Bulletin of Business and Economics, 13(1), 653-661
https://bbejournal.com
https://doi.org/10.61506/01.00252

Analyzing the Software Architecture of ML-based Covid-19 Detection System: Future Challenges and Opportunities

Romaisa Sabir[1], Salman Hassan[2], Muhammad Hamza Ittifaq[3], Muhammad Waseem Iqbal[4], Mohsin Raza[5], Ahmad Raza[6], Pehroze Fatima[7]

Abstract
Two major study topics have emerged because of the challenges in software architecture and ML working together, as modern software systems produce a vast amount of data that is supported particularly by machine learning (ML), and artificial intelligence (AI) to produce useful insights. Software architecture for machine learning systems that primarily concerned with creating architectural methods for creating ML systems more effectively; ii) ML for Software architectures is concerned with creating ML methods for better-developing software systems. This study focuses on the ML-based software systems' architecture to highlight the many architectural methods currently in use. To more clearly identify a set of acceptable standards for designing ML-based software systems, we explore four crucial components of software architecture in this work that demand the focus of ML and software developers. These areas are based on an ML-based software system for addressing challenges in the COVID-19 detecting system.
Keywords: Software architecture, COVID-19, Machine learning, Evaluation of Architecture

1. Introduction

As Artificial intelligence (AI) and machine learning (ML) are still gaining popularity, and widespread adoption, they present unique challenges for development practices, deployments, data quality assurance, and other areas. In addition to the difficulties faced by traditional software systems, these issues require new approaches to architecture for ML-based systems (Wan et al., 2019). While ML systems generate vast amounts of data, they also face a range of architectural challenges (Hamid, Ibrar, et al., 2024). ML can help address some of the challenges facing ML-based systems but better architecting practices are also needed to ensure these systems thrive on data. This combination of challenges has resulted in two main research areas at the intersection of software architecture and ML. The first area involves developing architectural techniques to speed up the development of ML systems, while the second area focuses on using ML to improve software architecture (Hamid, Muhammad, et al., 2023).

1.1. Background

There is an amount of data generated by current software systems. We live in a software-powered data-driven world with a wealth of data produced by many sources such as online apps, smartphones, and sensors. As computing infrastructure has developed over time, these data have been supported particularly by machine learning (ML), and artificial intelligence (AI) to produce useful insights. It has clarified the path for the creation of software platforms and services that enable things like Netflix recommendations, Google search results, and driverless vehicles. However, along with the difficulties associated with a traditional software system, the growing use of AI, particularly ML, has created new challenges related to deployments, development procedures, ensuring data quality, etc. Better architectural techniques are required to overcome these issues with ML-based software systems (*2021 IEEE International Conference on Software Maintenance and Evolution ICSME 2021*, n.d.). On the one side, software systems produce a lot of data but have various architectural challenges. On the other side, ML-based systems rely on data to function but also demand stronger architectural standards. ML can help to solve some of these problems (Muccini & Vaidhyanathan, 2021).

Due to the limitations in both software architecture and machine learning, two main study fields have emerged: i) ML system software architecture. It is mostly devoted to creating architectural approaches to more effectively designing machine learning (ML) systems, while ML for Software Architecture is focused on creating ML techniques for better building software systems. In this study, I focus on the former end of the spectrum to provide an overview of the numerous architectural practices used when creating ML-based software systems (Hamid, Muhammad, Iqbal, Bukhari, et al., 2022). To better, define a standard set of principles for designing ML-based software systems. In this paper, I address four essential components of software architecture that deserve the attention of both ML and software practitioners to properly establish a common set of rules for creating ML-based software systems. These components were developed based on a software approach that uses machine learning to address problems with chest X-ray imaging in COVID-19 identification (Hamid, Iqbal, Fuzail, Muhammad, Nazir, et al., 2022).

1.2. Software Architecture and Machine Learning

The design and organization of software systems are referred to as software architecture, whereas machine learning is the process of developing algorithms that can learn from data and make predictions or judgments based on it.

Overall, incorporating machine learning into a software architecture requires careful consideration of the system's design and goals, as well as the specific requirements of the machine learning algorithms being used. One issue is how to create a software system that meets both functional and non-functional requirements. On the other hand, it addresses how to address the numerous machine learning-related challenges. The process, stakeholders, concerns, etc. are divided even if they exist together in the same software in the modern world.

Figure 1 illustrates the high-level view of an ML-based software system and how it is perceived by modern software architects. While the system is unified, it can be seen as two major subsystems: the machine learning subsystem, which focuses on data, algorithms, and models, and the software subsystem, which encompasses components, connectors, and their interactions. The software subsystem utilizes machine learning models and continuously generates the data needed for the machine learning

[1] Department of Computer Science, Superior University, Lahore, Pakistan, romaisasabir98@gmail.com
[2] Department of Computer Science, Superior University, Lahore, Pakistan, maliksalmanhassan9@gmail.com
[3] Department of Computer Science, Superior University, Lahore, Pakistan, hamzakamboh82@gmail.com
[4] Department of Software Engineering, Superior University, Lahore, Pakistan, waseem.iqbal@superior.edu.pk
[5] Faculty of Computer Science Alshifa Institute of Life Sciences Pakistan, razam2888@gmail.com
[6] Department of Computer Science, University of Engineering and Technology Lahore, Pakistan, **m.ahmadraza457@gmail.com**
[7] Department of Computer Science, University of Engineering and Technology Lahore, Pakistan, **pehrozefatima@gmail.com**

subsystem. Each subsystem involves different stakeholders with their respective concerns. The modern software architect plays a vital role in coordinating between these two subsystems, which possess distinct characteristics, properties, and team dynamics. This coordination raises questions about various aspects of architecting, including standardizing architecting practices and addressing barriers that arise from this setup. The paper goes on to describe existing practices in architecting ML-based software systems and highlights the future directions to develop improved approaches for architecting such systems (Hamid, Muhammad, Iqbal, Nazir, et al., 2022).

