Zhe-Ming Kuo¹, Kuan-Fu Chen¹, and Yi-Ju Tseng¹

¹Affiliation not available

January 2, 2024
MoCab: A Framework for the Deployment of Machine Learning Models across Health Information Systems

Zhe-Ming Kuo, Kuan-Fu Chen and Yi-Ju Tseng, Member, IEEE

Abstract—Machine learning models are increasingly vital for clinical decision making. However, the integration of these models into health information systems (HISs) poses significant challenges, mainly due to the disparate formats of electronic health records (EHR) in various healthcare facilities. In response to this challenge, we proposed Model Cabinet Architecture (MoCab), a framework designed to leverage fast healthcare interoperability resources (FHIR) as the standard for data storage and retrieval when deploying machine learning models across various HISs. MoCab simplifies the deployment process by importing and configuring saved prediction models, facilitating patient data retrieval from FHIR servers, and transmitting it to the prediction model for analysis. The framework further incorporates Clinical Decision Support (CDS) Hooks for issuing clinical alerts and uses Substitutable Medical Apps Reusable Technologies (SMART) on FHIR to develop Apps for displaying laboratory results. Finally, MoCab offers the ability to continuously fine-tune and enhance the performance of deployed models over time. We demonstrate MoCab’s efficacy through the successful integration of three model types, highlighting its potential in streamlining decision making amidst heterogeneous EHRs. Our proposed MoCab framework not only promotes the reusability of machine learning models across multiple EHRs but also contributes to improving clinical decision making, offering a promising solution to the challenges of integrating machine learning models into healthcare settings.

Index Terms—Fast Healthcare Interoperability Resources, Clinical Decision Support, Machine Learning, SMART on FHIR.

I. INTRODUCTION

PREDICTION models have become indispensable in modern healthcare [1]. They play a crucial role in clinical decision making by extracting information from patient’s diagnosis history, laboratory results, demographic information, and extensive clinical data [2]. These models are evolving with new research and serve as an instrument for accurately predicting patient outcomes and aiding diagnostic processes [3].

The increasing adoption of predictive models in clinical practice has highlighted the need for a unified management system capable of streamlining workflows [4], [5]. Integrating health information systems (HISs) with predictive models is imperative to automate and streamline prediction processes. However, this integration faces a challenge due to the diversity of electronic health record (EHR) formats in clinical institutions [6], [7]. This diversity requires repeated adjustments to data retrieval procedures when transitioning between EHR systems and predictive models, which imposes a complex task of applying a single model consistently across different HISs.

The precision of clinical decision support (CDS) systems is based on a regular update of predictive models with new data. This is essential to counteract the degradation of performance caused by outdated information [5]. The shift from traditional evidence-based medicine to “medicine-based evidence” underscores the importance of considering a broader data context surrounding patients [8], [9]. In this context, continuous training of machine learning models assumes vital importance [5], [10], [11], empowering these models to adapt to emerging trends and evolving patterns, thus ensuring precise predictions [12].

To address these challenges, we proposed the Model Cabinet Architecture (MoCab). MoCab serves as a centralized repository designed to store and manage diverse predictive models, including scoring, machine learning, and deep learning models. Its objective is to streamline the utilization of patient data for clinical decision making, with models stored within an easily accessible “cabinet” and readily retrievable as needed. MoCab’s integration with the Fast Healthcare Interoperability Resources (FHIR) standard ensures compatibility with various EHR systems, thus enabling the effortless integration of predictive models with a wide variety of healthcare environments. Moreover, MoCab emphasizes continuous model training, allowing regular updates with newly acquired data, thus preserving the accuracy of predictions. Ultimately, MoCab aligns seamlessly with clinical workflows and data integration requirements, enhancing the effectiveness of predictive models to support clinical decision making and ultimately improving patient care outcomes.
II. RELATED WORKS

In recent years, tools and platforms have been developed to harness the potential of predictive modeling in healthcare. EPOCH® and ePRISM® are examples of such web-based platforms designed to assist healthcare professionals in creating and utilizing predictive models [13]. ePRISM®, a versatile regression model framework, integrates prognostic models, but faces challenges in EHR integration due to its reliance on manual data entry, a time-consuming and complex process that hampers seamless integration of clinical workflows.

