algorithms

MDPI

Article

A Big Data Pipeline Approach for Predicting Real-Time Pandemic Hospitalization Risk

Vishnu S. Pendyala *, Mayank Kapadia, Basanth Periyapatnaroopakumar, Manav Anandani and Nischitha Nagendran

Department Applied Data Science, College of Information, Data, and Society, San Jose State University, San Jose, CA 95192, USA; mayank.kapadia@sjsu.edu (M.K.); basanth.periyapatnaroopakumar@sjsu.edu (B.P.); manavrajesh.anandani@sjsu.edu (M.A.); nischitha.nagendran@sjsu.edu (N.N.)
* Correspondence: vishnu.pendyala@sjsu.edu

Abstract

Pandemics emphasize the importance of real-time, interpretable clinical decision-support systems for identifying high-risk patients and assisting with prompt triage, particularly in data-intensive healthcare systems. This paper describes a novel dual big-data pipeline that includes (i) a streaming module for real-time epidemiological hospitalization risk prediction and (ii) a supplementary imaging-based detection and reasoning module for chest X-rays, with COVID-19 as an example. The first pipeline uses state-of-the-art machine learning algorithms to estimate patient-level hospitalization risk based on data from the Centers for Disease Control and Prevention's (CDC) COVID-19 Case Surveillance dataset. A Bloom filter accelerated triage by constant-time pre-screening of high-risk profiles. Specifically, after significant experimentation and optimization, one of the models, XGBoost, was selected because it achieved the best minority-class F1-score (0.76) and recall (0.80), outperforming baseline models. Synthetic data generation was employed to mimic streaming workloads, including a strategy that used the Conditional Tabular Generative Adversarial Network (CTGAN) to produce the best balanced and realistic distributions. The second pipeline focuses on diagnostic imaging and combines an advanced convolutional neural network, EfficientNet-B0, with Grad-CAM visual explanations, achieving 99.5% internal and 99.3% external accuracy. A lightweight Generative Pre-trained Transformer (GPT)-based reasoning layer converts model predictions into auditable triage comments (ALERT/FLAG/LOG), yielding traceable and interpretable decision logs. This scalable, explainable, and near-real-time framework provides a foundation for future multimodal and genomic advancements in public health readiness.

Keywords: hospitalization risk prediction; real-time streaming analytics; chest X-ray classification; explainable AI (XAI); bloom filter; Conditional Tabular Generative Adversarial Network (CTGAN); Kafka–Spark streaming; clinical decision support

Academic Editor: Frank Werner

Received: 28 September 2025
Revised: 7 November 2025
Accepted: 14 November 2025
Published: 21 November 2025

Citation: Pendyala, V.S.; Kapadia, M.; Periyapatnaroopakumar, B.; Anandani, M.; Nagendran, N. A Big Data Pipeline Approach for Predicting Real-Time Pandemic Hospitalization Risk. *Algorithms* **2025**, *18*, 730. https://doi.org/10.3390/a18120730

1. Introduction

Early in 2020, the COVID-19 epidemic quickly spread across continents in a few short weeks, causing a global sense of unease and anxiety [1]. Within a short period, a microscopic virus was able to stop the speed of nations, upending society and having a significant negative effect on the world economy. This crisis primarily affected healthcare systems, which saw sharp increases in hospitalizations, intensive care unit admissions, and mortality. Critical shortages of medical resources [2], such as hospital beds, ventilators,

and oxygen supply, resulted from the abrupt demand. This circumstance exposed a basic problem: the discrepancy between the quick increase in patient danger and the clinical decision-making systems' later reactions. Clinical prioritization, bed management, and efficient triage all depend on timely and accurate hospitalization risk prediction.

1.1. Motivation

The essential need for real-time tools that can assist clinical decision-making in uncertain and resource-constrained situations was brought to light by the COVID-19 pandemic. Effective triage, in particular, calls for tools that strike a balance between speed, dependability, and interpretability. There is a need to screen patients quickly and in large quantities, operate reliably in streaming contexts, and clearly explain each decision [3].

We suggest a two-pronged real-time approach to meet these needs. The first pipeline predicts hospitalization risk at the patient level by utilizing extensive epidemiological monitoring data. In order to validate diagnoses and identify early warning signs, the second pipeline classifies chest X-ray pictures, concentrating on radiological evidence. These complementary pipelines work together to improve readiness for upcoming pandemics, appropriately escalate high-risk cases, and produce outputs that can be clinically interpreted. Our platform seeks to provide dependable and auditable triage support by combining streaming infrastructure, powerful machine learning models, synthetic data generation, and a lightweight LLM-generated decision log that uses model outputs and Grad-CAM [4] summaries to produce auditable triage notes (ALERT/FLAG/LOG) [5].

1.2. Research Questions

This study is guided by the following research questions:

1. Real-time viability: Is it feasible to create and execute a pipeline in almost real-time for the classification of chest X-rays and the prediction of epidemiological hospitalization risk while preserving accuracy and interpretability?
2. Latency enhancement: How much prediction delay in the hospitalization risk pipeline can be effectively reduced by Bloom filter encoding without sacrificing recall for high-risk cases?
3. LLM integration: Without altering the model's outputs, how successfully can an LLM convert forecasts and Grad-CAM into succinct, auditable comments that support clinician interpretability?

1.3. Contributions

The main contributions of this work are as follows:

1. With XGBoost [6] as the central model for hospitalization risk assessment, we use the CDC COVID-19 Case Surveillance dataset [7] to create a real-time epidemiological risk prediction pipeline.
2. To produce auditable triage outputs, we develop a supplementary chest X-ray classification pipeline, refine it with EfficientNet-B0 [8], and include Grad-CAM visual explanations and a lightweight GPT-based reasoning layer.
3. Since time is critical in controlling the spread of pandemics, we add Bloom filter [9] encoding to the epidemiological pipeline to boost efficiency, allowing for quick prescreening of low-risk cases while maintaining recall for high-risk patients.
4. To test the streaming pipeline and evaluate the trade-offs between realism and robustness, we methodically examine three synthetic data generation strategies: random sampling [10], distribution-aware sampling [11], and CTGAN [12].

5. We show how structured outputs from the classification process, when combined with explainability approaches (Grad-CAM), improve clinical interpretability and offer useful information for decision support.

The proposed method integrates two complementary real-time operations into a single pandemic-triage architecture. The first epidemiological pipeline, which uses machine learning models and Bloom filter pre-screening, enables large-scale screening and prioritizing by swiftly estimating hospitalization risk from tabular case-surveillance data. The second imaging-based pipeline uses chest X-rays to validate diagnoses. It includes a GPT-based reasoning layer that creates auditable triage summaries (ALERT, FLAG, and LOG), Grad-CAM for visual explanation, and EfficientNet-B0 for classification. This strategy is comparable to real-world methods such as population-level risk classification, interpretable decision support, and imaging-based validation.

2. Related Work

The nexus of real-time streaming analytics and machine learning has become a lively research topic, especially for hospitalization risk prediction and pandemic preparedness. Previous research has usually concentrated on either radiological detection utilizing chest X-rays or epidemiological risk modeling with tabular data, but infrequently, both have been studied in the same real-time environment. The creation of scalable and interpretable clinical decision-support systems is informed by these two complementary lines of research and the methods that enable them (such as explainable AI, Bloom filters, and synthetic data generation), which are expanded upon in our work. Before describing the remaining gap, we evaluate relevant studies in these areas below.

2.1. Tabular Risk Prediction and Real-Time Streaming

Early COVID-19 hospitalization and severity prediction studies focused primarily on clinical and demographic data such as age, gender, comorbidities, and lab results [13]. Classical and ensemble techniques for admission or severity prediction, such as random forests, support vector machines, and gradient boosting models, have yielded AUCs ranging from 0.66 to 0.87. Studies employing clinical characteristics and demographics have demonstrated varying performance levels [14]. Hwangbo et al. showed that random forests and XGBoost performed well in multicenter validations [15]. However, these methodologies were not integrated into bigger decision-making pipelines, were typically taught offline, and lacked real-time streaming capabilities.

Hospital operations investigate concurrently predicted aggregate admissions using time-series and ensemble models to guide bed allocation and staffing decisions. These models were valuable for population-level planning, but they did not provide patient-specific explanations or customized risk classification.

Distributed streaming frameworks like Apache Kafka and Spark Streaming are being used more and more in real-time healthcare analytics to overcome latency and scalability issues [16,17]. Cloud-based streaming platforms for chronic disease monitoring have shown promising results [18]. Recent developments in data streaming for healthcare applications continue to evolve [19]. Predictive monitoring for chronic diseases has been demonstrated by proof-of-concept systems, with models updated in real time as new data becomes available. However, these systems frequently only partially integrate into clinical operations and remain technical prototypes.

Additionally, efforts to create synthetic data have improved privacy protection and robustness testing [20]. When it comes to maintaining inter-feature dependencies, Conditional Tabular GANs (CTGANs) perform better than random or copula-based approaches [21]; yet, there are still few applications in streaming pipeline validation. Bloom filters have

also been investigated for secure cohort finding and privacy-preserving record linkage [22], with recent work exploring clinical trial cohort discovery applications, but they are still in the early stages of adapting to quick pre-screening in clinical triage.

2.2. Imaging-Based COVID-19 Detection and Explainability

Simultaneously, chest X-rays and deep learning techniques have demonstrated impressive COVID-19 detection capabilities [23]. Accuracy levels above 95% are regularly reported by architectures like EfficientNet-B0, ResNet variations, and MobileNetV2, with EfficientNet-B0 demonstrating strong generalization across datasets [24]. Comprehensive validation studies have verified specificities above 99% and sensitivities above 93%. Recent advances in COVID-19 severity detection using segmentation approaches show continued progress [25].

Explainability is a crucial prerequisite for clinical deployment. Clinicians can determine if models concentrate on pertinent pulmonary features by using Gradient-weighted Class Activation Mapping (Grad-CAM), which has been routinely used to reveal radiologically significant regions impacting predictions [26]. Research shows that effective visual explanations can increase diagnostic reliability, interpretability, and confidence [27]. Recent studies focus on Grad-CAM-based explainable AI applications in medical diagnosis [28]. Despite these developments, the majority of chest X-ray models can only be evaluated offline and have little incorporation into real-time or streaming systems.

2.3. Research Gap

When combined, previous studies have improved explainability in chest X-ray categorization, streaming infrastructures, synthetic data generation, Bloom filters, and tabular epidemiological models. These strands are still separated, though. Imaging-based detection and tabular risk stratification are rarely investigated in the same real-time framework, and the implementations that are currently in use are either proof-of-concept or offline.