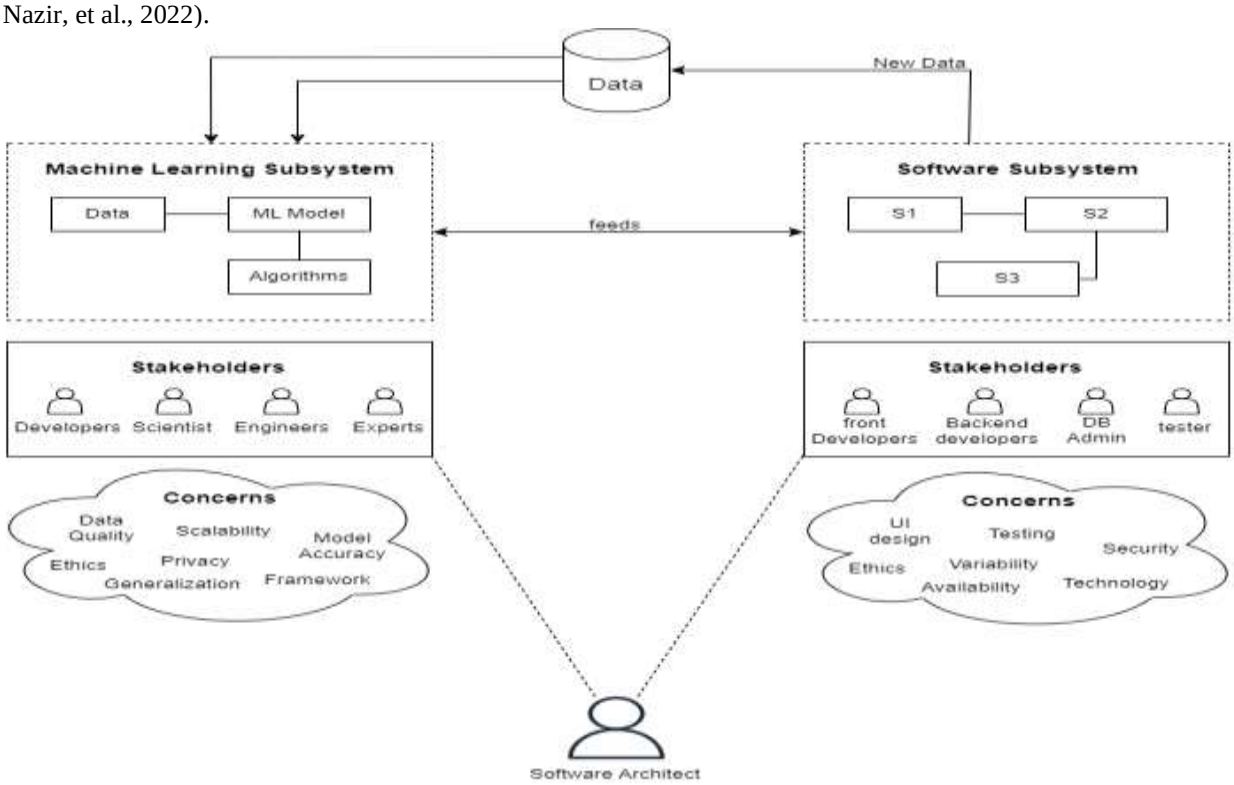

Fig 1: High-level view of ML-based Software system

2. Architecting ML-Based Software System: Identification of COVID-19

2.1. Using Machine Learning

Coronavirus 2 causes fatal acute respiratory syndrome, (SARS-CoV-2; formerly proviso-named 2019 distinct coronavirus or 2019-nCoV) disease (COVID-19) and has become a significant public health concern (Hamid, Aslam, et al., 2024). Castiglioni and his colleagues described the application of a deep learning model, specifically a convolutional neural network (CNN), to analyze chest X-ray images of patients suspected of having COVID-19. The study used a dataset of chest X-rays from both COVID-19 positive and negative patients, with the AI model trained to differentiate between the two. The results of the study showed promising findings, indicating that the AI model achieved high accuracy in detecting COVID-19 infection from chest X-ray images. It is suggested that the use of AI in this context could potentially assist healthcare professionals in the rapid triage and diagnosis of COVID-19 cases, particularly in regions with a high volume of cases and limited resources (Hamid, Muhammad, Iqbal, Nazir, et al., 2022). However, there are some limitations to different identification of COVID-19 studies. These include the relatively small sample size, the absence of external validation, and the retrospective nature of the analysis. The paper highlights the need for further research and validation to determine the generalizability and real-world applicability of AI in chest X-ray analysis for COVID-19 diagnosis.

Overall, the study demonstrated the potential of AI, specifically deep learning models, in aiding the diagnosis of COVID-19 infection using chest X-ray images. It provides an initial experience from Lombardy, Italy, which suggests that AI-based approaches could be valuable tools in supporting healthcare professionals during the COVID-19 pandemic. This epidemic's spread can be stopped because of the early detection of COVID. But the main issue is the limited supply of test kits. To tackle this, AI is useful and even used in COVID-19 identification and prediction. A model for COVID-19 detection from chest X-rays using CheXNet, architecture accurately identified the COVID and normal binary classes with 99.9% accuracy. Applying the ChestXray14 dataset, the CNN network CheXNet was trained to look for anomalies in chest X-rays. The Authors widened their model, in general, to recognize each of the 14 diseases in the chestXray14 dataset (Hamid, Iqbal, Fuzail, Muhammad, Basit, et al., 2022). They employed Densenet121, a pre-trained version of it, in their model to distinguish COVID-19 from binary classes. With subsequent advancements, it was put into practice in real-time COVID-19 detection settings. An overall understanding of gathering data, processing it, choosing a model, training it, evaluating it, deploying, and ongoing improvement is needed to construct an ML-based software system for COVID-19 identification utilizing artificial intelligence with chest X-ray imaging. It also needs access to a sizable database of chest X-ray pictures and diagnostic imaging and imaging knowledge.