The FHIR standard has emerged as a solution that promotes data exchange between HISs, facilitating interoperability [14], [15]. The adaptability of FHIR not only has simplified data access and sharing for healthcare providers, but has also catalyzed the development of new healthcare applications and services [16], [17]. Platforms such as KETOS utilize FHIR to allow researchers to conduct statistical analyses and develop, train, and deploy machine learning models in a secure environment [18].

Despite these advances, integrating multiple predictive models into HISs introduces challenges such as "app fatigue" among physicians due to the abundance of available models and applications [19]. With a growing number of models and applications, healthcare providers face the difficult task of discerning the optimal timing and context of their use. This challenge is further exacerbated by the diverse functionalities offered by platforms such as KETOS.

CDS Hooks and SMART on FHIR address these issues. CDS Hooks provide advanced alerting and clinical decision support by analyzing patient data for specific risk factors, ensuring timely recommendations [20]. SMART on FHIR offers an interoperable app platform for EHRs, facilitating standardized patient data access across different FHIR-based EHR systems [21].

The standard FHIR API, while effective, struggles with the voluminous data demands of comprehensive model training [22]. The bulk FHIR service mitigates this by allowing efficient retrieval of large datasets from FHIR servers, crucial to the development and deployment of machine learning models within EHRs [9].

III. SYSTEM ARCHITECTURE

The overall architecture of MoCab described in fig. 1, performs primarily three functions. First, the Data Service Center is responsible for the retrieval of patient data from FHIR server, ensuring that the appropriate information is accessed and prepared for analysis. Second, these data are then transferred to the predictive model in the Knowledge Model Center. Moreover, the architecture includes a Model Retraining Center, ensuring that the models are regularly updated and refined over time, thus maintaining their accuracy in ever-evolving clinical settings. In addition to these core functionalities, MoCab integrates both CDS Hooks and SMART on FHIR endpoints. This integration is to improve clinical decision-making processes. CDS Hooks serve a dual purpose, proactively alerting healthcare professionals to potential risks and allowing real-time tracking of health indicators. The source codes of MoCab are available at GitHub [23].

A. Data Service Center

The Data Service Center comprises two main services: the Data Retrieval Service (DRS) and the Data Extraction Service (DES) (fig. 1, black lines).

The DRS is responsible for fetching data from the FHIR server with the REST API. Each feature required by the predictive model is listed in a "feature table" (table I), including specific codes and the eligible time frame for the records. For example, hypertension configuration uses code 110 in the ICD-10 code system, and to extract data covering the last seven years, using a designated data_alive_time setting "0007-00-00T00:00:00". The DRS has the ability to handle multiple coding scenarios for a single feature. Furthermore, when multiple records are available, the DRS uses specified search criteria (e.g., latest, maximum, or minimum), defined in search_type, to select the most relevant record. For example, in table I, LOINC 8310-5 and 8331-1 both refer to body temperature, and the latest prefix enables DRS to obtain the closest data. In cases where the data do not match the query, the service returns a default value in default_value or an API error, depending on the configuration in the feature table.

Post-retrieval, the DES takes over and extracts data from the fetched resources to fit the format conducive to model input. DES extracts specific values and associated timestamps from these resources, tailoring the data extraction to the nature of the resource. For example, observation resources yield laboratory results and observation dates, whereas condition and procedure resources are processed into Boolean values and corresponding dates, representing whether the server returns resources, and the date is the time of the latest recorded data. In addition, patient resources are utilized to calculate the patient's age.

Features may be stored in different columns in the resource. For example, the margin of surgery is stored in the "bodysite" of procedure resources based on the definition of the Taiwan Cancer Registry team [24]. The DES can be further augmented by a resource route table (table II), which delineates the extraction path for data stored in nonstandard columns within a resource. This ensures accurate and efficient data extraction, even for complex data structures. Finally, after extracting all the necessary features, the extracted data can be packed and sent to the Knowledge Model Center.