By creating and assessing two complementary real-time pipelines, we fill this gap: (i) a pipeline for tabular epidemiological risk prediction that uses XGBoost, Bloom filter pre-screening, and CTGAN-based stress testing; and (ii) a pipeline for classifying chest X-rays that uses EfficientNet-B0 and Grad-CAM for explainability. We establish a proof-of-concept for scalable, interpretable, and effective decision support by putting these pipelines into practice in a Kafka-Spark context, setting the stage for future integration in pandemic triage systems.

3. Multimodal Data Description

3.1. Tabular Data

Our multimodal pipeline's tabular component is taken from the COVID-19 Case Surveillance Public Use dataset from the Centers for Disease Control and Prevention (CDC) [29]. Recent advances in multimodal AI have demonstrated significant improvements in diagnostic accuracy when multiple healthcare data sources are systematically combined [30,31]. Research exploring the integration of multimodal real-world data in healthcare shows promising results for enhanced diagnostic capabilities [32]. We ensured reproducibility and consistent schema selection by retrieving the dataset directly through the CDC Socrata API (https://data.cdc.gov/resource/vbim-akqf.json, accessed on 20 July 2025), as opposed to static CSV dumps. Using the `$limit` and `$offset` parameters from the API, a custom Python script was created that downloaded the dataset in batches of 10,000 rows per request. A brief lag between queries was included to avoid rate-limiting problems, and the maximum number of records that could be retrieved was 4,000,000. The following fields were chosen for analysis:

- *sex*, *age_group*, and *race_ethnicity_combined* are the demographic attributes.
- *hosp_yn* (hospitalization), *icu_yn* (ICU admission), *death_yn* (mortality), and *medcond_yn* (comorbidity status) are examples of clinical attributes.
- *cdc_case_earliest_dt*, *current_status* are Case metadata.

For preprocessing and modeling later on, the combined JSON replies were saved as a CSV file. More than three million patient-level case records from throughout the US were made available by this collection technique, allowing for data fidelity and population-scale hospitalization risk prediction.

Class Distribution and Data Preprocessing

The raw data were preprocessed to remove missing values, standardize numerical fields as needed, and encode categorical characteristics (sex, age group, race/ethnicity, and comorbidity status). One of the most serious issues discovered was a significant class disparity between hospitalized and non-hospitalized cases. Figure 1 shows that the dataset has a considerable bias against non-hospitalized individuals.

We utilized random undersampling of the majority (non-hospitalized) class to correct this imbalance. This strategy was specifically chosen over oversampling to preserve the integrity of categorical feature distributions and avoid introducing duplicate or fraudulent data that could influence model learning. Figure 2 shows that the undersampling approach resulted in a about 1:1 ratio of non-hospitalized cases to hospitalized ones.

As mentioned in Section 4.1.1, the balanced dataset generated by this technique was subsequently used for model training and evaluation.

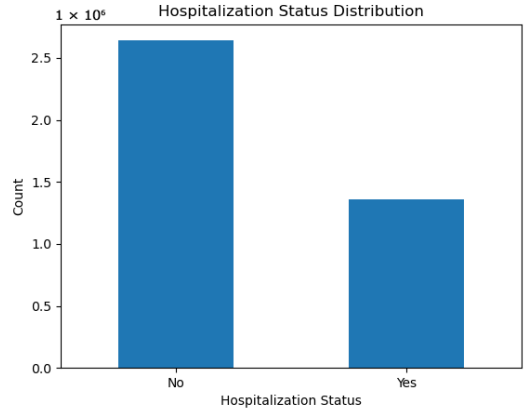


Figure 1. Distribution of hospitalization status in the CDC dataset. Non-hospitalized cases dominate, motivating the use of imbalance-handling strategies.

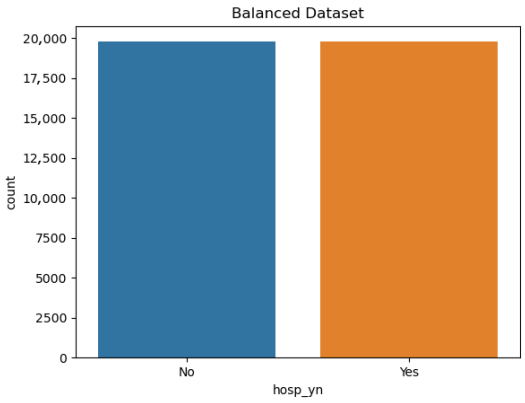


Figure 2. Balanced dataset obtained after applying undersampling. The number of non-hospitalized (0) and hospitalized (1) cases was adjusted to achieve a 1:1 ratio.

3.2. Chest X-Ray Images

We selected chest X-ray datasets from openly accessible archives that have been approved for use in clinical studies. The TCIA COVID-19 Imaging Archive [33], which included DICOM-formatted medical scans, served as the main source. There were only 256 chest X-rays that tested positive for COVID after filtering and conversion. We added the COVID-19 Radiography Database (Kaggle) [34], which included 3611 COVID-positive and 10,192 Normal pictures, to this. Following the merger, 3867 COVID-positive and 10,192 normal pictures were included in the combined dataset.

3.2.1. Preprocessing and Augmentation

Before being used in the classification pipeline, the raw datasets needed to be preprocessed. The TCIA archive's DICOM files were first converted to PNG format and filtered to preserve chest X-ray studies. By resizing, normalizing the grayscale, and storing them in a defined folder structure, all photos were made uniform.

The significant class imbalance—normal photos outnumbering COVID-positive instances by over 3:1—was a serious problem. When we first investigated synthetic generation using DCGAN [35], we discovered that the generated samples lacked characteristic COVID pathologies and that baseline models misclassified them. As a result, we went with a more reliable strategy based on conventional data augmentation [36]. We increased the dataset while maintaining clinical validity by applying changes such as rotations, brightness/contrast tweaks, horizontal flips, and affine transformations.

The vision models were trained using the balanced dataset of 3000 normal and 3000 COVID-positive photos that was produced by the augmentation method.

3.2.2. Dataset for External Testing

We tested the trained model using an independent chest X-ray dataset to determine its reliability and generalizability. The split for the dataset is shown in Table 1. The 317 Normal and 116 COVID-positive pictures in this external set were never viewed during training or augmentation. Section 5 (Experiments and Results) reports the findings from this external dataset.

Table 1. Chest X-ray dataset composition before and after augmentation, and in the external test set.

Category	After Merge (TCIA + Kaggle)	After Augmentation (Train)	External Test Set
Normal Images	10,192	3000	317
COVID Images	3867	3000	116
Total	14,059	6000	433

4. Methodology

The approach used for two separate data modalities is explained in this section. The end-to-end pipeline for tabular data is shown in Section 1, with a focus on hospitalization risk estimation. The pipeline for classifying chest X-rays is described in Section 2. A lightweight LLM decision layer is added to facilitate the final decision-making process. Figure 3 depicts the whole architecture of the proposed dual pipeline. For diagnostic validation, imaging data is routed through the EfficientNet-B0 + Grad-CAM + GPT reasoning module, while tabular data is processed by XGBoost with Bloom-filter pre-screening for hospitalization risk prediction. Both data streams begin with ingestion and preprocessing. The streaming layer records outputs, as well as time and intelligible labels. This graphic summarizes the complete process, bringing data intake, model inference, and decision tracking together into a coherent framework.

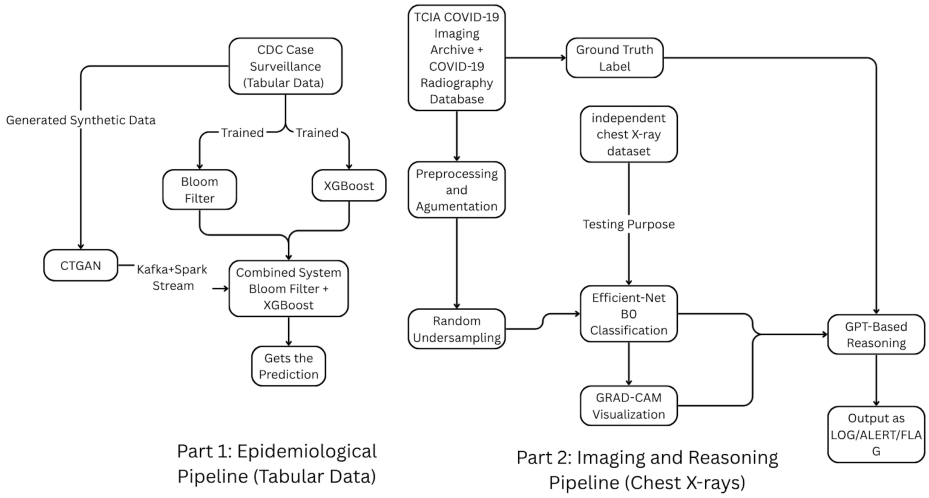


Figure 3. Overall architecture of the dual big-data pipeline integrating (i) an epidemiological hospitalization-risk prediction module using CDC case-surveillance data and (ii) an imaging-based diagnostic reasoning module for chest X-rays. Both operate within a Kafka–Spark streaming framework, with CTGAN providing synthetic tabular data and a unified decision-logging layer consolidating predictions and auditable reasoning outputs.

4.1. Part 1: Tabular Hospitalization Risk Prediction Pipeline

The methodology used for the CDC dataset (as presented in the Dataset section) is shown in Figure 4. Through comparative analysis, we assess several conventional machine learning models for hospitalization risk prediction and determine which is the most successful. We use Grid Search [37] for hyperparameter optimization in order to maximize model performance. To enhance data processing efficiency, we also introduce Bloom filter encoding and employ suitable strategies to address the issue of class imbalance. We create synthetic data and link the pipeline with Spark and Apache Kafka to enable real-time deployment [38]. The following is a detailed discussion of each of these elements.

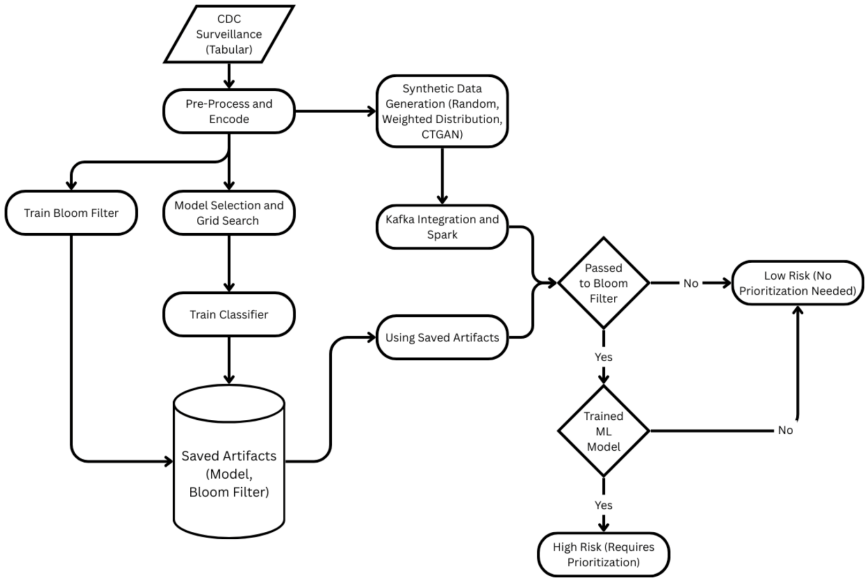


Figure 4. Integrated surveillance data-based end-to-end pipeline for hospitalization risk at the patient level. Preprocessing/encoding, Grid Search model selection, XGBoost training, building a Bloom filter on high-risk profiles, and CTGAN-based synthetic generation are all included in the training path. To classify instances as Low-Risk or High-Risk (requiring priority), Kafka-Spark first ingests and encodes records, then conducts a Bloom-filter membership check and, if detected, scores using the learned model.

