IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 22 May 2024, accepted 10 June 2024, date of publication 13 June 2024, date of current version 20 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3414184

RESEARCH ARTICLE

Smart Summary: A Distributed Medical Recommender System for Patients in the ICU Using Neural Networks

AHMAD AYAD[1], (Member, IEEE), YU-HSUAN TAI[1], GUIDO DARTMANN[2], (Senior Member, IEEE), AND ANKE SCHMEINK[1], (Senior Member, IEEE)

[1]Chair of Information Theory and Data Analytics (INDA), RWTH Aachen University, 52056 Aachen, Germany
[2]Chair of Distributed Systems and Artificial Intelligence, Umwelt-Campus Birkenfeld, 55761 Birkenfeld, Germany

Corresponding author: Ahmad Ayad (ahmad.ayad@inda.rwth-aachen.de)

This work was supported by the Federal Ministry of Education and Research (BMBF, Germany) as part of NeuroSys: Impulse durch Anwendungen (Projekt D) under Grant 03ZU1106DA.

ABSTRACT In the medical domain, particularly in intensive care units (ICUs), the immense volume of patient data presents a significant challenge for clinicians, often resulting in the oversight of critical information or excessive time consumption in accessing it. Recommender systems have been introduced to facilitate targeted, data-driven decision-making and ease the burden on healthcare professionals. This paper introduces Smart Summary, a novel distributed medical recommender system aimed at streamlining the analysis of extensive patient data and improving diagnostic accuracy by focusing on essential information. Smart Summary leverages patients' admission reports and past lab values to predict International Classification of Diseases (ICD) codes, extract disease names, and forecast future abnormalities in lab values. Using this information, Smart Summary builds a comprehensive patient profile that covers the patient's case precisely. Additionally, it recommends the most relevant laboratory values for individual patients by analyzing their data through various modules, including lab values abnormality prediction, automatic ICD codes prediction, and disease-named entity recognition. Furthermore, Smart Summary enhances its performance by incorporating doctors' feedback, utilizing this information to refine recommendations for patients with similar profiles within the same cluster. Experimental results demonstrate that Smart Summary effectively learns to recommend relevant lab values for patients, achieving a Precision@10 of 0.92 after training on doctors' feedback. Moreover, Smart Summary employs an efficient distributed machine learning method based on a split learning mechanism to ensure patient data privacy. This mechanism not only guarantees data privacy and security but also reduces communication overhead by 72% and computation overhead by 45.2% compared to the original split learning mechanism. To our knowledge, Smart Summary is the only system that creates comprehensive patient profiles using multiple machine learning models and recommends relevant lab values while ensuring privacy, efficiency, and security across various data sources.

INDEX TERMS Distributed machine learning, green AI, ICU, medical informatics, recommender systems, split learning.

I. INTRODUCTION

Information overload refers to the phenomenon of having too much information to process or make sense of [1]. In the field of medicine, data overload is a major problem that can negatively impact patient care and healthcare delivery. The amount of data available to healthcare professionals, including patient medical records, lab results, and imaging studies, is growing rapidly. This makes it challenging for doctors, nurses, and other healthcare providers to access and

The associate editor coordinating the review of this manuscript and approving it for publication was Dominik Strzalka.

comprehend relevant information and identify patterns and trends that could be used to improve patient outcomes and healthcare [2]. Additionally, data overload can lead to delays in diagnosis and treatment and an increased risk of medical errors. It also makes it difficult for healthcare professionals to stay up-to-date with the latest medical knowledge and guidelines.

To address the problem of data overload in medicine, healthcare organizations are turning to data management and analytics solutions to better organize, access, and interpret their data. Additionally, the use of recommendation systems, natural language processing, and other technologies can help healthcare professionals easily access and interpret the information they need [3].

Recommender systems are becoming increasingly important in managing medical information, as they can help doctors and patients easily access relevant and up-to-date information. These systems use data mining and machine learning techniques to analyze large amounts of data, such as electronic health records (EHRs), and make personalized recommendations based on patient's individual needs and preferences. This can greatly improve the efficiency and effectiveness of medical care, as well as increase patient satisfaction. Moreover, recommender systems can also help with decision-making by providing healthcare professionals with evidence-based recommendations. Overall, the implementation of recommender systems in managing medical information can greatly benefit doctors and patients [4].

A. RELATED WORK

Since the rapid increase in the availability of medical data for training machine learning models, there has been a lot of work on using machine learning-based recommender systems in medicine [5]. For example, the authors in [6] propose a decision support system that reads the patient's medical records and processes them using a sliding-window-based time-series prediction algorithm to predict short-term risk for heart failure. According to the risk, the system will recommend whether the patient needs a certain medical test, like a heart rate test. Similarly, the authors in [7] use an autoencoder-based time-series prediction algorithm to recommend physical activity for senior adults.

Moreover, the authors in [8] have developed a recommender system based on random forests to classify diseases according to symptoms and recommend a list of relevant precautions for that disease accordingly. Similarly, the authors in [9] use singular value decomposition and decision trees to recommend suitable doctors for patients.

Additionally, the authors in [10] proposed a hybrid recommender system framework that combines artificial neural networks and case-based reasoning to support general practitioners (GPs) in personalized clinical prescriptions. The system creates a patient feature based on demographic information, lab test results, and free text. Then, the system clusters drugs by a k-means algorithm based on the frequency of concurrence with symptom features. The system uses multi-label classification to predict the chosen drug cluster based on the patient feature and then rank the drugs within the cluster using case-based ranking.

TABLE 1. Comparison between recommender systems in the medical domain. CF: collaborative filtering. SVD: singular value decomposition. NN: neural network.