B. Knowledge Model Center

Upon retrieval and extraction of the data by the Data Service Center, MoCab directs these data to the Knowledge Model Center. Before importing data into models, we need to adapt the data to formats compatible with different models. For example, some models accept numeric inputs, such as white blood cell count, while others require nominal data, such as an indicator of overweighted or underweighted, or hypothermia in patients. In addition, certain features, such as BMI, require further calculations based on the original records. To efficiently address these varying input format requirements, MoCab incorporates a Prediction Adapter module, which converts features into the model-specific preferred input format. A "transformation table" (table III) is used to outline the data
Fig. 1. MoCab Architecture. The black lines represent the process of applying the model using patient identification to retrieve data from FHIR server, while the red lines indicate the process with data input by users or other applications. The green lines represent the continuous training mechanism. The yellow lines highlight the retrieval process of historical patient data for the SMART on FHIR App. The blue lines delineate the SMART authentication process, which is critical for secure data access. The blue keys represent the SMART OAuth token, which is utilized to protect access to the FHIR server.

<table>
<thead>
<tr>
<th>Model</th>
<th>Feature</th>
<th>Code</th>
<th>Code system</th>
<th>Type of data</th>
<th>Data alive time</th>
<th>Default value</th>
<th>Search type</th>
<th>Value route</th>
<th>Datetime route</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>body-temp</td>
<td>8310-5</td>
<td><a href="http://loinc.org">http://loinc.org</a></td>
<td>observation</td>
<td>0007-00-00T00:00:00</td>
<td>36</td>
<td>latest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>body-temp</td>
<td>8331-1</td>
<td><a href="http://loinc.org">http://loinc.org</a></td>
<td>observation</td>
<td>0007-00-00T00:00:00</td>
<td>36</td>
<td>latest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>hypertension</td>
<td>I10</td>
<td><a href="https://icd.who.int/browse10/2019/en">https://icd.who.int/browse10/2019/en</a></td>
<td>condition</td>
<td>0007-00-00T00:00:00</td>
<td>latest</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>margin_of_surgery</td>
<td>90</td>
<td><a href="https://mitw.dicom.org.tw/IG/TWCR_LF">https://mitw.dicom.org.tw/IG/TWCR_LF</a></td>
<td>procedure</td>
<td>0007-00-00T00:00:00</td>
<td>latest</td>
<td>margin_of_surgery</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>conditions</th>
<th>methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>margin_of_surgery</td>
<td>Procedure.bodySite.0.coding.0.code</td>
</tr>
<tr>
<td>gender</td>
<td>Patient.gender</td>
</tr>
<tr>
<td>age</td>
<td>Patient.get_age()</td>
</tr>
<tr>
<td>encounter_type</td>
<td>Encounter.class.code</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>model</th>
<th>feature</th>
<th>type</th>
<th>formulate</th>
<th>index</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHARM</td>
<td>Hypothermia</td>
<td>category</td>
<td>1=lt 36</td>
<td>1</td>
</tr>
<tr>
<td>CHARM</td>
<td>Hypothermia</td>
<td>category</td>
<td>0=ge 36</td>
<td>1</td>
</tr>
<tr>
<td>SPC</td>
<td>BMI_label</td>
<td>category</td>
<td>1=BMI lt 18.5</td>
<td>1</td>
</tr>
<tr>
<td>SPC</td>
<td>BMI_label</td>
<td>category</td>
<td>2=BMI ge 18.5 &amp; lt 24</td>
<td>1</td>
</tr>
<tr>
<td>SPC</td>
<td>BMI_label</td>
<td>category</td>
<td>3=BMI ge 24</td>
<td>1</td>
</tr>
<tr>
<td>SPC</td>
<td>BMI</td>
<td>formula</td>
<td>[weight]/([height]/100)</td>
<td>1</td>
</tr>
</tbody>
</table>

transformation configurations, enhancing the adaptability of the system to various model requirements.