1) Data Collection: The first step in building an ML-based system for a sizable collection of X-ray images of the chest will be gathered for COVID-19 identification utilizing chest imaging.

The transfer learning approach could be used to develop similar models for other medical imaging applications. It emphasizes the potential of transfer learning techniques for creating deep learning models for the processing of medical images and contends that these models could be crucial in the worldwide fight against the COVID-19 epidemic.

Chest X-ray images have been the main source of data for several studies that implement artificial intelligence techniques for the disease's autonomous classification. Promising results have been made thus far in this area. In the most recent scholarly

literature, some issues were brought up by the automatic classification of COVID-19 using artificial intelligence approaches (Narayan Das et al., 2022). It can be difficult for radiologists to distinguish COVID-19 from CXR images. They must be aware of the typical patterns of the illness, which are frequently shared with other widespread pneumonia and lead to incorrect diagnoses. A more accurate method for illness identification is CT imaging. The data that have been revealed so far, in contrast to what has been suggested; seem to favor CXR over CT. A small number of COVID-19-positive CXR images are made openly accessible online for use by the scientific community. Most studies add negative images from other data sources to complete their data. Different sets of these images differ significantly from one another. When assessing using a subset of the original set of photos, produces exceptionally good outcomes for the automatic categorization of COVID-19. When assessing the trained models in their own sets, several studies find limited consistency to no generalization power. Even models developed by the use of preprocessing methods, which attempted to remove the biases inherent in the data sets, produced very modest results. As a result, the majority of findings to date, as published in the Scientific research, attribute models that notice the traits of the training sets. Since there is no good evaluation protocol, most of the developed models are still of limited use in clinical settings.

While machine learning (ML)-based COVID detection systems have shown promise in aiding diagnosis and screening processes, several potential faults and limitations need to be considered. Some of the faults in ML-based COVID detection systems include: Limited training data is the main fault in the COVID-19 detection system. ML models require large and diverse datasets to learn effectively. In the case of COVID-19 detection, access to comprehensive and balanced datasets can be challenging, particularly during the early stages of a pandemic or in resource-constrained regions. Limited training data may lead to biases and inaccuracies in the ML model's predictions.

3. Objective

To emphasize the numerous architectural techniques now in use, this study concentrates on the ML-based software systems' architecture. In this paper, we address four essential components of software architecture that deserve the consideration of both ML and software practitioners to properly establish a common set of rules for creating ML-based software systems. Architectural framework, architectural process, self-adaptive architecture, and architectural evaluation fall under this category.

3.1. The software architecture of ML-based system: Future challenges and opportunities

To emphasize the numerous architectural techniques now in use, this section discusses four essential components of software architecture that should be taken into account by ML practitioners as well as software developers to appropriately construct a set of guidelines for developing ML-based software systems. We describe in detail what currently exists in each of these categories and what we think needs to be done in the future. These components have been derived based on architecting ML-based software systems for addressing challenges in the COVID-19 detecting system.

3.2. Architecture Framework

When it comes to the architectural framework of a COVID-19 detection system, there can be several limitations. Here are some potential limitations that may arise:

Data Availability and Quality are the major ones. The effectiveness of a COVID-19 detection system heavily relies on the availability and quality of data. Limited access to diverse and comprehensive datasets can limit the accuracy and generalizability of the system. Additionally, if the data used for training the system is flawed or biased, it can lead to inaccurate results and potential disparities in detecting COVID-19 cases. Architectural frameworks may face challenges in scaling up to accommodate large-scale COVID-19 testing requirements. As the number of tests increases, the system should be able to handle a higher volume of data, perform computations efficiently, and handle increased traffic. Scaling up the infrastructure and computational resources to meet the growing demand can be a significant challenge. No detection system is perfect, and there is always a chance of false positives (incorrectly identifying someone as positive) or false negatives (incorrectly identifying someone as negative). False positives may result in needless panic and strain on healthcare resources, while false negatives can result in infected individuals going undetected and potentially spreading the virus. Deep learning models used in architectural frameworks can be complex and lack interpretability. It can be difficult to comprehend the logic behind the system's decisions and identify the features contributing to the detection outcome. In critical scenarios like COVID-19, interpretability, and explainability are crucial to gain the trust of users, healthcare professionals, and regulatory bodies. Integrating a COVID-19 detection system into existing healthcare infrastructure and workflows can be challenging. The system may need to integrate with electronic health records, laboratory information systems, and other healthcare software. Ensuring seamless deployment, compatibility, and interoperability with existing systems can be complex. Architectural frameworks must take ethical issues like data security, privacy, and informed permission into account. Collecting and storing sensitive health data raises concerns about privacy breaches and potential misuse. Proper safeguards should be in place to protect individuals' personal information and ensure compliance with relevant regulations and guidelines. COVID-19 has undergone several mutations, resulting in the emergence of different variants. The architectural framework should be adaptable and robust enough to detect these variants accurately. Keeping pace with the evolving nature of the virus and updating the system to detect new variants can be a continuous challenge. It is important to address these limitations while developing and implementing architectural frameworks for COVID-19 detection systems to ensure their reliability, accuracy, and ethical considerations are properly addressed. In the field of COVID-19 identification utilizing X-ray imaging and machine learning, several architecture frameworks have been used.

3.3. Existing Architecture Frameworks

CNNs have been widely employed for COVID-19 detection from X-ray images (Memon et al., 2023). Architectures like VGGNet, ResNet, DenseNet, and InceptionNet have been utilized to extract features and classify images. Transfer learning has been extensively applied by leveraging models that have already been trained using massive datasets like ImageNet (Hamid, Iqbal, Fuzail, Muhammad, Basit, et al., 2022). Researchers fine-tune models like ResNet, InceptionNet, or EfficientNet on X-ray images to detect COVID-19. Autoencoders, including convolutional variants, have been utilized to learn compressed representations of normal and COVID-19 X-ray images (Hamid, Iqbal, et al., 2023a). These learned representations can aid in classification tasks. RNNs, particularly Long Short-Term Memory (LSTM) networks, have been employed to capture temporal dependencies and analyze the progression of COVID-19 in X-ray images over time (Noaman et al., 2023). Ensemble models,

which combine multiple individual models (Hamid & Iqbal, 2022), have been explored to improve overall performance and robustness in COVID-19 detection. Ensemble techniques such as bagging, boosting, and stacking have been applied to CNNs or other architectures.