Approach	Model Type	Distributed	Recommended Items	Ref.
Time-series forecasting	Moving window	No	Medical advice	[6]
	Autoencoder	No	Physical activity type	[7]
Decision trees	Random forest	No	Diseases, Precautions	[8]
	SVD + Decision tree	No	Doctors	[9]
Clustering	NN + K-means	No	Drugs	[10]
Filtering based	CF + Blockchain	Yes	Treatments	[12]
	NN + Federated learning	Yes	Drugs	[13]
Smart summary	NN + K-means + Split learning	Yes	Lab values	

All of the aforementioned systems are centralized systems that work on big datasets with all the patient's information. However, the need for distributed machine learning (DML) systems has recently increased because they allow sensitive medical data to be processed on local machines, where the data often originates rather than being sent to a central server. This can help protect patient privacy and maintain medical data security [11]. Additionally, real-time data processing is often required in medical settings, such as in an intensive care unit (ICU). DML allows for models to be deployed on edge devices, such as in hospitals or clinics, enabling real-time decision-making and scalability [11]. For example, the authors in [12] proposed a system named HealthMudra to recommend treatments for patients with diabetes. In this work, a filtering-based recommendation system is proposed to prevent diabetes. To achieve this goal, a decentralized database utilizing Blockchain technology is employed to store many doctor-generated recommendations for reducing symptoms of diabetes. Furthermore, the authors in [13] use federated learning to recommend drugs to doctors in a distributed and private way. Finally, Table 1 summarizes the reviewed literature and how they compare to our proposed approach.

This work investigates the diagnosis and treatment of critically ill patients in the intensive care unit (ICU). Following our previous work in [14] and [15], we focus on mechanically ventilated patients. The motivation behind this work lies in the significant amount of laboratory data that is regularly collected during the treatment of these patients. However, due to the large number of values that need monitoring in the ICU, which can sometimes exceed 100 lab tests [16], important anomalies, trends, or relevant

values may not be identified. Especially since many of the lab tests ordered at the ICU might not be essential for a specific patient [17]. Identifying relevant lab values can aid in resource allocation and save lives by enabling timely intervention. Furthermore, healthcare workers spend $30\% - 50\%$ of their time in front of computers and must deal with a vast amount of patient data [15], [18]. Any savings in that time can allow them to spend more time with patients. Therefore, we developed a distributed recommender system that analyzes the patient's historical data collected in the EHR during their stay and recommends the most relevant lab test results (lab values) for that patient to healthcare professionals, highlighting which lab values are predicted to be abnormal in the next time that lab test is done. Additionally, the system analyzes the admission reports. It automatically predicts ICD codes and extracts disease names to make them easily available for doctors to analyze and use to group similar patients. The system can also receive feedback from healthcare professionals on its recommendations and learn to give better ones in the future for similar patients in all of the hospitals participating in our distributed system without sharing the patients' data. Our DRS can support healthcare professionals in making more efficient, comprehensive, and better diagnoses, which could further enhance healthcare quality.

Our overall system is shown in Fig. 1. The patients' data is collected and saved in the EHRs. The different modules of our system process this data to generate the patients' profiles and cluster similar patients according to these profiles. Then, the system will record the doctors' interactions and save them locally in the feedback dataset (the user-item matrix). Additionally, the deep neural network, which is the core of our DRS, that trains on the feedback dataset, is split into two parts following the split learning architecture [19]: A client part that learns from the local user-item matrix and a server part that uses the output of the client model to continue the training on the rest of the model. The client model is the same for all the hospitals/clinics, where each model trains on local data. This ensures that we only share the output of the client models and not the raw input data, leading to secure and private learning.

B. OUR CONTRIBUTIONS

The contributions of our work are as follows:

- We integrate our lab values abnormality prediction module introduced in our previous work [15]. The system reads the patients' lab values and predicts which lab values will be abnormal shortly.
- We developed a model for ICD code prediction based on a Multi-CNN architecture and the attention mechanism that outperformed similar models in this task. The model outputs a list of predicted ICD codes for a specific patient.
- We used the BioBERT model to analyze admission reports, extract disease names to be shown to doctors, and use in the patients' profiles.
- We propose a novel DRS architecture that combines the output of the aforementioned modules by stacking them to build a patient profile.
- We use the K-means algorithm to group patients' profiles. The patient clusters are then utilized to recommend relevant lab values to similar patients within the same cluster.
- We tested the ability of our DRS to learn from doctors' feedback by creating testing scenarios that include a variable number of patient groups.
- We utilized our innovative modified split-learning approach, as presented in our prior work [20], to revamp the DRS architecture. This ensured data privacy for all participating hospitals while significantly reducing communication and computation overhead during the learning phase.
- We investigated the explainability aspect of our system to confirm the usability of our DRS by doctors.

C. PAPER ORGANIZATION

The remainder of this paper is organized as follows. In Section II, we explain our proposed system architecture and its sub-systems. In Section III, we cover the experimental setup and discuss the overall system results and performance as well as the explainability aspect of the system. Finally, we conclude our work in Section IV.