Within MoCab, features are categorized into three types: numeric, categorical, and formulaic. Numeric features are directly fed into the model without transformation. On the contrary, categorical and formulaic types undergo a transformation based on user-defined rules. The formulaic type facilitates operations on one or more features to derive a new feature. Subsequently, the transformed data are compiled into an array ready for model prediction, with each feature indexed according to the transformation table (table III).

The Prediction Adapter extends its capabilities to handle pre-processed records (fig. 1, red lines). It verifies the completeness of these records before they are transmitted to the requested predictive model, thus ensuring the integrity and applicability of the data used in models stored in the Model Cabinet.

Model Cabinet is a dedicated repository for storing the
models along with their respective dependencies. Each model is housed in its folder, which contains a 'predict' function. This function is integral to the prediction process, where it receives input from the Prediction Adapter, makes the prediction, and relays the prediction result back to MoCab. The results, along with the processed data, are then sent to the end-user through an API.

To facilitate model integration, MoCab provides a template compatible with models exported from frameworks such as Joblib or TensorFlow. This template enables for seamless model replacement and updating; it requires only the substitution of the trained model and the updating of the model's name to 'register_model'. In addition, necessary data configurations, including features and transformation tables, are integrated, making the model ready for prediction.

C. Model Retraining Center

The concept of continuous training is crucial in the realm of machine learning, particularly in maintaining the accuracy of models in dynamic environments. This process involves periodically updating the model using transfer learning with new data to account for possible changes in the underlying data patterns or operational context. The MoCab Model Retraining Center synthesizes the retraining process into six modules: Scheduler, Data Retrieval Parser, Data Transformation Bundler, Model Trainer, Model Evaluator, and Model Register. This approach ensures that the models integrated within MoCab remain current and accurate, thereby providing reliable predictions in a clinical setting.

1) Scheduler: The fundamental mechanism for configuring this Scheduler is the training parameter table. By specifying the time interval for retraining in the training parameter table, the Scheduler is programmed to automatically initiate the continuous training pipeline at predetermined time intervals.

2) Data Retrieval Parser: The Data Retrieval Parser extracts data from the FHIR server and parses them into a format suitable for model training. To facilitate the retrieval of large datasets, this parser uses the FHIR bulk service. Upon issuing a 'kick-off' request, the FHIR bulk service returns data in Newline Delimited JSON (NJSON) format, which is compliant with the FHIR specification. The subsequent stage involves data retrieval and extraction from these resources. The configuration of required features is established in the Data Service Center. This feature table allows MoCab to identify relevant features or prediction targets by matching them with corresponding codes and code systems via the DRS. The DES then extracts specific values from these resources. Temporal alignment of the data is also ensured by further configuration. For example, in a model designed to predict hypertension, it is imperative to use the data recorded before the appearance of hypertension. MoCab addresses this by setting appropriate time ranges between features and prediction targets in the 'filter' field of the training parameter table. This approach to temporal data alignment is vital to the rationality of the training process.

3) Data Transformation Bundler: The Data Transformation Bundler undertakes the transformation of input features and targets into appropriate formats and bundling of these data into training and testing sets. Continuing the transformation steps in the Knowledge Model Center, the Data Transformation Bundler utilizes the Prediction Adapter to transform the data. This process is governed by the specifications described in both the transformation table and a corresponding table for the targets, which allows the integration of patient data into a cohesive dataset.

The next step is to exclude the patient who is not the study target. For example, as demonstrated in the second primary cancer prediction model, patients who did not experience lung cancer as their first primary cancer were deemed non-target for the model and subsequently excluded. For records with missing values, developers can specify the threshold for exclusion or select appropriate imputation methods. The resultant training data is then funneled into the Model Trainer, while the testing data is reserved for the Model Evaluator. An additional preparatory step, particularly for models dealing with categorical variables, involves encoding these variables into a binary format (dummy or indicator variables). This encoding process is executed before the distribution of datasets to the subsequent modules.