Table 1 summarizes the limitations and benefits of various ML models based on different data types commonly used in COVID-19 detection and medical diagnosis. It serves as a quick reference guide for researchers and healthcare professionals exploring ML-based approaches for COVID-19 detection and other medical diagnosis tasks. By understanding the limitations and benefits associated with each combination of data type and ML model, researchers can make informed decisions regarding model selection, data preprocessing, and potential trade-offs in accuracy and performance. However, it is essential to note that this table provides a generalized overview, and specific applications may require further investigation and validation based on the available data and research context.

Table 1: Comparison of various ML models based on different data types

Study	Data Type	ML Models and Techniques	Limitations	Benefits
1	Chest X-ray Images	Convolutional Neural Networks (CNN)	Limited spatial information in 2D images, Potential biases, limited availability	Fast processing for screening,
2	Chest X-ray Images	Transfer Learning, DenseNet201	Suffer from data bias and limited interpretability.	High accuracy
3	Chest X-ray Images	Deep Autoencoders	May require large datasets for training	Potential for improved accuracy
4	Chest X-ray Images	GANs	Introduce biases or produce unrealistic images	Enable data augmentation and enhance the robustness
5	CT Scans	Convolutional Neural Networks (CNN)	High radiation dose in CT scans	Detailed imaging for accurate detection
6		Transfer Learning	Costly and time-consuming imaging	Efficient in severe and complex cases
7		Deep Autoencoders	Large data storage requirements	Effective in detecting lung abnormalities
8	Clinical Notes	Deep Learning Architectures (RNN, LSTM, Transformer)	Unstructured and noisy text data	Incorporate textual information
9		Naive Bayes	Difficult to extract relevant features	Support early diagnosis and treatment
10		SVM	May require natural language processing	Predict disease progression
11	Time Series Data	Deep Learning Architectures (RNN, LSTM, Transformer)	Irregular and missing data points	Real-time monitoring and prediction
12		SVM	Sensitive to data preprocessing	Identify temporal patterns
13		Random Forest	May need specialized handling methods	Early detection of worsening condition
14		Gradient Boosting Machines (GBM)	Computationally expensive and prone to overfitting	High predictive accuracy and the ability to capture complex patterns
15	Clinical Features	Decision Trees	Limited features for complex scenarios	Easily interpretable by healthcare professionals
16		Naive Bayes	May not capture subtle patterns	Assist in risk assessment and prognosis
17		SVM	Incomplete representation of patients	Facilitate personalized treatment plans
18		Random Forest	susceptibility to overfitting with noisy data and limited representation	high accuracy and interpretable

3.4. Future development

Explainable Transfer Learning COVID-Net (ETL-COVID) is designed for COVID-19 detection from medical imaging data, such as chest X-rays or CT scans. The model leverages the benefits of transfer learning and incorporates explainability features to provide insights into its decision-making process, making it interpretable for healthcare professionals.

3.5. Key Features of ETL-COVID

- Transfer Learning: ETL-COVID uses transfer learning, a technique where a pre-trained deep learning model (usually trained on a large and diverse dataset, e.g., ImageNet) is adapted to a specific task with limited labeled data. By utilizing pre-trained weights, the model benefits from learned features from general medical image data, making it more data-efficient and improving its performance.
- CNN Architecture: ETL-COVID is built upon a Convolutional Neural Network (CNN) architecture. CNNs are known

for their ability to learn hierarchical features from images, enabling them to automatically extract relevant patterns and information from the input data.

- Interpretability through Attention Mechanism: To achieve interpretability, ETL-COVID incorporates attention mechanisms, such as Grad-CAM (Gradient-weighted Class Activation Mapping). These mechanisms generate heatmaps that highlight the regions in the medical images that significantly influence the model's predictions. This provides visual explanations for why the model made a specific decision, increasing trust and understanding for clinicians. Here's a mathematical explanation of Grad-CAM:
  f: $R^{H\times W\times C} \rightarrow R$ be the CNN model used for COVID-19 detection, where H and W are the height and width of the input image, and C is the number of channels. The output of the last convolutional layer of the CNN is denoted as A, which is a H′×W′×K feature map, where K is the number of filters in that layer.
  Grad-CAM aims to visualize the importance of different regions of the input image x concerning the model's final prediction f(x) by computing the gradient of the predicted class Y_c concerning the feature maps A of the last convolutional layer:
  $a_{k=}^{c} \frac{1}{Z} \sum i \sum j \frac{\partial y_c}{\partial A_{ij}^{k}}$
  Where:
  - a_k^c is the importance weight of the k-th feature map for the predicted class c.
  - Z is a normalization factor.
  - $\frac{\partial y_c}{\partial A_{ij}^{k}}$ represents the gradient of the predicted class score $with$ respect to the activation value A_{ij}^{k} in the k-th feature map.