II. SYSTEM MODEL

In Fig. 2, the overall framework of our recommender system is shown. The framework shows, from a single local user's perspective, the end-to-end recommendation system, including the sub-modules A, B, and C that are used to construct the patient's profile which is used then by the recommendation system (sub-module D) to generate the patient's smart summary which includes the relevant lab values list alongside the predicted ICD codes and the extracted disease names. The framework begins by reading the following patients' data from the EHR:

1) Lab values: these are the results of the lab tests that are done during the patient's stay at the ICU. Because the lab values are irregular in terms of frequency, additional preprocessing was required to deal with them. We have chosen the most frequent 25 lab values from the MIMIC dataset (Sodium, Potassium, white blood cell count, etc…) as our main set of lab values. The full list of chosen lab values is described in our previous work [15].
2) Demographics: these include the age, gender, and weight of the patient. The statistical properties of these chosen features are also mentioned in our previous work [14].
3) Admission notes: a patient's status is routinely recorded in a continual document, which wraps up with a discharge report. Since we want our recommendation system to start working at the admission time, we had to take some parts from the discharge report that are

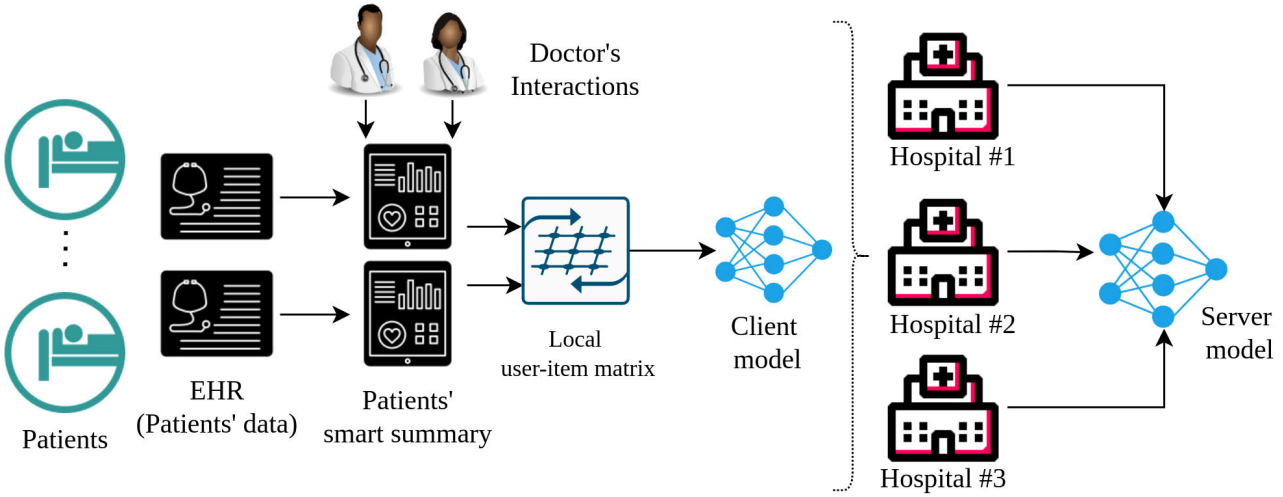

FIGURE 1. Illustration of our proposed DRS. The system processes the patient data to generate the patient profile. The profile is used then by the DRS to recommend relevant lab values.

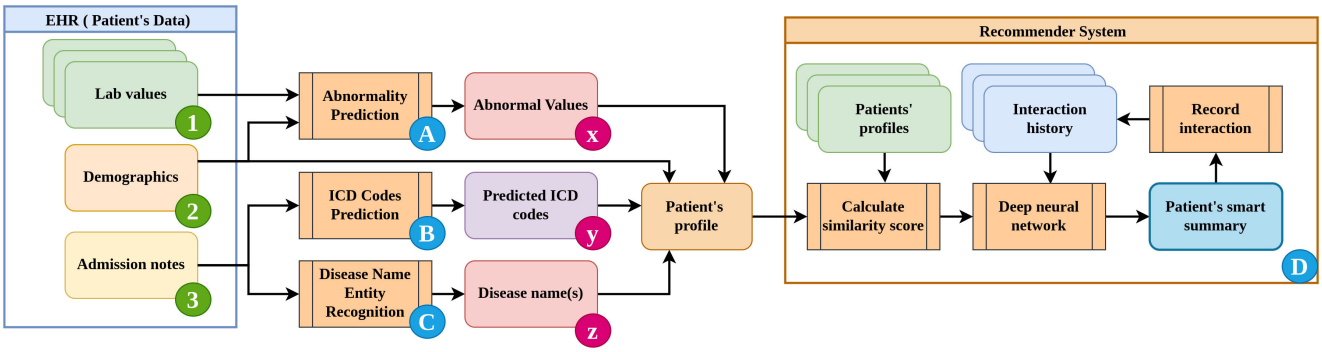

FIGURE 2. The framework of the proposed DRS. The system has 3 modules (A, B, C), and their outputs (x,y,z) are used to generate the patient profile. This profile is used to generate recommendations and record interactions.

known to be recorded at admission. These include a history of present illness, medical history, admission medications, and family history.

Each one of these data types will be input into the three main modules: the abnormality prediction module, the ICD codes prediction module, and the disease-named-entity recognition module. The output of these modules is used to construct the patient's profile by stacking the patient demographics, predicted abnormal values, predicted ICD codes, and the extracted disease names together. The profile is used by the RS to calculate the similarity scores with other patients. Then, the RS will rank the lab values and show the ones the model thinks most relevant to the medical staff, alongside the predicted ICD codes and the extracted diseases' names. Finally, the system will record the doctors' interaction with the recommended summary and save it in the interaction history so the RS can learn from it the next time the system works on a similar patient. In the next sections, we will discuss each sub-module in more detail and the fully distributed RS architecture.