4) Model Trainer: The Model Trainer module is responsible for fine-tuning machine learning models with newly prepared data. This process begins with the retrieval of the existing model from its respective folder, followed by the integration of the new training data. The model is then fine-tuned with transfer learning approach, a process facilitated by the training function embedded within each model's directory. To streamline the training process, MoCab offers a specialized continuous training template, simplifying the training procedure by allowing users to replace the training compiler. Once training is completed, the updated model is saved under the designation "new_model" for further evaluation in the Model Evaluator module.

5) Model Evaluator: Recognizing that a fine-tuned model may not necessarily outperform its predecessor, MoCab employs an evaluation process using the test data set provided by the Data Transformation Bundler. This evaluation process focuses on the computation of various metrics to determine the performance of both existing (old) and newly fine-tuned (new) models. Key metrics such as accuracy, F1 score, and the area under the receiver operating characteristic curve (AUROC) are used for comparison. The selection of the most effective model is based on the validation_index specified in the training parameter table.

6) Model Register: This final step involves integrating the model that demonstrated higher performance during the eval-
TABLE IV
EXAMPLE OF THE TRAINING PARAMETER TABLE

<table>
<thead>
<tr>
<th>model</th>
<th>Filter</th>
<th>Interval</th>
<th>Null value strategy</th>
<th>Test size</th>
<th>Validation index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(date)lt[hypertension]</td>
<td>0001-00-00T00:00:00</td>
<td>(drop)gt[3&amp;(mean)]</td>
<td>0.2</td>
<td>accuracy</td>
</tr>
<tr>
<td>2</td>
<td>(date)lt[target_1]</td>
<td>0000-06-06T00:00:00</td>
<td>(drop)gt[5&amp;(median)][feature_1]&amp; (mode)</td>
<td>0.2</td>
<td>auroc</td>
</tr>
</tbody>
</table>

ulation phase into the Model Cabinet (fig. 1, green line linked to the Model Register). The Model Register also documents model metadata for the model, the number of patients involved in the training set, the timestamps of the last training and data acquisition, and the evaluation results (table V). These records provide information on the training history, facilitating transparency and traceability of model development.

IV. INTEGRATION WITH CLINICAL DECISION SUPPORT SYSTEM

MoCab offers three interaction modalities. For entities that utilize open FHIR endpoints or their own methods for data retrieval, the MoCab regular endpoint (fig. 1, Regular Endpoint) is tailored to their needs. To combat app fatigue caused by the proliferation of CDS services, MoCab incorporates a strategic approach with CDS Hooks, which is designed to generate predictions for at-risk patients and guide the delivery of appropriate alerts to physicians (fig. 1, CDS Hooks Endpoint) [19]. We also provide an option to activate MoCab from SMART on FHIR App (fig. 1, SMART Endpoint).

Each model within MoCab caters to specific use cases. The CDS Hooks table specifies the contextual usage of each model (table VI). For example, the CHARM model [30] is designed primarily to predict mortality among patients with suspected sepsis for emergency department applications, and an emergency department encounter in FHIR is coded as “EMER” [31]. Accordingly, the CDS Hooks table can define a triggering condition as “encounter_type=EMER”, indicating that MoCab will activate the model for records originating from emergency departments. The sources of conditions are in the resource route table (table II). For instance, “encounter_type” is mapped to ”class.code” in the FHIR Encounter resources. MoCab extracts this information to determine the applicability of a model to a specific patient scenario. Additionally, for conditions requiring computations, such as patient’s age, the necessary function is specified in the resource route table (table II, ”age” condition).

A notable feature of CDS Hooks within MoCab is their ability to deliver customized alerts to physicians. CDS hooks generate CDS cards, which encompass a range of information, from basic text notifications (information card) and alternative action suggestions (suggestion card) to links that lead to applications or reference materials (app link card). The cards are categorized into three indicators – info, warning, and critical – each signifying the level of urgency or importance of the information presented.