  Next, Grad-CAM computes the class activation map $L_{GradCAM}^{c}$ by performing a weighted combination of the feature maps A using the importance weights a_k^c:
  $L_{GradCAM}^{c} = ReLU\ (\sum_{k=1}^{K} a_k^c A^k$
  Where A^k k-th feature map in the last convolutional layer.
  Finally, Grad-CAM generates the heatmap $H_{GradCAM}^{c}$ by upsampling the class activation map to the original input image size:
  $H_{GradCAM}^{c} = Upsample\ (L_{GradCAM}^{c})$

- Data Augmentation: To improve the model's robustness and reduce overfitting, ETL-COVID applies data augmentation techniques during training. This involves creating variations of the training images by applying transformations like rotation, scaling, and flipping, which diversifies the dataset and allows the model to learn from a more extensive range of variations.
- Efficient Inference: ETL-COVID is designed for efficient inference, enabling real-time or near-real-time predictions. This allows it to be used in point-of-care scenarios, where quick and accurate COVID-19 detection is critical for patient management and public health measures.
- Ensemble Approach: ETL-COVID can be part of an ensemble system that combines predictions from multiple models, each trained on different aspects of the data. Ensemble methods enhance the overall performance and robustness of the COVID-19 detection system.
- Ethical Considerations: ETL-COVID prioritizes patient privacy and ethical considerations. It adheres to data anonymization and follows strict security protocols to protect sensitive patient information, ensuring compliance with privacy regulations.
- Validation and Clinical Studies: ETL-COVID undergoes extensive validation and evaluation on diverse and real-world clinical datasets, representing different patient populations and imaging protocols. Rigorous clinical studies are conducted to assess its performance and effectiveness in detecting COVID-19.

ETL-COVID aims to provide an accurate, interpretable, and efficient solution for COVID-19 detection, assisting healthcare professionals in making informed decisions and improving patient care during the ongoing pandemic. However, it's essential to recognize that the success of ETL-COVID and any other model depends on access to high-quality data and continuous research advancements in machine learning and medical imaging.

4. Architecting Process

As the technology and understanding of COVID-19 continue to evolve, the architecting process for these detection systems will likely undergo improvements and refinements. Here are some potential developments to look out for in each phase of the architecting process:

What exists: Relevant X-ray images of COVID-19 patients and non-COVID-19 cases are collected from various sources. Data augmentation techniques may be applied to increase the diversity and size of the dataset. The collected data is preprocessed to enhance quality and remove noise. This may involve resizing images, normalizing pixel values, and applying filters or other image processing techniques. The appropriate ML architecture framework, such as CNNs or transfer learning models, is chosen based on the available data, computational resources, and performance requirements. According to Zhiyuan et al., the development of machine learning (ML) systems follows an iterative and incremental process. Initially, experiments are conducted to identify the most suitable algorithm or technique to fulfill specific requirements. This iterative approach often involves modifying the system's architecture to accommodate the chosen algorithm or technique. This could entail enhancing data collection mechanisms or adding additional components for pre-processing. Consequently, continuous collaboration between software architects, ML stakeholders (with a medical background), and software system stakeholders (with a programming background) is necessary. The divergent development methodologies and backgrounds of these teams can influence the various stages of the architecting process.

The selected model is trained on the preprocessed dataset. This involves splitting the data into training and validation sets, configuring hyperparameters, and optimizing the model using techniques like backpropagation and gradient descent. The trained model is evaluated using separate test data that was not used during training. Evaluation metrics such as accuracy, precision, recall, and F1 score are calculated to assess the model's performance. If the model's performance is not satisfactory, further iterations of training, hyperparameter tuning, or model architecture adjustments may be conducted to enhance its accuracy and generalization. The trained model is validated using independent datasets or external validation studies to assess its performance in real-world scenarios and compare it against existing approaches. Once the model demonstrates satisfactory performance, it can be deployed as part of a software system or integrated into existing healthcare infrastructure for practical use. Considerations like privacy, security, and regulatory compliance are taken into account during deployment.

Potential development: It's important to note that the architecting process is iterative and may involve continuous improvements and updates as new research findings and technological advancements emerge.

I. **Architecture Design:** Advancements in ML research may lead to the development of more specialized models for COVID-19 detection, tailored to handle specific data types (e.g., X-ray images, genomic data) and address particular challenges. Architecting may involve the design of ensemble models that combine the strengths of multiple ML models to improve overall accuracy and reliability in COVID-19 detection. COVID-19 detection systems may adopt domain-specific architectures that incorporate domain knowledge from medical experts, virologists, and epidemiologists to enhance the model's performance. As new variants of the virus emerge, the architecture may be adapted to ensure effective detection, considering changes in image features or clinical presentations.

II. **Architecture Analysis:** Architecture analysis will include evaluating ML models for potential biases and ensuring fairness in COVID-19 detection across different demographic groups to avoid disparities in healthcare outcomes. COVID-19 detection systems may incorporate techniques to estimate model uncertainty, providing more reliable and informative predictions, particularly in cases with limited or uncertain data.

III. **Architecture Realization:** The realization phase will focus on developing ML-based COVID-19 detection systems that can scale to handle large volumes of data and deploy across various healthcare settings, including hospitals, clinics, and mobile health units. Realization may involve optimizing ML models for edge computing, enabling faster and decentralized COVID-19 testing and diagnosis.

IV. **Architecture Representation:** An important outcome of the software architecting process is the creation of artifacts that can effectively communicate the system's architecture to relevant stakeholders.

Given the sensitive nature of healthcare data, architecture processes may include compliance with regulations such as GDPR and HIPAA, as well as ethical considerations surrounding data usage, bias mitigation, and transparency. Future architecture processes may focus on the seamless integration of ML-based COVID-19 detection systems into the clinical workflow, ensuring usability, interoperability, and integration with electronic health records (EHR) systems. Ongoing research and advancements in the field will likely contribute to further enhancements in the accuracy, reliability, and real-world deployment of these systems.

5. Self-adaptative architecture

Self-adaptative architecture gives the system the capacity to autonomously adapt its architecture (with minimal human intervention) to manage the numerous types of uncertainties that may result in a deterioration or failure of the Quality of Service. The ability of a system to dynamically modify its architecture in response to shifting circumstances or requirements is known as self-adaptive architecting. Using X-ray imaging and machine learning for COVID-19 detection, self-adaptive architecting can significantly increase system performance and flexibility. Though particular applications of self-adaptive architecting methods may differ, the following are some current strategies and prospective advancements in this field:

What exists: Since the phrase "Software Crisis" was first used in 1968 at the NATO Software Engineering Conference, the field of self-adaptive systems has developed over time [63]. Due to resource limitations, component failures, etc., modern software systems are prone to a variety of uncertainties. Self-adaptation, which enables the architectures to autonomously adapt to the various uncertainties to achieve the system's ultimate goals, has emerged throughout time as a potential remedy. Self-adaptive systems have been the subject of a significant amount of literature research. Some studies offered a thorough assessment of the many methods for designing self-adaptive systems. Various papers have been published in the self-adaptation literature in recent years that employ machine learning methods to improve the adaptability of conventional software systems. Additionally, efforts have been made to modify machine learning methodologies to more effectively guarantee robustness.