A. LAB VALUES ABNORMALITY PREDICTION MODULE

The main purpose of this module is to analyze the patient's lab values and classify which lab values are predicted to be abnormal in the future (the next time that test is done) and which will be normal. This module has been introduced in our previous work [15]. The module can be seen in Fig. 3, and it consists of the following main steps:

1) PREPROCESSING

The MIMIC-III dataset contains many different lab values, which necessitate preprocessing steps. In this work, we have chosen a comprehensive patient data profile comprising 25 lab values (L1, L2,..., L25) for each individual, covering the training, validation, and testing phases. These values were extracted as a multidimensional discrete time series at 4-hour intervals and were aggregated through appropriate averaging or summation methods. The lab values selected were deemed the most frequent and most relevant to the patient cohort focusing on mechanically ventilated patients, as identified in our previous work [14] by the medical experts from the University Hospital of Rheinisch Westfälische Technische Hochschule (RWTH) Aachen. Additionally, we had to apply methods to ensure that we got a proper time series suitable for the machine learning pipeline. First, we applied the time-windowed sample-and-hold method to fill in most of the missing values. The lab sample is held (repeated) for a certain

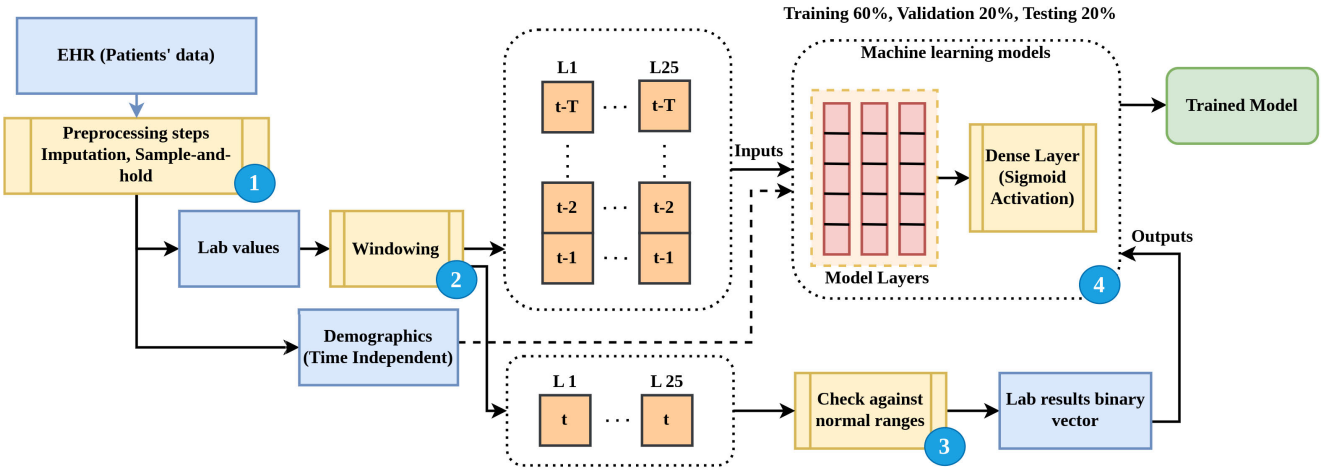

FIGURE 3. The lab values abnormality prediction (module A in Fig. 2) module introduced in [15].

time, depending on the frequency of the value or until the next available value. The data point will be ignored if the holding time exceeds the calculated maximum holding time. Additionally, k-nearest neighbor imputation with singular value decomposition (SVD) was used to fill the remaining missing values. Moreover, any ICU stay with more than 50% missing values was discarded. Finally, Tukey's range test was used to detect and remove anomalies in the data [14].

2) WINDOWING

The resulting regular time series will be split into multiple shorter sequences using the moving window technique. Each resulting windowed sequence will be used as an input sample, including the lab values from $t - T$, where T is the window size, till $t - 1$. The last time step of each sequence t will be used as the corresponding output binary vector [15].

3) CHECK AGAINST NORMAL RANGES

As mentioned before, the lab values vector from the last time step of each window will be compared against the reference ranges from the American College of Physicians (ACP) [21]. The output will be a binary vector representing the predicted abnormality of the corresponding lab values.

4) PREDICTION MODELS

The classification model's job in our work is to predict for an input sequence of lab values an output binary vector that represents which of the lab values are predicted to be normal (denoted by 0) or abnormal (denoted by 1). In our previous work [15], we tested multiple machine learning models used for time series classification. The models are long-term short memory (LSTM), convolutional neural network (CNN), Multi-CNN (MCNN), Transformer, Temporal convolutional network, and LightGBM. For example, the MCNN model's architecture can be seen in Fig. 4. The multiple convolutional operations allow processing the input of the sequence

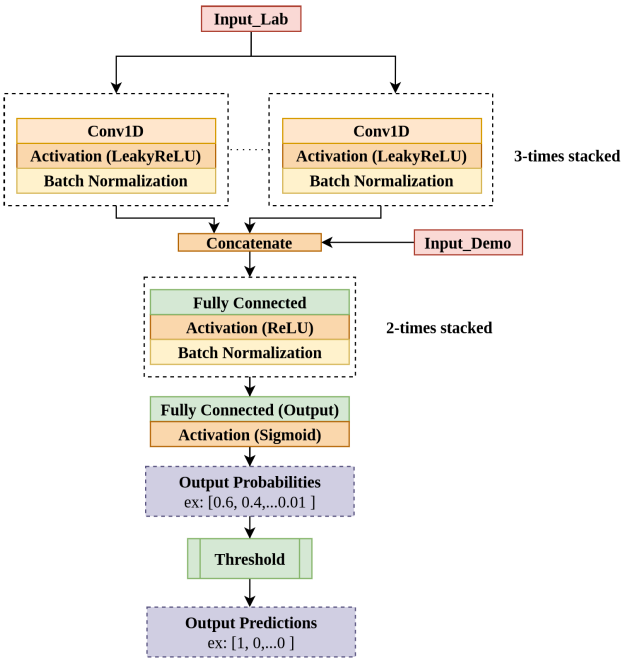

FIGURE 4. The Multi-CNN architecture used in the lab values abnormality detection module [15].

from different perspectives, capturing multiple different time dependencies.