In MoCab, the CDS Hooks endpoint is configured to return different cards based on the risk thresholds defined for each model (table VI). These cards not only display the model’s risk score and descriptive results but also include a SMART app link. Activating this link (fig. 1, black line from EMR client to SMART Endpoint) initiates a request for SMART on FHIR authentication from the EHR server to secure access permission, enabling the MoCab SMART app to retrieve targeted patient data from the FHIR server.

The MoCab SMART app integrates the core functions of MoCab, such as risk calculation, with additional features. A significant feature is to visualize historical patient data through trend charts, accessible through the SMART Endpoint (fig. 1, orange line). These charts provide physicians with a view of patient health over time, improving the depth and quality of clinical decision-making.

V. IMPLEMENTATION AND DEMONSTRATION

To demonstrate the practicality of MoCab, we implemented the framework using three different models: a scoring model (the quick COVID severity index [32]), a machine learning model (focusing on necrotizing soft tissue infections (NSTI) [33]), and a deep learning model to predict second primary cancer (SPC) in patients with lung cancer [28]). To simulate realistic data scenarios, we generated synthetic data that are consistent with the structure of one of the largest EHR database, the Chang Gung Research Database (CGRD) [34], supplemented by examples from the official FHIR documentation [35]. This section delineates the implementation steps, including the generation of FHIR resources, the deployment of models, and interactions with MoCab to obtain prediction results. The source codes of the models are available on GitHub [36].

A. Data and FHIR Server

Recognizing the growing prominence of FHIR in the landscape of EHR [37], we adopted FHIR version R4 as the standard protocol for data retrieval to create a system compatible with a variety of EHRs. We use HAPI FHIR [38], version 6.4.0, for its robust support of FHIR bulk services.

Data used in this study was simulated based on CGRD [34], then converted into FHIR format with the FHIR Extract Transformation Load (ETL) tool [39]. This process was guided by the official FHIR documentation [40] and was supplemented with information from the FHIR Implementation Guide (IG) documents [41]. The Chang Gung Medical Foundation Institutional Review Board approved this study (IRB no. 201901386B0) and waived the requirement of patient consent.

B. Quick COVID Severity Index (qCSI)

The Quick COVID Severity Index (qCSI) is a scoring model that helps predict the 24-hour risk of critical respiratory illness in patients with COVID-19 admitted to emergency
TABLE V
EXAMPLE OF THE TRAINING STATUS TABLE

<table>
<thead>
<tr>
<th>Model</th>
<th>Last training time</th>
<th>Last training data time</th>
<th>Numbers of patients</th>
<th>Old model evaluate</th>
<th>New model evaluate</th>
<th>Register model</th>
<th>Threshold</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2022-04-10</td>
<td>2018-11-01</td>
<td>481</td>
<td>0.683</td>
<td>0.741</td>
<td>register</td>
<td>0.125</td>
</tr>
</tbody>
</table>

TABLE VI
EXAMPLE OF THE CDS Hooks Table

<table>
<thead>
<tr>
<th>Model</th>
<th>Condition</th>
<th>Into range</th>
<th>Warning range</th>
<th>Critical range</th>
</tr>
</thead>
<tbody>
<tr>
<td>qCSI</td>
<td>encounter_type=EMER</td>
<td>lt/0.3</td>
<td>ge/0.3 &amp; lt/0.7</td>
<td>ge/0.7</td>
</tr>
<tr>
<td>NSTI</td>
<td>encounter_type=EMER</td>
<td>lt/0.3</td>
<td>ge/0.3 &amp; lt/0.7</td>
<td>ge/0.7</td>
</tr>
</tbody>
</table>

departments [32]. qCSI operates by evaluating three parameters: respiratory rate, $SP_{O_2}$ (oxygen saturation), and $O_2$ flow rate. The model translates these numeric values using a point system, where the cumulative score on these characteristics estimates the risk of critical respiratory illness. This scoring mechanism helps physicians make informed clinical decisions.