Based on variables like available resources, data features, and fluctuating performance requirements, the system can dynamically choose the best machine learning model for COVID-19 identification. As a result, the system can maximize detection accuracy and adjust to various conditions. The performance of a model can be greatly influenced by hyperparameters like regularization, batch size, and learning rate. The model's efficacy can be increased by automatically adjusting these hyperparameters based on real-time feedback through self-adaptive techniques like reinforcement learning or Bayesian optimization. Adaptive Data Preprocessing: Certain data preprocessing operations, such as filtering, normalization, and image scaling, can be modified by the unique properties of the input data. This guarantees that the X-ray images provide the most pertinent and appropriate data for the ML model. Runtime Configuration Adaptation: The system can modify its runtime configuration in response to several variables, including the availability of computational resources, network conditions, and processing speed requirements. To satisfy resource and performance limitations, the system, for instance, can optimize the inference process or dynamically modify the model's complexity.

Potential Developments in Self-Adaptive Architecting: The efficacy, flexibility, and robustness of ML-based COVID-19 detection systems may be improved by these prospective advancements in self-adaptive architecture.

I. **Architectural Adaptation by ML:** This refers to applying ML methods to implement architectural adaptations. For example, the machine learning methods from the machine learning subsystem (Figure 1) can be used to predict the uncertainties that the software subsystem might experience (such as excessive CPU usage, malfunctions, etc.). It can also be used to choose the optimal adaptation approach for the design on its own. These tactics could include

rearranging the component behavior, altering the software subsystem's structure, etc. However, inaccurate uncertainty estimates or poor strategy selection might result in less-than-ideal or impractical adaptations, which may have an unintentional impact on how a system operates. Furthermore, these methods must account for the probabilistic character of machine learning processes, which may contribute to uncertainty.

II. **Architectural Adaptation for ML:** The selection and use of the ML method affects the overall efficiency (response time, usage, etc.) of an ML-based software system. This is because the final model's complexity is determined by the algorithm chosen. For example, going back to the ML subsystem in Figure 1, we can tackle the same problem using multiple methods, which naturally yields different types of computationally complex models and offers different accuracy measurements. A more intricate deep learning technique could result in a more accurate and substantial model (such as one with multiple neural network layers). However, using the model may require additional CPU resources and processing time due to its density and increased number of number parameters, and software component(s) from the software subsystem Figure 1. On the other hand, employing a lighter model could provide better performance guarantees to the components that use it despite the loss of accuracy (e.g., reduced latency for creating a forecast and hence better response time). Different versions and types of ML models are generated at runtime throughout time by the ML subsystem using automated MLOps procedures. This would necessitate that the software subsystem uses techniques to automatically change between selecting ML models while taking the system's overall performance requirements into account.

III. **Architectural Adaptation of ML**: Accuracy and learning bias are two issues that ML faces [13]. Furthermore, as a result of data fluctuations over time, machine learning models experience model degradation, which causes the accuracy to gradually decline. For example, the ML system (Figure 1) may be using ML models that are optimized for particular sorts of data. However, as Figure 1 shows, additional data also enters the system over time, which may indicate that the models' predictions begin to deteriorate as well. This behavior may cause the software subsystem's components that consume the model to behave erratically (for example, a recommendation system may begin to give the user ridiculous recommendations). To ensure robustness, this may necessitate modifying the design or behavior of the neural network or machine learning algorithm itself (e.g., dynamically changing the layers of a deep neural network, adapting hyperparameters, etc.). This will become increasingly important in the upcoming years, particularly with the rise in popularity of Software 2.0 [14], which calls for the autonomous design and implementation of software components through machine learning techniques.

This suggests that more advanced methods will be needed:

- Ensemble learning, combining the predictions of multiple models trained with different architectural configurations. The ensemble can adapt by giving more weight to the models that perform better on specific data or conditions.
- Reinforcement learning techniques to adapt its hyperparameters, such as the architecture itself or the choice of pre-trained models. RL agents can learn to make architectural choices that optimize the model's performance.

Explainable Transfer Learning implements a learning rate scheduler that adjusts the learning rate during training. If the model's performance plateaus or deteriorates, the scheduler reduces the learning rate to allow the model to adapt more slowly. Conversely, if the model is learning quickly, the learning rate can be increased to speed up adaptation. It is designed to adapt its knowledge from a source domain (e.g., general medical images) to a target domain (COVID-19 cases) in real time. This adaptation can help the model respond to changes in the data distribution. ETL-COVID can incorporate reinforcement learning (RL) agents to optimize hyperparameters. These agents can autonomously explore different architectural configurations, regularization methods, and learning rates, adapting the model's architecture for improved performance.

More advanced self-adaptive methods should be created and used as the field's research advances to meet the demands and difficulties of COVID-19 detection by X-ray imaging. Heatmaps are utilized to accurately and understandably convey forecasts of COVID-19 situations using an XAI method called LIME. To improve explainability, the suggested model is also expanded using the LIME and heatmap approaches. XAI technologies provide explainability and transparency, which aid non-expert end users in understanding the black box AI model. It gives the user feedback and explains, i.e., by giving more details and going all the way down to the inner workings of the black box AI mod. The researchers detected COVID-19 with extremely high accuracy by using edge fuzzy images based on Explainable artificial intelligence for detection and identification. They attain an accuracy of 0.95 utilizing quantization technology. Even though the researcher's COVID-19 detection accuracy, sensitivity, and specificity were good. With the hope that it may be improved upon to hasten research in this field, the analysis's goal is to present radiologists, data scientists, and the scientific community with a multi-input CNN model that may be used to detect COVID-19 promptly.