B. ICD CODES PREDICTION MODULE

The International Classification of Diseases (ICD) is a healthcare classification system developed and maintained by the World Health Organization (WHO) [22]. This system assigns unique codes to various diseases and health statuses based on specific rules. Currently, the classification of diseases heavily relies on human resources, and professionals must review a large amount of textual data to make classifications. Even with professional disease classifiers, this process is time-consuming and requires a balance of

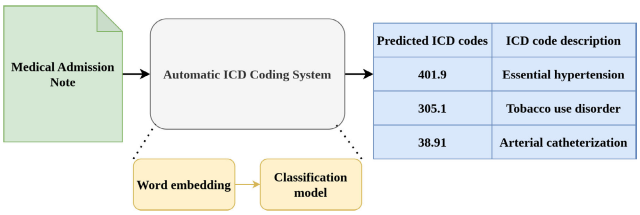

FIGURE 5. Illustration of automatic ICD coding system. The system tokenizes the admission notes and classifies the resulting word embeddings into output ICD codes.

efficiency and accuracy. The use of artificial intelligence in this process could potentially save time for hospitals. In our work, we built an automatic ICD-9 code classification system that uses natural language processing (NLP) techniques as well as other machine learning classification methods on the admission note to predict the ICD codes which will be used to build the patient's profile that the recommender system needs to give its recommendations. The system uses the subjective component of the admission report that primarily includes subjective data about the patient, including patient complaints, feelings, and opinions, where no ICD codes are mentioned. This allows the ICD prediction system to serve as a self-diagnosis assistance and patient sorting tool by describing their condition before seeking medical advice. Furthermore, we have chosen the ICD-9 coding system instead of the newer ICD-10 because of the prominent availability of the first (ICD-9) in the MIMIC-III dataset. Fig. 5 illustrates our automatic ICD coding system where the system predicted 3 ICD codes. The system starts with encoding the text from the admission reports with tokenizers to a sequence of integers called input tokens. Then, the input tokens are mapped into vectors that encapsulate the meaning of the words they represent in a process called word embedding. Moreover, these vectors will be used as input to a classification model that outputs the predicted ICD codes for the input text. Generally, we can view this task as a multi-label text classification task.

There has been a lot of work on ICD code prediction using different approaches and datasets, like the work in [23] and [24]. Many methods utilized different models that were applied to different data sources, such as death certificates or radiology reports. Additionally, some approaches focused on predicting the full ICD codes, while others focused on a partial subset. Therefore, it is hard to compare the performance of the majority of the work done in this field. Our research focuses on work that applies deep learning methods on unstructured text (admission reports) included in the MIMIC-III dataset to predict ICD-9 codes. Most approaches here utilize a different combination of word embedding and classification models. Generally, the classification models can be grouped into the following types:

- Long-term short memory (LSTM): It is a type of recurrent neural network (RNN) that is used often for sequence classification tasks. The LSTM classification model receives the word embeddings as a sequence and outputs the most probable code/s for the patient. It has been used for automatic ICD coding in many works, including the work in [24].
- Convolutional neural network (CNN): CNNs have many applications in computer vision and NLP. Moreover, in the context of automatic ICD coding, some authors used CNNs for this task, like the work in [25]. We have used CNN for our task, similar to the model in Fig. 6, where we have one Conv1D stream and no attention attached.
- Attention: The attention mechanism is first introduced in [26], which is used for machine translation. In our work, we apply the multiplicative attention variant [27]. Finally, due to the nature of the attention mechanism, it is possible to provide an intuitive explanation through the visualization of the attention weights. This characteristic is particularly important when applying the mechanism in the medical domain, as it requires a high level of explainability.

Additionally, combinations of the aforementioned model types were proposed in the literature to learn various levels of features or input embeddings and enhance classification accuracy. For example, the combination of CNN and LSTM could capture both the local and global features within the texts. Moreover, the authors propose the state-of-the-art model in the field of automatic ICD coding in [28], which utilizes a combination of a CNN with the attention mechanism on top of it. The model uses only one CNN layer applied to the word embeddings and an attention layer afterward. In this work, we propose a modified hybrid architecture for automatic ICD coding with multiple CNN layers with different filter lengths applied in parallel on the word embeddings. The output of each CNN layer is then followed by a self-attention layer. The architecture of our system is shown in Fig. 6. The different convolutional layers and the different filter sizes allow processing of the input embeddings sequence from different perspectives, capturing different dependencies between the tokens. Moreover, the attention model on top of the CNN layers allows for the explainability enabled by measuring the attention weights. Finally, we have experimented with the aforementioned types of models for our ICD codes prediction task, and our modified architecture proved to work better than the SOTA model, as shown in the results section.

C. DISEASE NAME RECOGNITION MODULE

The main goal of a named-entity recognition (NER) system is to extract key information (entities) from unstructured texts. The task is to identify entities in texts and classify the detected entities into predetermined information units such as country, time, person, etc. In our case, we use the disease-named-entity recognition (DNER) system to extract the names of diseases related to a patient from the doctor's admission notes. This information is very useful for building the patient's profile to be used by the DRS and making it

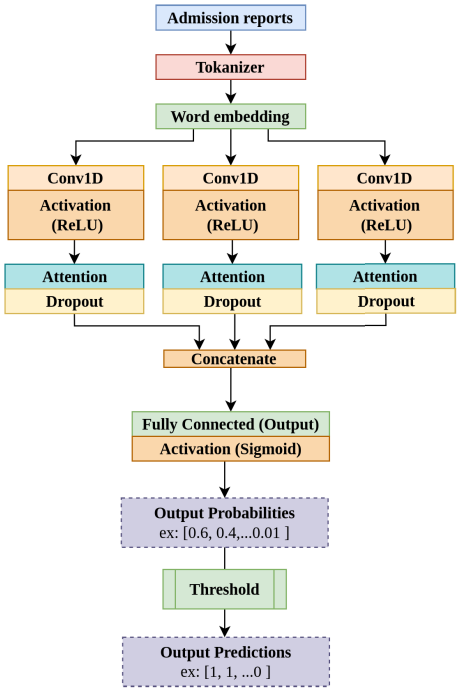

FIGURE 6. The neural network architecture of our automatic ICD coding model.

easier for doctors to find this information rather than looking for it in the full admission text.