4.1.1. XGBoost Classifier with Grid Search

We tested three machine-learning models to predict hospitalization risk in tabular data: Extreme Gradient boosting (XGBoost), Random Forest [39], and Logistic Regression [40]. Random Forest provides nonlinear decision boundaries and tolerance to noise, while Logistic Regression provides a straightforward linear baseline. The balanced dataset described in Section 3 was used as the training set for all models.

XGBoost was chosen as the main classifier for further optimization due to its ability to handle heterogeneous feature types, capture nonlinear interactions, and successfully control class imbalance qualities, all of which make it particularly ideal for large-scale epidemiological data. XGBoost outperformed other gradient-boosting algorithms (LightGBM, CatBoost) in terms of performance and interpretability.

We used GridSearchCV with threefold cross-validation to maximize model performance. The computational efficiency and reproducibility of this deterministic search approach in a confined hyperparameter space influenced its adoption. An additional comparison using fivefold cross-validation confirmed that threefold CV provided sufficient stability for our dataset, yielding an almost identical mean F1-score (0.7609 vs. 0.7608). The adjustable parameter grids include the learning rate, number of estimators, subsampling ratio, maximum tree depth, and column sample per tree. A detailed comparison of the models is presented in Section 5, and the final configuration of the model was chosen based on cross-validation results.

4.1.2. Bloom Filter Encoding

Bloom filter encoding was introduced as a probabilistic pre-screening approach [9] to expedite risk detection in our classification pipeline, which was novel. We trained the Bloom filter on high-risk rows from the dataset, utilizing prior knowledge of hospitalization characteristics. Bloom filters support $O(1)$ membership searches with predictable false-positive rates, making them ideal for quick triage in high-throughput environments. We used a Bloom filter structure to encode high-risk patient profiles identified during training. During inference, each incoming record was initially checked against the filter. If the record was marked as possibly high-risk, it was forwarded to the more expensive XGBoost model for thorough evaluation. This design lowered total latency by filtering out low-risk cases early on while retaining recall on high-risk patients, boosting the speed and efficiency of the hospitalization risk prediction pipeline.

4.1.3. Synthetic Data Generation for Streaming Simulation

In order to simulate near-real-time situations and assess the pipeline's resilience, synthetic data streams were created after the machine learning classifier and Bloom filter were trained. As a result, the Kafka–Spark combination could be stress-tested under constant input flow. Three different approaches were used:

- Random Generator: Feature values were separately sampled using the `random` module that comes with Python. Despite being computationally straightforward, our method was unable to identify feature dependencies that existed in the actual dataset.
- Weighted Distribution Sampling: Here, the empirical feature distributions in the CDC dataset were mimicked to construct synthetic samples. This approach yielded more realistic samples that more accurately represented observed statistical patterns than random creation.
- Conditional Tabular GAN (CTGAN): We used CTGAN, a cutting-edge generative model for tabular data, to generate higher-fidelity synthetic data. In contrast to the more straightforward techniques, CTGAN produced realistic samples appropriate for streaming simulation by maintaining both class proportions and inter-feature

relationships. To assess the quality of the synthetic data and convergence, models were trained across 300, 500, and 1000 epochs.

Section 5 (Experiments and Results) provides a comparative analysis of different methods, delving deeply into their efficacy in real-time simulation.

4.1.4. Kafka-Spark Streaming Integration

The creation of an end-to-end pipeline with near real-time operation was made possible by the integration of Apache Spark and Apache Kafka version 3.9.0 (Scala 2.12 build). For the continuous intake of patient records, comprising both real and generated data streams, Kafka was used as the message broker. After parsing the incoming data, Spark Streaming deployed user-defined functions (UDFs) that successively called the XGBoost classifier and the trained Bloom filter. While the XGBoost model offered more computationally demanding risk score for situations identified as potentially high-risk, the Bloom filter functioned as a quick pre-screening method. The results were then saved for further examination after being formatted into structured JSON data. This streaming approach is shown schematically in Figure 4.

4.2. Part 2: Chest X-Ray Classification and LLM Integration

This section discusses how deep learning-based vision models are used to classify chest X-ray images, as described in the data section above. In addition to the base classifier, we also utilize explainable AI techniques to provide additional interpretability, with Grad-CAM being employed to visualize regions influencing each prediction. Furthermore, we introduce a decision-making layer powered by large language models (LLMs) [41], i.e., GPT-3.5 or GPT-4o, which assesses model predictions along with Grad-CAM outputs. Recent work in LLM-based clinical decision support has shown promising results. Through prompt engineering, we elicit clinically-aligned reasoning from the LLM, enabling the system to agentively decide to `ALERT`, `FLAG`, or `LOG` each case. Multi-agent frameworks have demonstrated effectiveness in emergency department triage scenarios. This introduction of LLM [42] is intended to further the trustworthiness and human-alignment of the overall system.

4.2.1. Model Selection and Final Classifier

To find the optimal deep learning model architecture for chest X-ray classification, we have compared several convolutional neural networks (CNNs), like a custom baseline CNN [43], MobileNetV2 [44], ResNet-18 [45], and EfficientNet-B0. The models have been trained using a standard pipeline: input images resized to 224×224, normalized, and augmented with random horizontal flip, rotation, brightness/contrast jitter, and affine transforms to allow for improved generalization.

Training was conducted over various epochs (20, 30, and 40) on every architecture to assess performance convergence. Hyperparameters were set to batch size (32) and optimizer (Adam) [46] in all experiments. The performance of every model was assessed in accuracy, precision, recall, and F1-score on a test set held out.

Of all the candidates, EfficientNet-B0 had the most consistent and stable performance across different training runs, doing best in COVID-19 sensitivity and overall F1-score [47]. EfficientNet-B0 was therefore selected as the final classifier for further downstream explainability and LLM-generated decision logs.

The fine-tuning setup is provided in Figure 5, where the pre-trained EfficientNet-B0 model (trained originally on ImageNet [48]) was adapted for binary classification by replacing its final fully connected layer. The larger-scale quantitative comparison of all architectures that were evaluated is presented in Section 5.

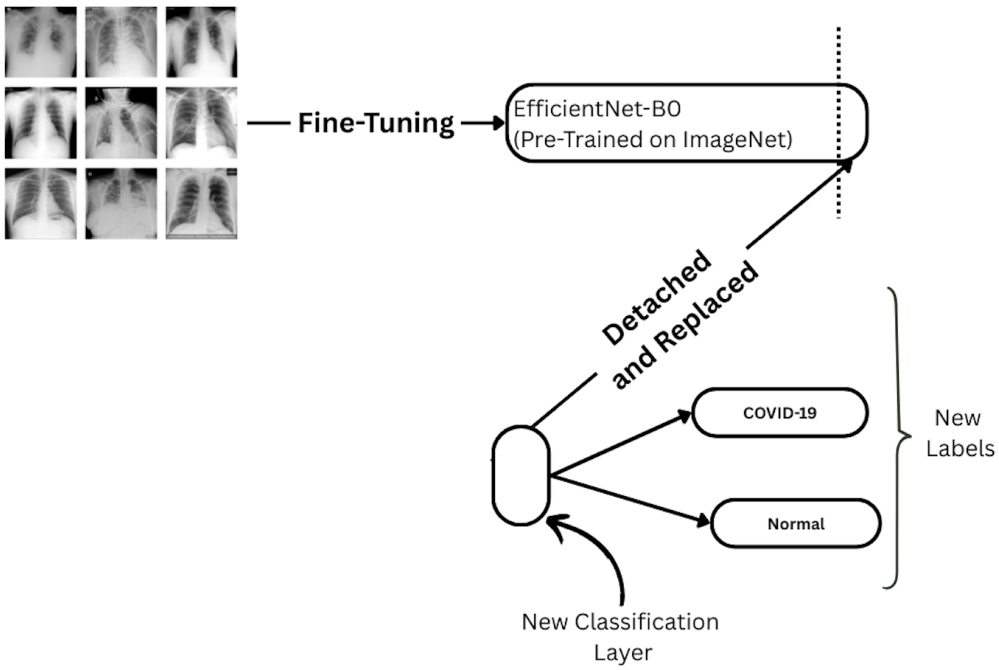


Figure 5. Fine-tuning setup for EfficientNet-B0. The final classification head is detached and replaced with a new layer for binary prediction (COVID-19 vs. Normal).

4.2.2. Grad-CAM for Visual Explanation

To increase model decision interpretability, we used Gradient-weighted Class Activation Mapping (Grad-CAM) on the trained EfficientNet-B0 model. Grad-CAM is a post-hoc visual explanation that highlights the most significant areas in the input image responsible for the model prediction. This is especially helpful in radiologic applications, where model transparency can contribute to radiologists trust and decision-making.

As illustrated in Figure 6, the Grad-CAM process begins by finding the last convolutional layer of the trained model. Gradients of the predicted class are utilized to weight the feature maps of the same layer. The result is a heatmap of the salient regions, which is overlaid on the original chest X-ray.

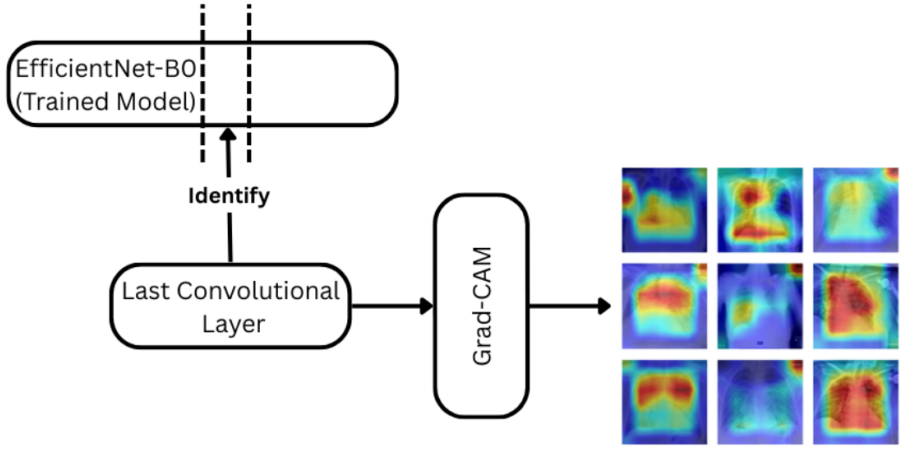


Figure 6. Grad-CAM pipeline applied with the EfficientNet-B0 model trained. The penultimate convolutional layer is used to compute attention heatmaps for each prediction.The color gradients (red–yellow–blue) represent attention intensity, where red indicates regions of higher model focus and blue indicates lower attention.