6. Architecture Evolution

Architecture evolution is the process of continuously enhancing and changing a system's architecture through time through iterations. Architecture evolution is a critical component in improving the resilience, performance, and adaptability of the COVID-19 detection system when it comes to machine learning. An outline of current theories on the evolution of architecture and future directions for this field is provided below:

What Exists: The literature on the evolution of software architecture has undergone a great deal of attention. The architecture of contemporary software systems, particularly ML-based software systems, is always changing. It is anticipated that the design of these systems will continuously change during runtime. This is mostly because as time goes on, more data becomes accessible along with the diffusion of improved and newer algorithms. The learning algorithm may need to be updated as a result of this procedure. Additionally, this could mean installing a new database or altering the existing one to accommodate the additional data, modifying software components since the ML component's interface has changed, and so forth. Furthermore, more sophisticated methods are needed to explore the limits of data and learning at each stage of the evolution. Additionally, disruptive events have the potential to alter the dynamics of the system, necessitating an upgrade or downgrade in previously obtained knowledge, or even the loss of utility. More investigation and discussion are needed on how these circumstances should be handled and how to satisfy both functional and non-functional requirements by resolving them. ML models that are employed

to detect COVID-19 may be frequently retrained or adjusted using fresh data (Hamid, Iqbal, et al., 2023b). The model may learn from additional samples of tagged X-ray pictures as they become available, allowing it to detect patterns better and improve detection accuracy.

Potential Developments in Architecture Evolution: The data and model, dependent on the learning algorithm selected, are the main forces behind ML-based systems. The design of the entire system may need to adapt over time to accommodate any changes in requirements brought about by the data or algorithm to meet the overall requirements (Hamid, Aslam, et al., 2024). This raises two crucial points that should be taken into account:

I. **Data-induced evolution:** In ML-based systems is a pivotal strategy for ensuring the continued relevance, accuracy, and adaptability of these systems in a dynamic and data-rich environment. It involves the seamless integration of new and diverse data sources, enabling the system to process a wide array of data types and formats. This approach not only enhances data quality and cleaning processes but also equips ML models with the ability to handle imbalanced datasets more effectively (Salahat et al., 2023). Adaptive learning and real-time learning mechanisms empower these systems to autonomously refine their performance, making them agile in responding to changing data patterns and insights. Importantly, data-induced evolution incorporates measures for data drift detection and adaptation, ensuring that ML models remain accurate as data distributions evolve (Hamid, Muhammad, Iqbal, Nazir, et al., 2022) (Hamid, Ayub, et al., 2024). This adaptability extends to the personalization of services, with ML systems tailoring their responses based on individual user behavior or unique data characteristics (Hamid, Iqbal, Nazir, Muhammad, et al., 2022). Collaboration, both on a global scale and with ethical considerations, is central to responsible evolution, while continuous model updates ensure that these systems remain at the forefront of knowledge and utility in their respective domains.

II. **ML Algorithm-induced evolution:** Selecting the learning algorithm(s) to be utilized and the kind of model(s) that will be produced throughout the learning process is the second important component of any machine learning software system. Following architecture implementation, the machine learning component of the system often continues to produce new ML models regularly, contingent upon the availability of qualitative as well as quantitative information. Some common MLOps procedures are used to further deploy these models into production. To create more accurate or lighter models for performance, the learning algorithm may need to be adjusted over time. It might be necessary to change the model interface and gather additional data to achieve this. This can also necessitate changing the software's architecture to better meet the demands of the new learning algorithm. The evolution may encompass the integration of cutting-edge techniques, such as advanced optimization algorithms, novel loss functions, or state-of-the-art neural network architectures. Moreover, model interpretability methods can become more sophisticated, offering a deeper understanding of model decisions. Such algorithmic evolution ensures that ML systems can keep pace with the rapidly advancing field of machine learning and the ever-changing demands of data-driven applications, fostering a path to more accurate, efficient, and adaptable solutions (Hamid, Waseem Iqbal, Arif, Mahmood, et al., 2022).

There will be an increasing focus on the models' interpretability and explainability when evaluating COVID-19 detection systems. Gaining the confidence of regulatory agencies and medical professionals requires explainable AI. Scholars must evaluate the degree of interpretability offered by ETL-COVID systems. Research will continue to improve performance metrics related to COVID-19 detection. Other metrics, such as the F1 score, area under the receiver operating characteristic curve (AUC-ROC), and area under the precision-recall curve (AUC-PR), may be prioritized in addition to sensitivity, specificity, and accuracy. A crucial component of the evaluation will be how resilient the models are to changes in the distribution of data and how well they generalize to other populations. It will be crucial to use cross-validation methods that take into consideration various data splits and sources. The system's capacity to adjust to shifting data distributions, new viral strains, and changing medical procedures will be the main areas of assessment. Adaptability and strategies for continuous learning will be evaluated. Human-in-the-loop systems, such as ETL-COVID, may include input and interaction from healthcare professionals as a major part of the review process. These models of collaboration will be evaluated for efficacy. The assessment of COVID-19 detection systems may entail determining how well they adjust to various geographical conditions as well as how well international cooperation and data exchange enhance detection accuracy. Models such as ETL-COVID may require regulatory approval from agencies such as the FDA or CE marking. Compliance with medical device regulations will be part of the evaluation process.

7. Conclusion

The software architecture of ML-based systems has been the focus of our discussion in this work regarding the state of the software architecture of ML-based COVID-19 detection system. We think that future software architects' roles will shift from being simply another software architect to being ML-aware software architects based on the context of talks made in light of our experience designing an ML-based system. In essence, the architect's vision will not have a dividing boundary. There won't be any distinctions made at the design level between software and machine learning components, resulting in a more cohesive perspective. Instead, they will be seen as an additional component, similar to a data/event generator, data/event consumer, or model generator. Better architecting techniques can be achieved by using this perspective to help the architect understand an ML-based software system from the perspectives of software architecture and machine learning. We have suggested numerous paths in this work that will enable the community to make progress toward the achievement of this standard. The four topics we have examined in this study are based on our knowledge of designing the COVID-19 Detection system's ML-based software system.