For named-entity recognition, one of the recent models often used is the Bidirectional Encoder Representations from Transformers (BERT) model [29]. It is a contextualized word representation model that considers the contextual meaning behind words, unlike word embeddings, which give the same vector to words that share the same spelling but have different meanings. Furthermore, after fine-tuning the BERT model, it could be used to train downstream tasks, such as question answering, named entity recognition, and document classification. Therefore, this would reduce the cost of re-establishing architecture for different NLP assignments.

The two main data sources for training BERT are Wikipedia and BooksCorpus. However, the two data sources are relatively general datasets. On the other hand, training models for DNER require data sources that are more medical-related. Therefore, BioBERT is established to bridge this gap [30]. Apart from Wikipedia and BooksCorpus, BioBERT consists of PubMed abstracts and PMC Full-text articles [30]. Furthermore, BioBERT works well for disease-named entity recognition as the work in [31]. Therefore, we have chosen this model for the DNER task. Fig. 7 shows an example input/output of our DNER system. The word labels are encoded using the BIO scheme. In this scheme, when a word is tagged as "B", it refers that it is at the beginning of the disease entity, "I" represents the middle of the entity, and "O" (outside) describes those that are not relevant to our entity. Finally, we extract the words labeled with "B" or "I" from the text, and these words represent the disease name/s the patient has.

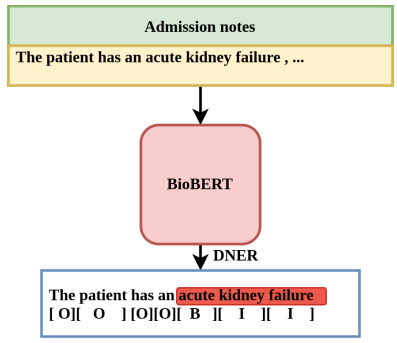

FIGURE 7. Illustration of our disease named-entity recognition system. The BioBERT model classifies each word in the sentence. These output classifications are used to identify the full disease name/s.

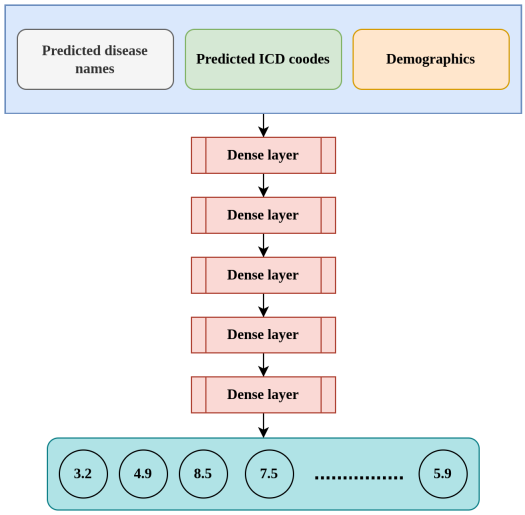

FIGURE 8. Simple illustration of our recommender system. The neural network will learn from the input patient profile and output a score of importance for each lab value.

D. RECOMMENDER SYSTEM

Our recommender system, which is presented as sub-module D in Fig. 2, receives the patient's profile that consists of the list of abnormal lab values from sub-module A, the predicted ICD codes from sub-module B, and the disease names from sub-module C. The patient's profile is used to group similar patients and provide a recommendation similar to the closest patient. Additionally, doctors can provide feedback on the recommended results by rating them. Then, the system uses the interaction history from doctors to relate the recommendations to the respective patient group and optimize the recommender system to achieve a more accurate recommendation for new patients that are similar to the ones the doctors already gave feedback for.

The architecture of our recommender system can be viewed as a regression and a ranking task. Fig. 8 shows the architecture of our proposed recommender system. The architecture inputs are the predicted disease name, ICD codes, and demographics, while the outputs are the predicted rating of each lab value (from 0 to 10). Due to the lack of a training dataset with predefined ratings, which is our target

for the recommender system, we created our training dataset by randomly distributing ratings based on the normality of each lab value. Since the exact lab values that doctors are expected to see are still unknown initially, we first assume that all abnormal values are more likely to be the most important ones. Therefore, we randomly distribute 6 to 10 points to the abnormal values and 1 to 5 points to those that are normal to train our model. The ranking is then based on the predicted rating. For instance, lab values ranked in the top five will be recommended to the doctors at the top of the list. We will illustrate more on the architecture implementation in the results section.

To test whether our recommender system can learn the similarities between patients, we try to mimic the behaviors of doctors. Assuming that doctors would give similar patients the same lab value ratings, we use clustering techniques to group patients based on the predicted disease name, predicted ICD codes, and demographics. We use the K-means clustering technique for the segmentation of the patients [32]. The algorithm takes the patient data and the desired numbers of clusters (groups) as an input and outputs the respected group for each patient as an output according to its algorithm [32].