For qualitative assessment of model attention across different prediction types, we selected one representative image per true positive (TP), true negative (TN), and false nega-

tive (FN) case. There were no false positives (FP) predictions in this test set; hence, that cell is marked as N/A. Figure 7 shows that the model tends to focus on the lower or peripheral regions of the lung in cases of COVID-positive, as documented radiographic patterns.

Figure 7. Prototype Grad-CAM visualizations for TP, TN, and FN samples. The blank FP panel is intentional, as no false-positive cases occurred in this test set. Red/yellow areas indicate high model attention.

4.2.3. GPT-Based Agentic Reasoning and Prompt Engineering

To simulate independent, clinically significant triage decision-making, we integrated a GPT-based agentic reasoning module into our system. The module accepts structured input from the prediction, confidence score, and Grad-CAM interpretation of the vision model and produces a clinical-style decision and rationale.

All test scenarios used the same model configuration and prompt structure to ensure reproducibility. Each prompt had four structured fields: (i) a triage request; (ii) a confidence score; (iii) Grad-CAM explanatory text; and (iv) a predicted label (Normal or COVID-19). To promote deterministic and concise reasoning, the GPT-3.5-turbo model was accessible via the OpenAI API with temperature = 0.2 and max_tokens = 300. Parsing the model's textual answer yielded three standardized triage outcomes—ALERT, FLAG, or LOG—as well as their justifications. To provide auditable decision logs, this configuration was deployed equally to all images.

We used the GPT-3.5 language model [49] via the OpenAI API [50], with temperature 0.2 and 'max tokens' 300 [51]. The token limit was selected to allow for coherent, paragraph-level explanations of around 200–250 words. As 1 token corresponds to roughly 0.75 words, this allows the agent to provide context, reasoning, and a decision in natural medical language. A temperature of 0.2 was chosen to promote deterministic, reproducible answers—an imperative requirement in high-stakes medical applications where randomness could introduce interpretive risk.

Each test case was rendered into a prompt containing:

- Image ID
- Predicted label (COVID-19 or Normal)
- Confidence score

- Ground truth label (for analysis only)
- A description of the Grad-CAM attention map

A sample production prompt is given below:

```
You are an AI clinical assistant. Based on the model prediction and Grad-CAM
image, determine whether the system should ALERT a doctor, LOG the result, or
FLAG for manual review.
Image: NORMAL(1562).jpg
Prediction: Normal
Confidence: 0.9999
Ground Truth: Normal
Explanation: The Grad-CAM has high activation in the upper and middle lung
zones.
Question: What action should the system take? Please explain your reasoning.
```

Example GPT-generated decision and rationale:

```
Decision: LOG
Reasoning: The model predicted Normal with very high confidence (99.99%), and
the Grad-CAM heatmap is in agreement with the expected normal lung appearance.
There is no abnormal activation sign, and the output aligns with the ground
truth. Logging the case is appropriate in this scenario.
```

Another example of an `ALERT` response is as follows:

```
Decision: ALERT
Reasoning: The model is calling COVID-19 with 100% certainty, and the Grad-CAM
heatmap is showing very dense bilateral activation in the lower lung fields-a
classic appearance for COVID-19 pneumonia. The ground truth is COVID-19. This
case should be referred to a physician for immediate follow-up.
```

All outputs were logged into a structured CSV file for downstream auditing and human-in-the-loop analysis. This layer adds autonomous decision support that is explainable, reproducible, and clinically grounded-a necessary step toward real-world deployment.

5. Experiments and Results

5.1. Model Evaluation (Tabular Data)

This section compares machine learning models to classify epidemiological hospitalization risk. We use common classification measures to assess the effectiveness of the Random Forest, Logistic Regression, and Extreme Gradient Boosting (XGBoost). XGBoost was chosen as the final model based on this comparative analysis because of its exceptional capacity to manage class imbalances and non-linear feature interactions. After that, we go into great detail about the hyperparameter tuning procedure that was used to improve the performance of the chosen model using GridSearchCV.

5.1.1. Evaluation Metrics

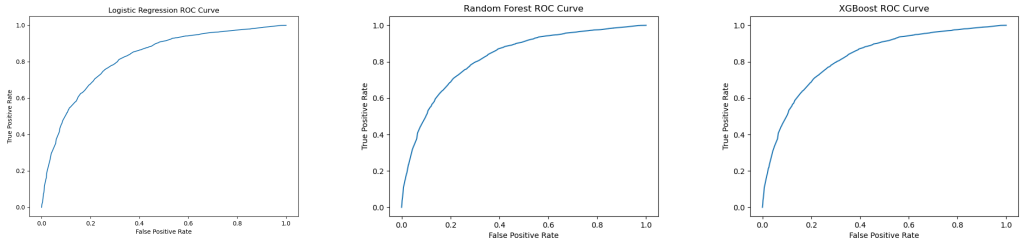
We used common classification metrics, such as accuracy, F1-score, mean F1-score, confusion matrix, and ROC curve, to evaluate the performance of the machine learning models.

The accuracy, F1-score for each class—Yes (Hospitalized) and No (Non-Hospitalized), Mean F1 Score—across the models are compiled in Table 2. The ROC curves are displayed in Figure 8.

Although overall accuracies were equivalent, XGBoost was preferred because it achieved the greatest recall and F1-score for hospitalized cases, decreasing false negatives in this clinically critical area.

Table 2. Comparison of model performance across traditional and gradient-boosting classifiers for hospitalization risk prediction. All models show comparable accuracy and F1-scores, confirming XGBoost as the most balanced and practical choice for the streaming pipeline.

Model	Accuracy	F1-Score (No)	F1-Score (Yes)	Mean F1
Logistic Regression	0.75	0.75	0.75	0.75
Random Forest	0.75	0.76	0.74	0.75
XGBoost	**0.75**	0.75	**0.76**	**0.75**
LightGBM	0.75	0.76	0.75	0.75
CatBoost	0.75	0.75	0.74	0.75



(**a**) Logistic Regression (**b**) Random Forest (**c**) XGBoost

Figure 8. ROC curves for Logistic Regression, Random Forest, and XGBoost. XGBoost shows the strongest trade-off between sensitivity and specificity.

XGBoost outperformed other models in terms of metrics balancing, achieving the highest F1-score for the minority class (Hospitalized) and competitive overall accuracy (Table 2). Furthermore, its ROC curve (Figure 8) outperforms the others across most false-positive rates, demonstrating a stronger capacity for discrimination. Because false negatives are expensive in this case, we prioritized F1-score for the hospitalized class, hence, XGBoost was chosen as the final classifier. To ensure thoroughness, we examined two more complex gradient-boosting frameworks, LightGBM and CatBoost. Both frameworks performed similarly to XGBoost (Accuracy ≈ 0.75 and F1 ≈ 0.75). XGBoost was selected as the best model for the epidemiology pipeline due to its balance of performance, speed, and interpretability. For interpretability, we report gain-based feature importance from the trained XGBoost model, providing a ranked list of the most influential features (see Figure 9). This analysis highlights which epidemiological attributes drive the model's decisions and complement the aggregate metrics above.

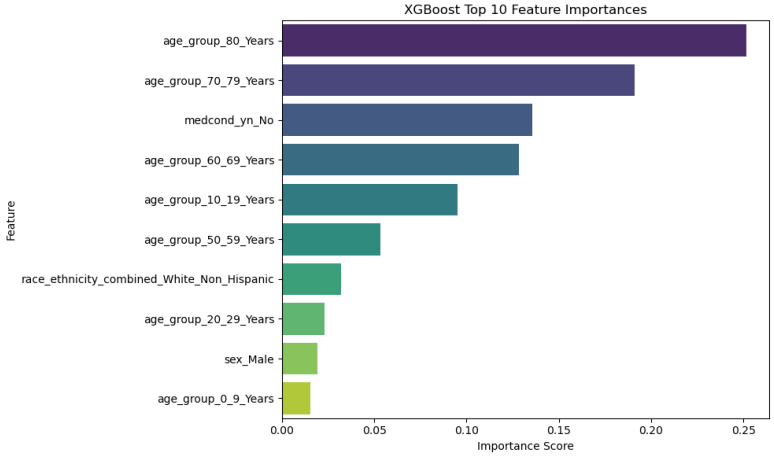


Figure 9. Top 10 Features and their Importance for XGBoost Model.

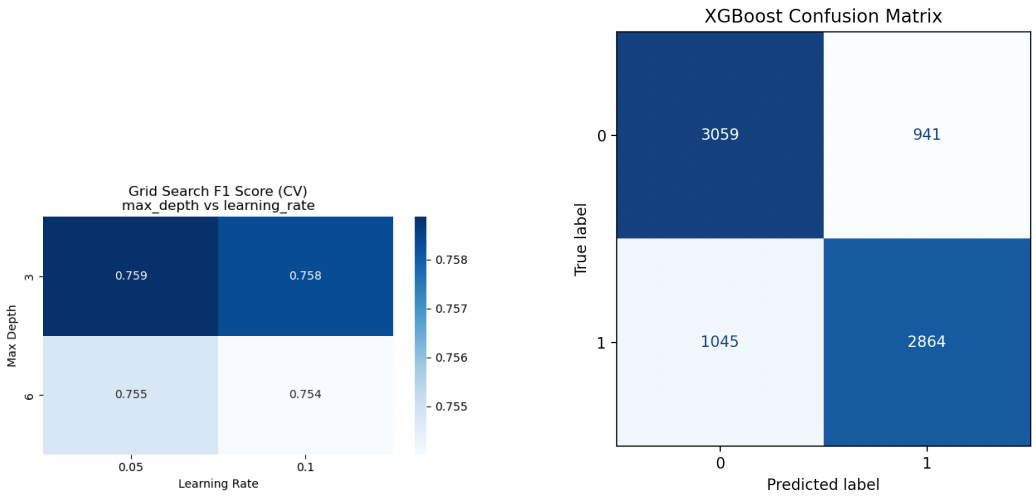
Appendix A, Figure A1 contains the confusion matrices for all models.

5.1.2. Model Selection and Hyperparameters

XGBoost was chosen as the final classifier based on the model comparison findings. We used GridSearchCV to adjust several important hyperparameters, such as `n_estimators`, `max_depth`, `learning_rate`, `subsample`, `colsample_bytree`, `min_child_weight`, and `gamma`, to further maximize its performance.