References

2021 IEEE International Conference on Software Maintenance and Evolution ICSME 2021. (n.d.).

Hamid, K., Aslam, Z., Delshadi, A., Ibrar, M., Mahmood, Y., & Iqbal, M. waseem. (2024). *Empowerments of Anti-Cancer Medicinal Structures by Modern Topological Invariants*. *7*, 668–683.

Hamid, K., Ayub, N., Delshadi, M. A., Ibrar, M., Rahim, N. Z. A., Mahmood, Y., & Iqbal, M. W. (2024). Empowered corrosion-resistant products through HCP crystal network: A topological assistance. *Indonesian Journal of Electrical Engineering and Computer Science*, *34*(3), Article 3.

Hamid, K., Ibrar, M., Delshadi, A. M., Hussain, M., Iqbal, M. W., Hameed, A., & Noor, M. (2024). ML-based Meta-Model Usability Evaluation of Mobile Medical Apps. *International Journal of Advanced Computer Science and Applications (IJACSA)*, *15*(1), Article 1.

Hamid, K., & Iqbal, M. waseem. (2022). Topological Evaluation of Certain Computer Networks by Contraharmonic-Quadratic Indices. *Computers, Materials and Continua*, *74*, 3795–3810.

Hamid, K., Iqbal, M. waseem, Aqeel, M., Liu, X., & Arif, M. (2023a). *Analysis of Techniques for Detection and Removal of Zero-Day Attacks (ZDA)* (pp. 248–262).

Hamid, K., Iqbal, M. waseem, Aqeel, M., Liu, X., & Arif, M. (2023b). *Analysis of Techniques for Detection and Removal of Zero-Day Attacks (ZDA)* (pp. 248–262).

Hamid, K., Iqbal, M. waseem, Fuzail, Z., Muhammad, H., Basit, M., Nazir, Z., & Ghafoor, Z. (2022). *Detection of Brain Tumor from Brain MRI Images with the Help of Machine Learning & Deep Learning*.

Hamid, K., Iqbal, M. waseem, Fuzail, Z., Muhammad, H., Nazir, Z., Ashraf, M. U., & Bhatti, S. (2022). Empowerment Of Chemical Structure Used In Anti-Cancer And Corona Medicines. *Tianjin Daxue Xuebao (Ziran Kexue Yu Gongcheng Jishu Ban)/Journal of Tianjin University Science and Technology*, *55*, 41–54.

Hamid, K., Iqbal, M. waseem, Nazir, Z., Muhammad, H., & Fuzail, Z. (2022). Usability Empowered By User's Adaptive Features In Smart Phones: The Rsm Approach. *Tianjin Daxue Xuebao (Ziran Kexue Yu Gongcheng Jishu Ban)/Journal of Tianjin University Science and Technology*, *55*, 285–304.

Hamid, K., Muhammad, H., Iqbal, M. waseem, Bukhari, S., Nazir, A., & Bhatti, S. (2022). Ml-Based Usability Evaluation Of Educational Mobile Apps For Grown-Ups And Adults. *Jilin Daxue Xuebao (Gongxueban)/Journal of Jilin University (Engineering and Technology Edition)*, *41*, 352–370.

Hamid, K., Muhammad, H., Iqbal, M. waseem, Nazir, A., shazab, & Moneeza, H. (2023). Ml-Based Meta Model Evaluation Of Mobile Apps Empowered Usability Of Disables. *Tianjin Daxue Xuebao (Ziran Kexue Yu Gongcheng Jishu Ban)/Journal of Tianjin University Science and Technology*, *56*, 50–68.

Hamid, K., Muhammad, H., Iqbal, M. waseem, Nazir, Z., Irfan, D., & Rashed, R. (2022). Empowerments Of Chemical Structures Used For Curing Lungs Infections By Modern Invariants. *Jilin Daxue Xuebao (Gongxueban)/Journal of Jilin University (Engineering and Technology Edition)*, *41*, 439–458.

Hamid, K., Waseem Iqbal, M., Arif, E., Mahmood, Y., Salman Khan, A., Kama, N., Azmi, A., & Ikram, A. (2022). K-Banhatti Invariants Empowered Topological Investigation of Bridge Networks. *Computers, Materials & Continua*, *73*(3), 5423–5440.

Memon, A., Nazir, A., Hamid, K., & Iqbal, M. waseem. (2023). An efficient approach for data transmission using the encounter prediction m. Ashraf nazir khalid hamid muhammad waseem iqbal. *Tianjin Daxue Xuebao (Ziran Kexue Yu Gongcheng Jishu Ban)/Journal of Tianjin University Science and Technology*, *56*, 92–109.

Muccini, H., & Vaidhyanathan, K. (2021). *Software Architecture for ML-based Systems: What Exists and What Lies Ahead*.

Narayan Das, N., Kumar, N., Kaur, M., Kumar, V., & Singh, D. (2022). Automated Deep Transfer Learning-Based Approach for Detection of COVID-19 Infection in Chest X-rays. *Ingenierie et Recherche Biomedicale*, *43*(2), 114–119.

Noaman, S., Ibrahim, A., Ali, L., Iqbal, M., Ashraf, A., Haseeb, U., Muneer, S., Almajed, R., & Hamid, K. (2023). *Playing the video games during COVID-19 pandemic and its effects on player's well-being* (p. 5).

Salahat, M., Said, R., Hamid, K., Haseeb, U., Ghani, E., Abualkishik, A., Iqbal, M. W., & Inairat, M. (2023). *Software Testing Issues Improvement in Quality Assurance* (p. 6).

Wan, Z., Xia, X., Lo, D., & Murphy, G. (2019). How does Machine Learning Change Software Development Practices? *IEEE Transactions on Software Engineering*, *PP*, 1–1.