After patients are clustered into groups, we can use the groups to create our feedback dataset to mimic the doctor's behavior and test our recommender system. We will distribute the same ratings to the patients in the same group based on the assumption that doctors would give similar patients the same lab values ratings. Finally, the recommender system will also show the predicted ICD codes and their meanings, as well as the extracted disease names, to give a general overview of the patient case, which would make handling cases easier and faster.

E. SPLIT LEARNING

For our recommender system to work well, we need to have access to enough example cases. However, medical data originates from many different distributed sources (hospitals, clinics, etc…) and often follows strict rules when it comes to sharing patient data between these sources. This leads to the data staying where it originated and reduces the total amount of training data available to the local machine learning models. Therefore, we propose using a modified version of split learning for our DRS as a distributed machine learning technique that ensures privacy for patients' data and is energy efficient.

In split learning (SL), the machine learning model is split between the server and the client, as shown in Fig. 9 [19]. During training, each client trains its local layers on its local data. Then, the clients will send the output of the last layer of their model (split layer) to the server, which continues propagating through its layers and calculates the loss. During backpropagation, the server propagates the gradients through its layers and sends the gradients of its split layer to the client to continue backpropagating through its layers. This method ensures the collective learning of local models without sharing the raw patient data.

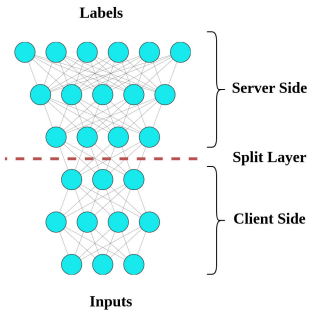

FIGURE 9. Simple illustration of the split learning mechanism. The neural network is split into parts, each residing in a different participating entity.

Split learning has the advantage over distributed machine learning like federated learning (FL). For example, SL is more computationally efficient as it trains only part of the model on the client [33]. Additionally, SL converges in most cases faster than FL [33] and, in many cases, has less overall communications overhead [34]. However, to reduce the computation and communication overhead of SL even further, we proposed, in our previous work [20], a modified split learning scheme that adds an autoencoder (AE) neural network and an adaptive threshold mechanism.

The autoencoder network (AE) is an unsupervised model that learns to recreate its input with minimal reconstruction loss. It consists of two parts: an encoder and a decoder. In our system, the encoder compresses the input into a much smaller latent vector that the decoder uses to reconstruct the input signal. The encoder model resides at the client and learns to compress the output of the split layer during forward propagation into a much smaller latent vector. That latent vector will then be sent to the server instead of the full output of the split layer, saving a lot of communication overhead. Additionally, the decoder model resides on the server side. It reconstructs the output of the client's split layer from the received latent vector and the forward propagation continues through the server layers. The AE in our system can be trained either on a similar dataset and then used as nontrainable layers or trained with the rest of the network for the first few epochs and then used as nontrainable layers.

The second addition to the original SL system is at the end of the forward propagation when the loss is calculated. The loss will be compared to a certain function (static threshold or a function of the epoch number) to decide whether the update is significant. If the loss is significant, the gradients will propagate through the server and the client layers. However, if the loss is insignificant, the gradients will only propagate through the server layers. Ignoring specific insignificant updates at the client can save significant communications overhead as well as computations at the client. In our previous work [11], we implemented our modified split learning mechanism to classify ECG signals, and the mechanism reduced the communication overhead by 73% and the computations at the client by 28% with

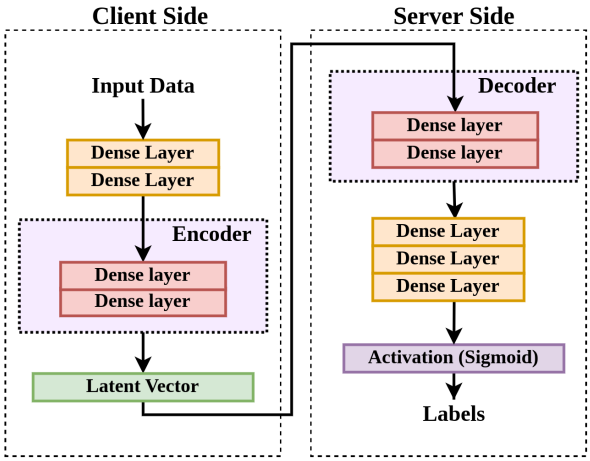

FIGURE 10. The architecture of our distributed recommender system. One hospital/clinic will assume the server role, and other hospitals/clinics will have the client train on their local data.

only 3.77% drop in the area under the curve (AUC). The significant reduction in the communication and computation overhead with a slight performance degradation proves that the modified SL can add a layer of privacy and security to the patients' data while reducing the communication and computation overhead needed to achieve this. Therefore, we have adapted our modified SL to the recommender system as shown in Fig. 10. In the results section, we will discuss the experiments we did to test the validity of the integration between the modified SL and the recommender system.

III. EXPERIMENTS AND RESULTS

A. DATASET AND COHORT DEFINITION

The main dataset we used to produce the subsets needed for the different modules in our DRS is the MIMIC-III dataset [35]. This dataset is a public-accessed dataset that consists of clinical texts and records from the intensive care unit (ICU) at Beth Israel Deaconess Medical Center in Boston, Massachusetts, from 2001 to 2012. It includes data for 31,532 unique ICU stays. In our previous work [15], we have used a cohort of patients focused on mechanically ventilated patients as the work in [14]. Using this cohort, we extracted 11, 943 ICU stays with mechanical ventilation events from the MIMIC-III dataset. The duration of the ICU patients' stays ranges from 12 hours to 72 hours in 4-hour time steps. Most of the lab values are recorded every 4 hours. Therefore, we considered a one-time step unit to equal 4 hours. Additionally, Table 2 shows patient demographics and clinical characteristics. The input data consists of 3 demographic features (age, sex, weight) and 25 lab values (white blood cell count, PaCO2, hemoglobin, etc). The lab values chosen are the most frequent in the dataset and are the most relevant to a mechanically ventilated patient [14]. Additionally, for each stay, we have the admission notes, including a history of present illness, medical history, admission medications, and family history. The admission notes' length varies largely (navg words/note = 396.3, std words/note = 233.3), which puts a lot of emphasis on the preprocessing steps required to work with the text.