The optimal configuration was `max_depth=3` and `learning_rate=0.05`, which produced a cross-validated F1-score of 0.759 (Figure 10a). This was determined by the cross-validation grid search that focused on `max_depth` and `learning_rate`. Nevertheless, there was just a slight improvement over the default setup. Overall accuracy peaked at roughly 0.75, according to the confusion matrix on the held-out test set (Figure 10b).

These results imply that the expressiveness and predictive ability of the available features, rather than the selection of hyperparameters, are what limit XGBoost's effectiveness in this situation.



(**a**) GridSearchCV F1-score heatmap for XGBoost.

(**b**) XGBoost confusion matrix on the test set.

Figure 10. XGBoost hyperparameter adjustment and evaluation. Cross-validated F1-scores for various `max_depth` and `learning_rate` combinations are displayed in a heatmap (**a**). (**b**) The best XGBoost model's confusion matrix on the held-out test set.

5.2. Latency-Based Bloom Filter Ablation (Not Accuracy)

The integration of the Bloom filter into the end-to-end pipeline for hospitalization versus non-hospitalization case classification is explained in this subsection.

First, we provide a brief introduction to Bloom filters and explain how we believe they can enhance the classification pipeline. By contrasting execution durations with and without the Bloom filter, we demonstrate its practical usefulness and highlight how it reduces latency. Lastly, we provide experimental findings to support the advantages of this integration.

5.2.1. Introduction to Bloom Filter and Our Interpretation

We evaluate a Bloom filter as an $O(1)$ probabilistic pre-screening layer that is integrated into the epidemiological risk-prediction pipeline to reduce end-to-end delay. Before the entire XGBoost classifier evaluates a record, the Bloom filter performs a brief first-pass check to see if an incoming patient profile matches previously recognized high-risk patterns. This approach allows the system to quickly filter or rank cases in high-throughput streaming environments.

Because the classifier (*XGBoost*) is constantly applied for each item in our query, the Bloom filter does not affect training metrics, accuracy, or labels. To determine potential time savings during inference, it is evaluated solely as a latency optimization technique.

A Bloom filter is a fixed-size bit array with k hash functions capable of executing constant-time set-membership queries. The false-positive rate (FPR) of k hashes, m bits, and n inserted items is

$$\text{FPR} = \left(1 - e^{-kn/m}\right)^k, \qquad k = \tfrac{m}{n} \ln 2.$$

Bloom filters may produce false positives for hidden pieces, but they do not provide false negatives for objects that have been placed. In this situation, a false positive indicates that a new patient is momentarily tagged as possibly high-risk, resulting in an extra model check rather than a wrong clinical assessment. The filter is built using compact signatures of hospitalized profiles from the training stream (i.e., we emphasize "insert" those keys). Hashing uses branchless lookups and double hashing to approximate k-independent hash algorithms. Avoid the phrase "trained on positives"; the filter is built rather than taught.

Algorithm 1 outlines the simple pre-screening process applied during streaming.

Algorithm 1 Bloom filter pre-screening logic

```
Input: Record r, Bloom filter B, trained model M
if r ∈ B then
    Flag as "possibly high-risk"
    Send to classifier M for detailed evaluation
else
    Send to classifier M (normal path)
end if
Output: Model prediction
```

This approach enables a lightweight latency ablation test, comparing the pipeline's overall inference time with and without the Bloom filter. Only timing gains are reported, but prediction performance remains constant because the classifier reviews each record at the end.

5.2.2. Experimental Results and Limitations

We evaluated the end-to-end execution times of two pipelines, (i) XGBoost only and (ii) Bloom filter + XGBoost, to evaluate the efficacy of Bloom filter integration. Both were tested over ten cycles, with 100,000 records per iteration.

The detailed results are shown in Table 3. With improvements of 3–6%, the Bloom filter pipeline outperformed XGBoost alone in several iterations (e.g., Iterations 1, 4, and 6). The change was either insignificant or somewhat in favor of the baseline model in other iterations. All things considered, these results show that Bloom filters can help lower pipeline latency in specific situations.The results demonstrate the feasibility of integrating a lightweight pre-screening layer for efficiency in high-throughput or resource-limited streaming environments, where even small timing gains scale to significant overall improvements, even though the observed latency reduction was only modest (3–6%).

The improvements were not constant throughout all runs, even if the Bloom filter occasionally beat the baseline. The main drawback of the dataset is its structure: hospitalized cases cover almost every possible combination, the feature space is dense, and there are only four categorical predictors. As a result, the Bloom filter's overall ability to reduce model effort was limited.

Regardless of these constraints, we include the Bloom filter for two reasons. First, it demonstrates that a lightweight, space-efficient probabilistic data structure can be included

as an $O(1)$ pre-screen without impacting labels or accuracy, giving a repeatable engineering template for higher-throughput deployments. Second, it creates a baseline for future *gating* versions (where some records can safely skip the model with guardrails). Our dataset has a dense feature space, a few categorical fields, and relatively high effective prevalence, so the observed gains are modest and variable. We position Bloom filtering as an *optional latency optimization*, likely to yield larger benefits in sparser, higher-dimensional streams or when C_{model} is higher (e.g., remote/GPU inference).

Table 3. Comparison of Bloom filter with XGBoost versus XGBoost alone (100,000 rows each iteration) in terms of execution time.

Iteration	Bloom + XGBoost (s)	XGBoost Only (s)
1	9.52	9.90
2	9.04	9.12
3	9.88	9.35
4	9.31	9.70
5	9.22	9.28
6	9.46	10.12
7	9.19	9.21
8	9.85	9.14
9	9.28	9.74
10	9.36	9.42

5.3. Synthetic Data for Streaming Simulation

In order to replicate real-time streaming conditions, this section explains the methods used to create synthetic data from the tabular dataset. A Random Generator, a Distribution-based Sampling strategy, and a Generative AI model based on CTGAN were the three approaches that were assessed. We also tested three different training epochs (300, 500, and 1000) for CTGAN in order to evaluate how convergence affected the calibre of the samples that were produced.

These techniques created artificial data streams, which were then incorporated into the Apache Kafka and Spark pipeline to allow for stress testing of the real-time system under realistic and continuous load.

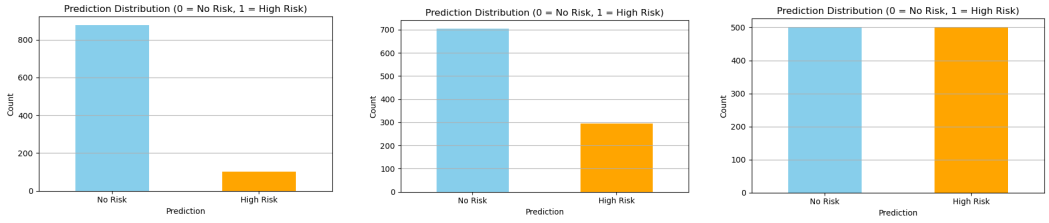
5.3.1. Generator Comparison: Random, Distribution-Aware, CTGAN

This part compares three methods for creating artificial tabular data streams: CTGAN, a distribution-aware generator, and a random generator. The following is a description of each technique:

- Random Generator: This method ignores the statistical characteristics of the original dataset and generates feature values entirely at random. Despite being easy to use, this approach does not maintain realistic class proportions or inter-feature interactions.
- Distribution-Aware Generator: This method uses the empirical statistical distributions of each feature in the original dataset to sample synthetic values. Class representation is improved over random generation; however, feature-to-feature dependencies are not captured, and features are still treated separately.
- CTGAN: A generative adversarial network created especially for tabular data is called Conditional Tabular GAN (CTGAN). In order to produce realistic synthetic records that better maintain class balance and correlations, it models both marginal distributions and inter-feature relationships. In order to assess convergence and sample quality, we trained CTGAN for three distinct epochs: 300, 500, and 1000.

By contrasting the class distributions of their synthetic samples, we assessed each approach. Figure 11 shows how the three methods were able to achieve class balance.

CTGAN outperforms both the random and distribution-aware approaches, yielding a more realistic and balanced distribution between the *Hospitalized* and *Non-Hospitalized* classes.



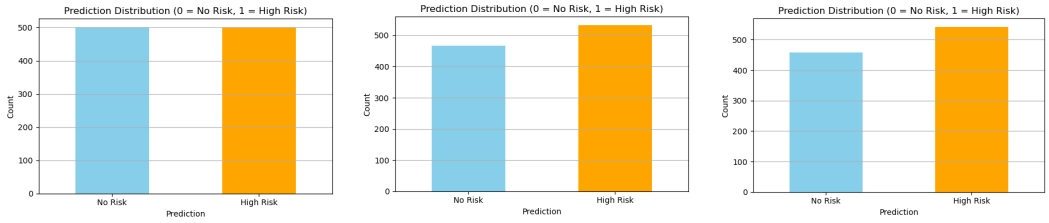
(**a**) (**b**) (**c**)

Figure 11. Comparison of class distributions generated by (**a**) Random, (**b**) Distribution-Aware, and (**c**) CTGAN approaches. CTGAN achieves the most balanced and realistic class proportions.

5.3.2. CTGAN Training and Epoch Tuning

CTGAN created synthetic data that resulted in a realistic and balanced distribution between the two classes, as was covered in the preceding section. By altering the number of epochs, we investigated the impact of training time in order to further assess its performance. In particular, models trained for 300, 500, and 1000 epochs were compared. For every epoch, the resulting class distributions are displayed in Figure 12. Class distributions generated by CTGAN trained for 300, 500, and 1000 epochs are compared in Figure 12. The 300-epoch model tends to underfit and fails to completely represent inter-feature relationships, although achieving a relatively balanced distribution (Figure 12a).

Training for 500 epochs produced the most balanced and consistent results, suggesting good convergence (Figure 12b). On the other hand, instability and slight divergence were introduced when training was extended to 1000 epochs (Figure 12c), indicating the beginning of overfitting.



(**a**) CTGAN (300 epochs) (**b**) CTGAN (500 epochs) (**c**) CTGAN (1000 epochs)

Figure 12. Comparison of class distributions generated by CTGAN at different training epochs. While 300 epochs show underfitting, 500 epochs provide the best balance and stability. Extending to 1000 epochs introduces signs of overfitting and instability.

500 epochs were therefore chosen as the ideal training period since it provided the finest balance between stability and realism in the synthetic data that was produced.

5.3.3. Realism and Use in Kafka-Spark Pipeline

Our tabular pipeline's last stage involved integrating the synthetic data generator, Bloom filter, and top-performing XGBoost model into the Kafka–Spark streaming framework. While Spark Streaming applied user-defined functions that sequentially queried the Bloom filter and activated the XGBoost classifier for cases designated as high-risk, Kafka acted as the ingestion layer, receiving events. Structured JSON format was used to write the results for later analysis and archiving.