For implementation, we refer to the format of the vital sign profile on the official FHIR website [40]. Subsequently, we extracted vital signs from the CGRD, converting this information into FHIR resources. It is important to note that the $O_2$ flow rate data, originally recorded in plain text, were mapped to the valueString field in FHIR. Details of this data mapping are given in table VII.

The feature retrieval process is based on the feature table [42], which specifies the CGMH code systems for respiratory rate and $SP_{O_2}$, along with LOINC terminology codes for $O_2$ flow rate. All data were used within a window. Subsequently, we defined the data transformation configurations [43] based on the variables of the qCSI scoring model [32]. In this phase, the Prediction Adapter converts numeric values into corresponding qCSI points according to the scoring model’s criteria. The culmination of this process involves the summation of points from each feature, upon which MoCab returns the final qCSI score. An additional challenge was handling the $O_2$ flow rate data, which was stored as plain text on the FHIR server. To address this, we implemented regular expression parsing to extract specific values from the text.

C. Necrotizing Soft Tissue Infection Model (NSTI)

Necrotizing fasciitis, a rare and critical infection that affects soft tissues, often requires immediate treatment to prevent fatal outcomes [44]. We included an NSTI model, built with logistic regression [33], to differentiate necrotizing fasciitis from other soft tissue infections. The NSTI model requires input on five features for analysis: seawater exposure and four laboratory results: white blood cell count, C-reactive protein, creatinine, and sodium (table VIII). However, the CGRD does not include records of seawater exposure. To address this, physicians can manually indicate seawater exposure in the MoCab SMART App.

The feature table presents the data retrieval configurations for the NSTI model [42]. The transformation table delineates the complete data transformation configuration [43]. Subsequent to the data transformation stage, these features are sent to the finalized model imported in Joblib format in Knowledge Model Center.

D. Second Primary Cancer (SPC)

SPC arises when an individual develops a new primary cancer following an earlier primary occurrence [45]. Factors such as specific cancer treatments, genetic predispositions, and exposure to cancer-causing substances significantly increase the risk of SPC [46], [47]. Lung cancer, which is notably prevalent and widespread worldwide, often requires careful monitoring of SPC occurrences [48], [49]. We trained an SPC prediction model for lung cancer survivors using TensorFlow, utilizing the data generated based on the cancer registry dataset in CGRD [28]. The model, which requires 27 features, integrates multiple FHIR resources, including patient, encounter, condition, observation, and procedure.

To create data in FHIR format, we applied the guidelines of the Taiwan Cancer Registry Center (TWCR) [24], the minimal Common Oncology Data Elements (mCODE) [50], and insights from the Taiwan ‘cancer2fhir’ team [51]. This approach facilitated conversion from the cancer registry dataset. Details of data retrieval and transformation configurations are provided on GitHub [42], [43].

For continuous training, we set specific parameters: 1) Features needed to be time-stamped between the first and second primary cancers (table IX, Filter). 2) Data with more than three missing values were excluded. 3) Missing values were imputed with the median (table IX, Null value strategy). 4) 20% of the data was selected as testing set. 5) The AUROC was used for validation (table IX, Test size and Validation index). The feature table and the transformation table for the target configuration are available on GitHub [52], and users can adjust these settings by modifying the configuration tables. The last training data interval was initialized as 2018-01-01 (table X, first row). After continuous training, the training status table was updated to reflect the latest training results (table X, second row).

E. Integrations with CDS Hooks and SMART on FHIR

To simulate the interaction between HIS and the MoCab framework, experiments were carried out using CDS Hooks and SMART on FHIR Sandbox [53]. The configuration of the CDS Hooks is in table VI. This setup includes predefined warning ranges for each model, enabling the system to issue appropriate alerts. For example, in the case of the qCSI model, the system is configured to present an information card for scores less than 4, a warning card for scores between 4 and 9, and a critical alert for scores exceeding 9 points (table VI, qCSI). For the NSTI and qCSI models, designed primarily for emergency department use, the models are automatically activated upon the onset of an emergency department encounter.
In preparation for this, we identified the required condition routes, as specified in the CDS Hooks scenarios. These routes were then documented in the resource route table [54]. Taking the qCSI model as an example, the patient’s encounter resource was retrieved in accordance with the model requirements and the definition of the encounter_type condition [54]. Upon verification that the type of patient encounter matched the criteria, MoCab initiated the qCSI model prediction process. This simulation demonstrated MoCab’s capability to accurately trigger and process model predictions based on specific criteria, thereby validating its practical utility.

