the Association for Computational Linguistics ACL 2024. Association for Computational Linguistics, Stroudsburg, PA, USA, pp 599–621
- 27. Zephyr M (2025) MONAI Integrates Advanced Agentic Architectures to Establish Multimodal Medical AI Ecosystem. In: NVIDIA Technical Blog. https://developer.nvidia.com/blog/monai-integrates-advanced-agentic-architectures-to-establish-multimodal-medical-ai-ecosystem/. Accessed 17 Oct 2025

28. Center for Devices, Radiological Health (2025) Marketing Submission Recommendations for a Predetermined Change Control Plan for Artificial Intelligence-Enabled Device Software Functions. In: U.S. Food and Drug Administration. https://www.fda.gov/regulatory-information/search-fda-guidance-documents/marketing-submission-recommendations-predetermined-change-control-plan-artificial-intelligence. Accessed 19 Oct 2025

Table 1. Test and inference evaluation for the Brain Tumor MRI dataset

Model	Evaluation Dataset	Accuracy	Precision				Recall					F1					
			Macro	Class 0	Class 1	Class 2	Class 3	Macro	Class 0	Class 1	Class 2	Class 3	Macro	Class 0	Class 1	Class 2	Class 3
EfficientNet	Test set	0.96	0.96	0.95	0.97	0.98	0.95	0.96	0.97	0.89	0.99	0.99	0.96	0.96	0.93	0.99	0.97
	Inference set	0.95	0.95	0.99	0.91	0.99	0.92	0.95	0.89	0.93	0.99	0.99	0.95	0.94	0.92	0.99	0.95
Fine-tuned	Inference set	0.96	0.96	0.96	0.95	0.98	0.94	0.96	0.96	0.89	0.98	0.99	0.96	0.96	0.92	0.98	0.97
EfficientNet																	
Incontion V2	Test set	0.98	0.98	0.99	0.97	0.99	0.98	0.98	0.98	0.95	0.99	0.99	0.98	0.98	0.96	0.99	0.98
InceptionV3	Inference set	0.84	0.90	1.00	0.98	0.66	0.97	0.83	0.58	0.87	1.0	0.89	0.84	0.73	0.92	0.80	0.93
Fine-tuned	Inference set	0.98	0.98	1.00	0.97	0.99	0.96	0.98	0.99	0.95	0.99	1.00	0.98	0.99	0.96	0.99	0.98
InceptionV3																	
ResNet50	Test set	0.98	0.98	0.97	0.99	0.99	0.97	0.98	0.99	0.95	0.98	0.99	0.98	0.98	0.97	0.98	0.98
Resiletsu	Inference set	0.98	0.98	0.98	0.96	0.99	0.97	0.98	0.98	0.96	0.98	0.99	0.98	0.98	0.96	0.98	0.98
VGG16	Test set	0.98	0.98	0.99	0.97	0.99	0.99	0.98	0.98	0.96	1.00	0.99	0.98	0.98	0.97	0.99	0.99
	Inference set	0.98	0.98	0.99	0.96	1.00	0.98	0.98	0.97	0.97	1.00	0.99	0.98	0.98	0.97	1.00	0.99

Table 2. Test and inference evaluation for the COVID-19 CT dataset

Model	Evaluation Dataset	Accuracy	Precision				Recall		F1		
			Macro	Class 0	Class 1	Macro	Class 0	Class 1	Macro	Class 0	Class 1
EfficientNet	Test set	0.94	0.94	0.92	0.95	0.94	0.95	0.92	0.94	0.94	0.94
	Inference set	0.91	0.91	0.88	0.94	0.91	0.95	0.87	0.91	0.91	0.90
Fine-tuned EfficientNet	Inference set	0.93	0.93	0.91	0.95	0.93	0.95	0.90	0.93	0.93	0.93
InceptionV3	Test set	0.97	0.97	0.96	0.98	0.97	0.98	0.96	0.97	0.97	0.97
	Inference set	0.63	0.78	0.57	1.00	0.63	1.00	0.26	0.57	0.73	0.41
Fine-tuned InceptionV3	Inference set	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
ResNet50	Test set	0.94	0.94	0.94	0.95	0.94	0.95	0.94	0.94	0.94	0.94
	Inference set	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96	0.96
VGG16	Test set	0.99	0.99	0.97	1.00	0.99	1.00	0.97	0.99	0.99	0.99
	Inference set	0.98	0.98	0.97	1.00	0.98	1.00	0.97	0.98	0.98	0.98

 Table 3. Test and inference evaluation for the Pneumonia X-ray dataset

Model	Evaluation Dataset	Accuracy		Precision			Recall		F1		
			Macro	Class 0	Class 1	Macro	Class 0	Class 1	Macro	Class 0	Class 1
EfficientNet	Test set	0.94	0.94	0.93	0.95	0.92	0.86	0.97	0.93	0.89	0.96
	Inference set	0.93	0.92	0.91	0.94	0.90	0.82	0.97	0.91	0.86	0.95
InceptionV3	Test set	0.94	0.92	0.87	0.97	0.94	0.92	0.95	0.93	0.90	0.96
	Inference set	0.95	0.93	0.91	0.96	0.93	0.89	0.97	0.93	0.90	0.96
ResNet50	Test set	0.93	0.91	0.87	0.96	0.92	0.89	0.95	0.92	0.88	0.95
	Inference set	0.92	0.91	0.87	0.95	0.90	0.85	0.95	0.90	0.86	0.95
VGG16	Test set	0.95	0.94	0.91	0.96	0.94	0.91	0.96	0.94	0.91	0.96
	Inference set	0.94	0.93	0.92	0.94	0.91	0.84	0.97	0.92	0.88	0.96
Fine-tuned VGG16	Inference set	0.94	0.93	0.92	0.95	0.92	0.86	0.97	0.92	0.89	0.96