By simulating realistic operating conditions, this configuration allowed for the near real-time examination of patient records under continuous input flow. No more numerical findings are presented here because this stage reflects pipeline execution rather than

a distinct categorization task. Rather, it confirms that the suggested approach, which combines low-latency streaming and predictive accuracy, can be operationalized inside a scalable big data framework.

5.4. Model Evaluation (Chest X-Rays)

This part provides the quantitative comparison of various deep learning vision models for the classification of chest X-rays. We present performance comparisons on several architectures, i.e., MobileNetV2, ResNet-18, and EfficientNet-B0, using standard classification metrics. From this comparison, we justify the selection of EfficientNet-B0 as the final model. We also analyze the confusion matrix to view common misclassifications and determine model weaknesses and strengths.

5.4.1. Classification Metrics and Performance

To compare model performance for chest X-ray classification, we experimented with three convolutional models: ResNet-18, MobileNetV2, and EfficientNet-B0. All models were trained with the same preprocessing, optimizer (Adam), and batch size, but the number of epochs varied over {10, 20, 30, 40}. Performance was estimated with standard metrics of classification: accuracy, precision, recall, and F1-score—each computed separately on the COVID-19 and Normal classes.

Accuracy is the global percentage of test images properly classified. However, in clinical cases like COVID-19 detection, class-specific measures are more relevant. Precision measures the proportion of positive predictions that were indeed correct (minimizing false positives), and Recall assesses the ability to detect all actual positives (minimizing false negatives). The F1-score is the harmonic mean of precision and recall and is of special relevance in imbalanced datasets where both types of errors are costly.

Table 4 summarizes test accuracy and F1-scores for two classes for models and epochs. To better represent model behavior visually, we also explored how precision and recall vary per class versus epochs. Figure 13 presents these trends in a 2×2 fashion—COVID vs. Normal, precision vs. recall. These plots give more information about the stability and sensitivity of classes over time.

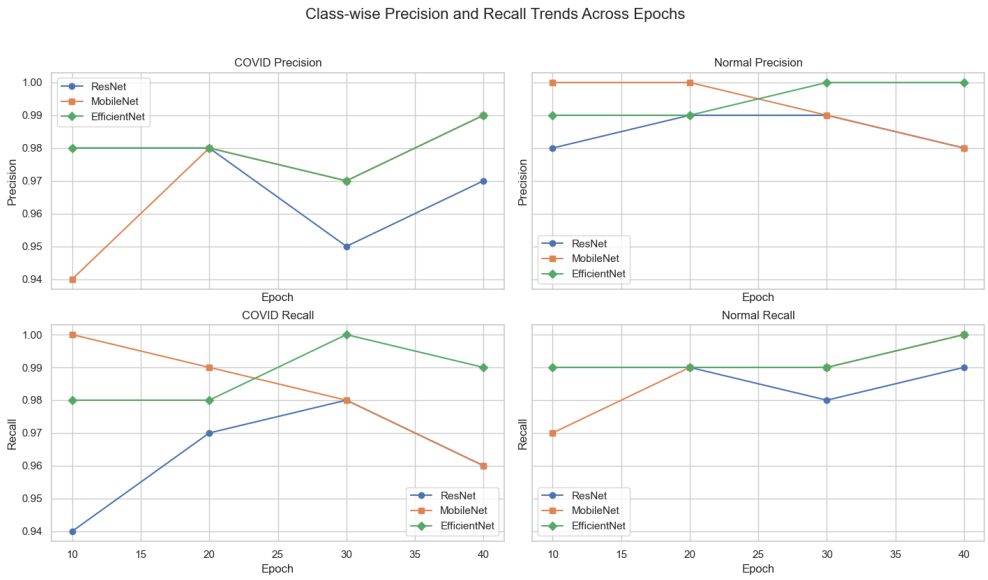


Figure 13. Class-wise precision and recall trends across training epochs for ResNet-18, MobileNetV2, and EfficientNet-B0. Each subplot isolates a specific metric and class, enabling detailed interpretation of model sensitivity and specificity over time.

Table 4. Performance comparison of deep learning models across training epochs.

Model	Epoch	Accuracy	COVID F1-Score	Normal F1-Score
ResNet-18	10	97.8%	0.96	0.98
	20	98.5%	0.97	0.99
	30	97.9%	0.96	0.98
	40	98.1%	0.96	0.99
MobileNetV2	10	98.1%	0.97	0.99
	20	99.1%	0.98	0.99
	30	98.6%	0.98	0.99
	40	98.5%	0.97	0.99
EfficientNet-B0	10	98.8%	0.98	0.99
	20	99.1%	0.98	0.99
	30	98.7%	0.98	0.99
	40	99.5%	0.99	0.99

Although all of the models performed well, EfficientNet-B0 consistently reported higher accuracy and class-wise F1-score across all epochs. Most importantly, at epoch 40, EfficientNet-B0 reported the highest test accuracy (99.5%) and F1-score (0.99) for both classes with a negligible difference in precision and recall. These results make it the strongest and most generalizable architecture to be utilized in downstream Grad-CAM and GPT-based decision reasoning in our pipeline.

Figure 13 provides an accurate class-by-class precision and recall behavior of each model across training epochs. The top row provides precision of COVID and Normal classes, and the bottom row represents recall behavior. EfficientNet-B0 has extremely high precision and recall of both classes in all epochs, leading to nearly perfect performance at epoch 40.

ResNet-18 is more volatile—especially at the first few epochs—with difficulty in maintaining COVID-19 prediction accuracy and with lower stability in recall. MobileNetV2 performs well but also drops slightly in COVID recall towards subsequent epochs. This comparison positively confirms that EfficientNet-B0 not only achieves the best accuracy and F1-score, but also maintains class-wise stable performance, especially in the clinician-sensitive COVID-19 detection task. These results further corroborate the selection of EfficientNet-B0 @ Epoch 40 as the selected model for downstream explainability and LLM-based decision logs.

5.4.2. External Evaluation on Unseen Test Set

To validate the generalizability of our top-performing model, EfficientNet-B0 (40 epochs), we evaluated its performance against a totally unseen external chest X-ray test set. The test set was acquired independently outside the training/validation pipeline and reflects real-world deployment conditions.

Figure 14 illustrates the resulting confusion matrix. The model generalized nicely, having 317 Normal and 112 COVID-19 samples classified correctly and only four false negatives and no false positives—a critical point for medical screening. The corresponding classification report (Table 5) confirms an overall accuracy of 99.3%, with F1-scores of 0.99 and 0.98 for the Normal and COVID classes, respectively.

Table 5. Classification report of EfficientNet-B0 on the unseen test set.

Class	Precision	Recall	F1-Score	Support
Normal	0.99	1.00	0.99	317
COVID-19	1.00	0.97	0.98	116
Avg/Total	0.99	0.99	0.99	433

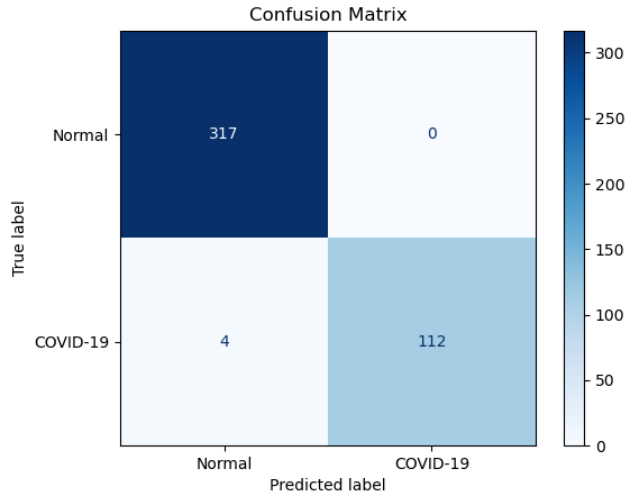


Figure 14. Confusion matrix for EfficientNet-B0 tested on an unseen test set.

Moreover, the confidence score distribution plot in Figure 15 shows the model's confidence in making predictions. Most of the predictions are in the high-confidence range (>0.98), affirming its reliability for downstream Grad-CAM interpretation and GPT-driven reasoning. The gray region represents the overlap between the confidence distributions of the COVID-19 and Normal classes, where both exhibit similarly high confidence values.

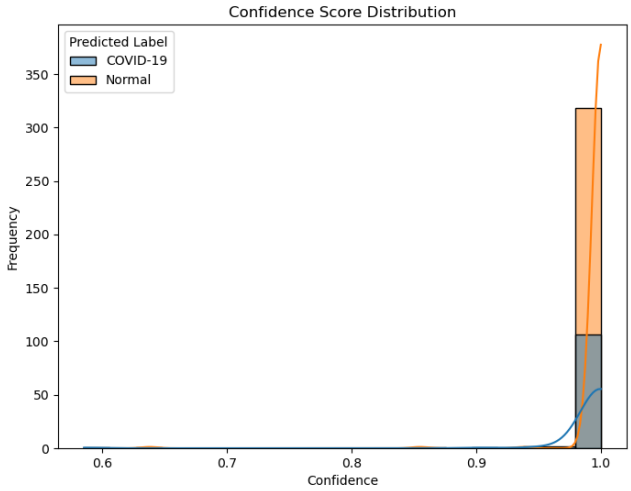


Figure 15. Confidence score distribution for predictions of EfficientNet-B0 on the unseen test set.

5.5. Grad-CAM Visual Explanations

This section explains the application of Grad-CAM (Gradient-weighted Class Activation Mapping) to visualize and understand our selected classifier, EfficientNet-B0, predictions. Grad-CAM enables post-hoc localization of class-discriminative regions in chest X-rays by generating heatmaps that highlight regions most responsible for the model's decision.

We add the Grad-CAM visualization module to the final convolutional layer of EfficientNet-B0. This is a conscious choice: convolutional layers retain spatial information and learn hierarchical features, whereas subsequent fully connected layers lose positional structure. Using the final convolutional layer ensures that the generated attention map captures localized areas of the image that had a direct influence on the model's classification.

Besides offering interpretability, Grad-CAM is also an activation validation method, which allows us to verify whether the model is paying attention to medically possible

areas—such as bilateral opacities or lower lobe consolidations—particularly in COVID-positive cases.

Correct vs. Incorrect Attention Patterns

Figure 16 shows the extreme difference between adequate and inadequate attention patterns in COVID-19 patients. The left image is a true positive (TP) prediction where the model correctly diagnoses the chest X-ray as COVID-19, and the Grad-CAM heatmap is centered around the lower and peripheral lung zones—regions previously known to be radiologically significant for COVID pathology.