One of the key contributions of MoCab is the creation of an intuitive SMART on FHIR application, designed to facilitate the interaction between physicians and the model in MoCab. The application is accessible through the ‘MoCab-App’ button on the CDS Hook card. Once activated, it presents a user interface that displays key parameters involved in the predictive analysis of the model (fig. 2). In addition, this interface offers a comprehensive overview of the observational data of a patient and trend analysis, significantly helping physicians in their decision-making process. In particular, it allows for real-time adjustments to clinical features, enabling physicians to view updated prediction results, and empowering physicians by providing greater control and flexibility. Furthermore, the application is enhanced with additional functionalities like timeline filtering. These features are not only user-centric but also provide valuable information, thus increasing the overall utility of the MoCab system in clinical settings.

### VI. Discussion and Conclusion

MoCab uses the FHIR standard to streamline the deployment and management of various models in various EHR systems. It offers numerous advantages, particularly in reducing the costs associated with the deployment of prediction models, supporting continuous training to improve the performance of models over time, and improving decision-making processes. MoCab integrates seamlessly with CDS Hooks and SMART...
on FHIR, providing timely alerts and decision-making support to healthcare professionals.

The MoCab system exhibits similarities with previous studies while simultaneously introducing key enhancements. Similarly to the EPOCH® and ePRISM® systems [13], MoCab aims to integrate machine learning models into healthcare systems. Furthermore, MoCab is consistent with the objectives of KETOS [55], which offers a solution seamlessly integrated into healthcare systems. However, unlike EPOCH® and ePRISM®, which require manual input of model parameters, MoCab capitalizes on the rapidly evolving FHIR standard. This utilization facilitates automatic data retrieval from EHR systems, thus obviating the need for manual parameter input. Moreover, while KETOS is geared toward providing a platform for model development and lacks mechanisms for active alerts to healthcare professionals, MoCab incorporates CDS Hooks. This integration enables the delivery of varying levels of alerts to healthcare professionals. Additionally, MoCab has the capacity for continuous learning. This feature allows the model to continuously train and refine its predictive accuracy.

MoCab exhibits certain constraints, such as the unavailability of specific features on the FHIR server. To address this, MoCab is designed to accept pre-processed records, allowing data gathered from users via the MoCab SMART App. Furthermore, MoCab has adopted the FHIR standard in conjunction with the FHIR server. Transformation of EHR data to comply with FHIR standards might present difficulties in some healthcare institutions. However, given the growing adoption of FHIR globally [56], we anticipate a reduction in this challenge, increasing MoCab’s utility as FHIR becomes more prevalent in healthcare facilities. Finally, in the current framework, the performance comparison between newly trained and pre-existing models is contingent upon performance metrics selected by the users. This approach, while practical, may introduce biases as it relies on a singular data set for evaluation. To mitigate potential biases and enhance the reliability of our model comparison, incorporating methodologies such as cross-validation and statistical tests is a potential solution.

In the future, MoCab aims to expand its functionality by developing a user-friendly model import interface and providing customization options for encoding methods [57], [58]. In addition, the incorporation of Clinical Quality Language (CQL) [59] is anticipated, which would further improve model integration capabilities.

In summary, MoCab represents a significant advancement in model reusability and optimization of machine learning models within healthcare settings. Its potential to support clinical decision making and improve the performance of imported models underscores its value in the evolving landscape of healthcare informatics.

REFERENCES


[31] C.-Y. Wu, Analysis of infection risk in necrotizing soft tissue of the extremities using machine learning tech...