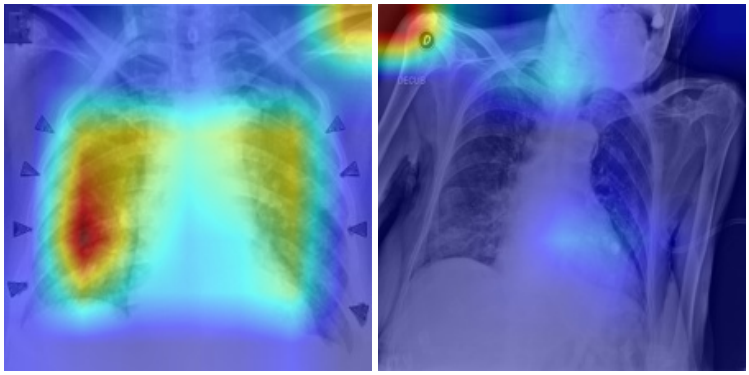


Figure 16. Grad-CAM attention patterns. (**Left**): Correctly classified COVID-19 case with strong activation in the bottom left lung, expected in healthy pathology. (**Right**): Incorrectly classified COVID-19 case (False Negative), where Grad-CAM attention is misplaced in irrelevant upper-body regions. Such visual misplacement indicates distraction or confusion by the model to irrelevant features. The color intensity (red = high model attention, blue = low attention) visualizes the regions influencing model prediction. Such visual misplacement indicates distraction or confusion by the model to irrelevant features.

On the other hand, the incorrect image shows a false negative (FN), where a COVID-positive image was misclassified as Normal. In this instance, active areas in non-germane areas of the anatomy, i.e., the shoulder and upper chest, are shown through Grad-CAM visualization. This mismatch suggests that the model either overlooked implicit pulmonary clues or was misled by image artifacts. These trends affirm the need for explainability in AI diagnosis and why agentic decision-making layers must be employed to indicate uncertain or risky errors.

In general, Grad-CAM heatmaps suggest that the model had learned to focus radiologically important regions—particularly lower lung lobes in COVID-positive samples—without spurious activation in normal cases. Consistency with clinical expectation boosts confidence in the model's decision and provides a foundation for GPT-based agentic reasoning integration, as discussed in the next section.

5.6. GPT-Based Agentic Reasoning

For the final step of our pipeline, we integrate GPT-based agentic reasoning to enable a prototype decision-support module with GPT-based reasoning. Following EfficientNet-B0, producing a classification and Grad-CAM as a visualization of spatial attention, this is presented as a clinical prompt and input to a GPT-3.5-turbo model. The language model, a radiology assistant, interprets the context of the image and the output of the model to recommend whether or not the case needs to be `ALERT` to a doctor, `FLAG` for a reviewer, or `LOG` as a typical result. Reuniting vision, interpretability, and language into a single pipeline is the move towards autonomously interpretable clinical AI systems.

We examined every decision log produced during inference on the external test dataset to objectively assess the behavior of the GPT-based reasoning layer. One structured reasoning output was generated for each chest X-ray case, which corresponded to one of three decision types: *ALERT*, *FLAG*, or *LOG*. Out of 433 external test cases, 317 (73.27%) were classified as LOG, 112 (25.81%) as ALERT, and only 4 (0.92%) as FLAG, as shown in Table 6. A distinct and persistent trend could be seen in the mean confidence scores: the highest mean confidence (0.998) was seen in LOG cases, followed by ALERT (0.993) and FLAG (0.872). This relationship between decision type and confidence shows that the reasoning layer operates consistently and predictably, automatically recording high-confidence, normal forecasts and issuing ALERT choices mainly for important or low-confidence circumstances.

Table 6. Quantitative summary of GPT-based reasoning outcomes on the external test dataset and their corresponding model confidence values.

Decision Type	Count	Percentage	Mean Confidence
LOG	317	73.27%	0.998
ALERT	112	25.81%	0.993
FLAG	4	0.92%	0.872

These findings, which were obtained solely from the external validation dataset, validate the agentic reasoning module's dependability for interpretable and auditable triage decisions in real-world deployment scenarios by demonstrating that it generalizes effectively and aligns with classifier confidence.

6. Discussion

This section provides an overview of our study's principal findings, with a focus on the key contributing elements in terms of performance, interpretability, and automation. We describe results that were strongly consistent with our design requirements, in addition to the constraints encountered when performing experiments.

6.1. Key Observations and Findings

This section summarizes the key findings from both parts of the proposed dual pipeline.

Part 1: Predicting hospitalization risk: To estimate hospitalization risk based on patient demographics and comorbidities, a real-time Kafka pipeline was developed. CTGAN enhanced streaming classification stability by correcting class imbalances while preserving feature dependencies, beating the other two synthetic-data techniques. Integrating a Bloom Filter enabled a lightweight, early-stage triage strategy for rapidly matching high-risk profiles in high-throughput circumstances.

Part 2: The Reasoning and Imaging Pipeline: A GPT-based reasoning layer for auditable triage notes, Grad-CAM for visual explanation, and EfficientNet-B0 for chest X-ray classification were all incorporated into a prototype decision-support system. The lower lung lobes, which are associated with ground-glass opacities, were consistently highlighted in the model's Grad-CAM heatmaps, confirming radiological data and validating clinical interpretability. Misdirected attention and occasional false negatives underlined the continued importance of explainability in clinical AI. Despite the study of GAN-based image synthesis, conventional augmentation provided adequate diversity and stability, indicating that, when generative models are unstable, it remains an effective strategy for validating medical imaging models.

An external test set of 433 previously unseen chest X-rays was used to evaluate the EfficientNet-B0 classifier and prove the imaging model's ability to generalize. With an

accuracy of 99.3% and F1-scores of 0.99 for normal and 0.98 for COVID-19, the model yielded no false positives. Furthermore, as evidenced by Grad-CAM images, the model consistently focused on clinically relevant locations, specifically the lower lung lobes associated with ground-glass opacities, demonstrating that predictions are based on true radiological features rather than overfitting noise. These findings show that the model does not overfit to the training data and has acceptable generalizability.

6.2. Limitations

Our research demonstrates that merging multimodal pipelines for real-time pandemic risk prediction is feasible; however, there are a number of limitations. First, the instability of GAN-based chest X-ray synthesis reduced its robustness across imaging circumstances, impeding research into data augmentation methods other than standard ways. Future studies will look into more reliable generating models, such as StyleGAN-ADA and diffusion models. Second, computational constraints hindered comparisons with larger ensemble architectures (such as DenseNet), and extending architectural baselines remains an important next step. Third, the reasoning layer based on GPT-3.5 was a proof of concept for automated explainable triage, but its application was limited by the API's cost and the model's capabilities. Advanced LLMs will be included in future versions for qualitative review. Finally, the epidemiological forecasts were based on CDC case-surveillance data that was only partially disseminated, which could contain noise. Although CTGAN-generated records were internally validated, generalizability must be proved via external validation with carefully selected hospital datasets.

7. Conclusions and Future Work

This work established a dual big-data pipeline that comprises a real-time epidemiological hospitalization-risk prediction model as well as an imaging-based diagnostic module. We used CTGAN-based stress testing, XGBoost with a Bloom filter pre-screen, and the CDC COVID-19 Case Surveillance dataset for near-real-time risk forecasting in a Kafka–Spark system. At the same time, EfficientNet-B0 with Grad-CAM explanations and a GPT-based reasoning layer enabled auditable decision logging and interpretable chest X-ray triage. Together, these components demonstrate that scalable, explicable, and nearly real-time clinical decision support for pandemic preparedness is achievable. Future research will explore advanced generative models like StyleGAN-ADA and diffusion models, extend the agentic reasoning workflow through the MCP framework with benchmarking and safety guardrails, and integrate these pipelines into multimodal systems that integrate larger and more varied datasets to enable clinically deployable pandemic-triage solutions.

Author Contributions: Conceptualization, V.S.P. and M.K.; methodology, V.S.P.; software, M.K.; validation, M.K. and V.S.P.; formal analysis, V.S.P., B.P. and N.N.; investigation, M.A. and V.S.P.; resources, B.P. and M.K.; data curation, N.N. and M.A.; writing—original draft preparation, M.K. and B.P.; writing—review and editing, V.S.P. and M.K.; visualization, M.A. and N.N.; supervision, V.S.P.; project administration, V.S.P.; funding acquisition, V.S.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: This study was exempt from ethical review and approval because it only used publicly accessible, de-identified datasets and does not include human subjects.

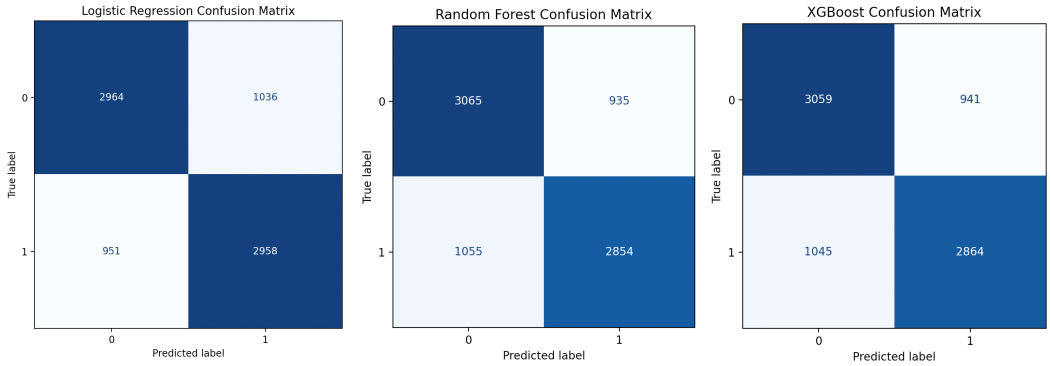
Informed Consent Statement: Patient consent was waived because the study used only publicly available, de-identified data.

Data Availability Statement: Public datasets: The (i) CDC COVID-19 Case Surveillance Public Use dataset [7] and the (ii) TCIA chest X-ray collection and COVID-19 Radiography Database [33] were

used. When accepted, synthetic tabular generators, configurations, and reproducibility seeds will be made public; editors and reviewers can request access to the private GitHub (web platform, accessed October 2024) repository. The repository does not contain any actual patient-level data.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Appendix A. Additional Figures



(**a**) Logistic Regression (**b**) Random Forest (**c**) XGBoost

Figure A1. Confusion matrices for Logistic Regression, Random Forest, and XGBoost. XGBoost achieves the best balance between true positives and true negatives.

Appendix B. Reproducibility Information

All experiments were conducted in the following environment to ensure reproducibility:

- **Operating System:** macOS Sonoma 14.5
- **Programming Language:** Python 3.10
- **Core Libraries:** scikit-learn 1.3, XGBoost 1.7, LightGBM 4.1, CatBoost 1.2, pandas 2.1, NumPy 1.26, Matplotlib 3.8, Seaborn 0.13
- **Deep Learning Frameworks:** PyTorch 2.0, Torchvision 0.15
- **Large Language Model API:** OpenAI GPT-3.5-turbo
- **Hardware Acceleration:** Apple Metal Performance Shaders (MPS) backend

These configurations were maintained for all model training and evaluation steps, enabling full reproduction of the reported results on compatible macOS systems.

References

1. Pokhrel, S.; Chhetri, R. A literature review on impact of COVID-19 pandemic on teaching and learning. *High. Educ. Future* **2021**, *8*, 133–141. [CrossRef]
2. Bojdani, E.; Rajagopalan, A.; Chen, A.; Gearin, P.; Olcott, W.; Shankar, V.; Cloutier, A.; Solomon, H.; Naqvi, N.Z.; Batty, N.; et al. COVID-19 pandemic: Impact on psychiatric care in the United States. *Psychiatry Res.* **2020**, *289*, 113069. [CrossRef]
3. Ko, J.Y.; Danielson, M.L.; Town, M.; Derado, G.; Greenlund, K.J.; Kirley, P.D.; Alden, N.B.; Yousey-Hindes, K.; Anderson, E.J.; Ryan, P.A.; et al. Risk factors for coronavirus disease 2019 (COVID-19)–associated hospitalization: COVID-19–associated hospitalization surveillance network and behavioral risk factor surveillance system. *Clin. Infect. Dis.* **2021**, *72*, e695–e703. [CrossRef]
4. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.
5. Vrdoljak, J.; Boban, Z.; Vilović, M.; Kumrić, M.; Božić, J. A Review of Large Language Models in Medical Education, Clinical Decision Support, and Healthcare Administration. *Healthcare* **2025**, *13*, 603. [CrossRef]
6. Chen, T.; Guestrin, C. Xgboost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 785–794.

7. Centers for Disease Control and Prevention (CDC). COVID-19 Case Surveillance Public Use Data. 2025. Available online: https://data.cdc.gov/Case-Surveillance/COVID-19-Case-Surveillance-Public-Use-Data/vbim-akqf/about_data (accessed on 15 July 2025).
8. Tan, M.; Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
9. Bloom, B.H. Space/time trade-offs in hash coding with allowable errors. *Commun. ACM* **1970**, *13*, 422–426. [CrossRef]
10. Olken, F.; Rotem, D. Random sampling from databases: A survey. *Stat. Comput.* **1995**, *5*, 25–42. [CrossRef]
11. Chakraborty, S.; Fremont, D.; Meel, K.; Seshia, S.; Vardi, M. Distribution-aware sampling and weighted model counting for SAT. In Proceedings of the AAAI Conference on Artificial Intelligence, Quebec, QC, Canada, 27–31 July 2014; Volume 28.
12. Xu, L.; Skoularidou, M.; Cuesta-Infante, A.; Veeramachaneni, K. Modeling tabular data using conditional gan. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; Volume 32.
13. Liang, C.; Lyu, T.; Weissman, S.; Daering, N.; Olatosi, B.; Hikmet, N.; Li, X. Early Prediction of COVID-19 Associated Hospitalization at the Time of CDC Contact Tracing using Machine Learning: Towards Pandemic Preparedness. *Res. Sq.* 2023, *preprint*. [CrossRef]
14. Schwab, P.; DuMont Schütte, A.; Dietz, B.; Bauer, S. Clinical predictive models for COVID-19: A systematic study. *arXiv* **2020**, arXiv:2005.08302. Available online: https://arxiv.org/abs/2005.08302 (accessed on 10 November 2025). [CrossRef] [PubMed]
15. Hwangbo, S.; Kim, Y.; Lee, C.; Lee, S.; Oh, B.; Moon, M.; Kim, S. Machine learning models to predict the maximum severity of COVID-19 based on initial hospitalization record. *Front. Public Health* **2022**, *10*, 1046296. [CrossRef] [PubMed]
16. Hassan, F.; Shaheen, M.; Sahal, R. Real-time healthcare monitoring system using online machine learning and spark streaming. *Int. J. Adv. Comput. Sci. Appl.* **2020**, *11*, 650–659. [CrossRef]
17. Waehner, K. Real Time Analytics with Apache Kafka in the Healthcare Industry. Available online: https://www.kai-waehner.de/blog/2022/04/04/real-time-analytics-machine-learning-with-apache-kafka-in-the-healthcare-industry/ (accessed on 10 July 2025).
18. AlMohimeed, A. Cloud-based real-time enhancement for disease prediction using Confluent Cloud, Apache Kafka, feature optimization, and explainable artificial intelligence. *PeerJ Comput. Sci.* **2025**, *11*, e2899. [CrossRef]
19. Waehner, K. The State of Data Streaming for Healthcare with Apache Kafka and Flink in 2023. *Blog Post*. Available online: https://www.kai-waehner.de/blog/2023/11/27/the-state-of-data-streaming-for-healthcare-in-2023/ (accessed on 27 November 2023).
20. Fang, M.; Dhami, D.; Kersting, K. DP-CTGAN: Differentially private medical data generation using CTGANs. In *Machine Learning and Knowledge Extraction*; Springer: Berlin/Heidelberg, Germany, 2022; pp. 141–160.
21. Kang, H.Y.J.; Batbaatar, E.; Choi, D.W.; Choi, K.S.; Ko, M.; Ryu, K.S. Synthetic tabular data based on generative adversarial networks in health care: Generation and validation using the divide-and-conquer strategy. *JMIR Med. Inform.* **2023**, *11*, e47859. [CrossRef] [PubMed]
22. Ziegeldorf, J.H.; Pennekamp, J.; Hellmanns, D.; Schwinger, F.; Kunze, I.; Henze, M.; Hiller, J.; Matzutt, R.; Wehrle, K. BLOOM: Bloom filter based oblivious outsourced matchings. *BMC Med. Genom.* **2017**, *10*, 44. [CrossRef]
23. Akter, S.; Shamrat, F.J.M.; Chakraborty, S.; Karim, A.; Azam, S. COVID-19 detection using deep learning algorithm on chest X-ray images. *Biology* **2021**, *10*, 1174. [CrossRef]
24. Constantinou, M.; Exarchos, T.; Vrahatis, A.G.; Vlamos, P. COVID-19 classification on chest X-ray images using deep learning methods. *Int. J. Environ. Res. Public Health* **2023**, *20*, 2035. [CrossRef] [PubMed]
25. Singh, T.; Mishra, S.; Kalra, R.; Satakshi; Kumar, M.; Kim, T. COVID-19 severity detection using chest X-ray segmentation and deep learning. *Sci. Rep.* **2024**, *14*, 19846. [CrossRef]
26. Suara, S.; Jha, A.; Sinha, P.; Sekh, A.A. Is Grad-CAM Explainable in Medical Images? In *Computer Vision and Image Processing*; Springer Nature: Cham, Switzerland, 2024; pp. 124–135. [CrossRef]
27. Moreau, L. AI Explainability with Grad-CAM: Visualizing Neural Network Decisions. Available online: https://www.edgeimpulse.com/blog/ai-explainability-with-grad-cam-visualizing-neural-network-decisions (accessed on 20 July 2025).
28. Zhang, H.; Ogasawara, K. Grad-CAM-Based Explainable Artificial Intelligence Related to Medical Text Processing. *Bioengineering* **2023**, *10*, 1070. [CrossRef]
29. Centers for Disease Control and Prevention (CDC). Coronavirus Disease 2019 (COVID-19). 2025. Available online: https://www.cdc.gov/covid/index.html (accessed on 10 July 2025).
30. Hao, Y.; Cheng, C.; Li, J.; Li, H.; Di, X.; Zeng, X.; Jin, S.; Han, X.; Liu, C.; Wang, Q.; et al. Multimodal Integration in Health Care: Development with Applications in Disease Management. *J. Med. Internet Res.* **2025**, *27*, e76557. [CrossRef]
31. Vinodhini Ravikumar, F.C.M. How Multimodal AI Is Impacting Healthcare. Available online: https://www.forbes.com/councils/forbestechcouncil/2025/04/29/how-multimodal-ai-is-impacting-healthcare/ (accessed on 10 August 2025).

32. Simbo AI. Exploring the Integration of Multimodal Real-World Data in Healthcare: A New Frontier for Personalized Treatment. Available online: https://www.simbo.ai/blog/exploring-the-use-of-multimodal-real-world-data-in-precision-medicine-and-its-benefits-for-treatment-personalization-1000978/ (accessed on 15 August 2025).
33. The Cancer Imaging Archive (TCIA). COVID-19-AR: Chest Imaging with Clinical and Genomic Correlates Representing a Rural COVID-19 Positive Population. 2020. Available online: https://www.cancerimagingarchive.net/collection/covid-19-ar/ (accessed on 15 July 2025).
34. Rahman, T. COVID-19 Radiography Database. Available online: https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database (accessed on 10 July 2025).
35. Radford, A.; Metz, L.; Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv* **2015**, arXiv:1511.06434.
36. Perez, L.; Wang, J. The effectiveness of data augmentation in image classification using deep learning. *arXiv* **2017**, arXiv:1712.04621. [CrossRef]
37. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
38. Armbrust, M.; Das, T.; Torres, J.; Yavuz, B.; Zhu, S.; Xin, R.; Ghodsi, A.; Stoica, I.; Zaharia, M. Structured streaming: A declarative api for real-time applications in apache spark. In Proceedings of the 2018 International Conference on Management of Data, Houston, TX, USA, 10–15 June 2018; pp. 601–613.
39. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
40. Nick, T.G.; Campbell, K.M. Logistic regression. In *Topics in Biostatistics*; Humana Press: Totowa, NJ, USA, 2007; pp. 273–301.
41. Chen, D.; Zhang, Q.; Zhu, Y. Efficient sequential decision making with large language models. *arXiv* **2024**, arXiv:2406.12125. [CrossRef]
42. Liu, Y.; Yao, Y.; Ton, J.F.; Zhang, X.; Guo, R.; Cheng, H.; Klochkov, Y.; Taufiq, M.F.; Li, H. Trustworthy llms: A survey and guideline for evaluating large language models' alignment. *arXiv* **2023**, arXiv:2308.05374.
43. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **2002**, *86*, 2278–2324. [CrossRef]
44. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
45. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
46. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
47. Powers, D.M. Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv* **2020**, arXiv:2010.16061. [CrossRef]
48. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
49. OpenAI. GPT-3.5: OpenAI Language Model. Available online: https://platform.openai.com/docs/models/gpt-3-5 (accessed on 23 September 2025).
50. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. *Adv. Neural Inf. Process. Syst. (NeurIPS)* **2020**, *33*, 1877–1901.
51. Wolf, T.; Debut, L.; Sanh, V.; Chaumond, J.; Delangue, C.; Moi, A.; Cistac, P.; Rault, T.; Louf, R.; Funtowicz, M.; et al. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online, 16–20 November 2020; pp. 38–45.