TABLE 2. Clinical and demographic properties of our MIMIC cohort. Data is presented in n (%), mean (SD) or median (IQR); LOS: length of stay [15].

Number of ICUs	5
Data acquisition timespan	2001-2012
Number of included patients (N)	11,443
Age (years), median (IQR)	66.9 (56.3-77.5)
Body weight in kg, mean (SD)	85.7 (18.1)
Sex, female, n(%)	4,329 (36.3%)
Sex, male, n(%)	7,614 (63.7%)
In-hospital mortality, %	11.1
LOS in ICU in days. median (IQR)	3.1 (1.6-6.1)
Admission report length in words, mean (SD)	396.3 (233.3)

TABLE 3. Testing results for the different models for all lab values (micro-average) on the MIMIC-III dataset [15].

Model	Accuracy	Precision	Recall	F1-score
LSTM	0.85	0.83	0.87	0.85
CNN	0.86	0.84	0.85	0.84
Multi-CNN	**0.88**	0.87	**0.89**	**0.88**
Transformer	0.86	**0.88**	0.81	0.84
TCN	0.86	0.87	0.85	0.86
LightGBM	0.83	0.82	0.76	0.78

B. ABNORMAL LAB VALUES PREDICTION

For our multi-label classification task, we experimented with LSTM, CNN, TCN, and Transformer-based architectures. The models were trained and tested on both the MIMIC-III dataset and the eICU dataset [36]. By cross-validating our algorithms across these two datasets, we not only broaden the scope of performance comparison but also gain insights into how diverse algorithms can generalize on previously unseen data. The results for the models are shown in Table 3. The Multi-CNN model had the best results in almost all the metrics. Therefore, we have chosen this model for our recommender system. The details of the models and the training process can be found in our previous work [15].

C. AUTOMATIC ICD CODING

Similar to previous work [37], we focused on the discharge summaries, which provide us with basic information about a patient and are labeled based on a set of ICD-9 codes by human coders, describing the diagnosis and procedures during the patient's stay. Additionally, the subjective component of the admission report, which has no ICD codes, is included in the discharge reports. A total of 8, 921 unique ICD codes are included in the MIMIC dataset. However, in our study, we selected the top 50 and top 30 as the most frequent codes to experiment with. The 50 top codes are used a lot in literature, but we wanted to experiment with the top 30 codes as well to study the effects of the number of ICD codes on the performance of the RS. Every admission note in this study has at least one of the top 50 or 30 ICD codes. The filtered result has 8, 066 summaries for training, 1, 573 for validation, and 1, 729 for testing in the top 50 contexts, whereas 7, 919 summaries for training, 1, 519 for validation, and 1, 693 for testing in the top 30.

TABLE 4. Results of the different machine learning models for the automatic ICD codes prediction task on the MIMIC-III-50 test set.

Model	AUC		F1-score		Precision	
	Macro	Micro	Macro	Micro	Macro	Micro
CNN	0.88	0.91	0.55	0.65	0.63	0.71
CNN (+Att)	0.89	0.92	0.55	0.65	0.65	**0.73**
Bi-LSTM	0.87	0.91	0.51	0.62	0.63	0.70
Bi-LSTM (+Att)	0.89	0.92	0.55	0.65	0.60	0.72
RCNN	0.87	0.90	0.53	0.59	0.64	0.70
RCNN (+Att)	0.89	0.92	0.56	0.66	0.65	0.72
MultiCNN	0.89	0.91	0.57	0.66	0.62	0.67
MultiCNN (+Att)	**0.90**	**0.93**	**0.58**	**0.64**	**0.66**	**0.73**
TCN	0.83	0.87	0.50	0.58	0.60	0.68
TCN (+Att)	0.87	0.91	0.55	0.63	0.57	0.66

TABLE 5. Results of the different machine learning models for the automatic ICD codes prediction task on the MIMIC-III-30 test set.

Model	AUC		F1-score		Precision	
	Macro	Micro	Macro	Micro	Macro	Micro
CNN	0.90	0.93	0.63	0.70	0.70	0.74
CNN (+Att)	0.90	0.93	0.61	0.68	0.70	0.74
Bi-LSTM	0.89	0.92	0.60	0.67	0.70	0.73
Bi-LSTM (+Att)	0.90	0.93	0.64	0.70	0.69	0.74
RCNN	0.89	0.92	0.63	0.69	0.69	0.73
RCNN (+Att)	0.89	0.92	0.63	0.68	0.68	0.72
MultiCNN	0.90	0.92	0.64	0.70	0.69	0.73
MultiCNN (+Att)	**0.91**	**0.94**	**0.65**	**0.71**	**0.71**	**0.75**
TCN	0.86	0.90	0.58	0.65	0.67	0.73
TCN (+Att)	0.88	0.91	0.60	0.67	0.60	0.67

Following the previous work by the authors in [25] and [37], we tokenized the text, removed tokens relating to non-alphabetic characters, and then transformed the tokens into lowercase. We used the preprocessed data from all the admission notes to train the word embeddings with the dimension size of 100 using the CBOW Word2Vec method [38]. All admission notes were truncated to a maximum length of 2500 tokens to reduce the computation cost since no significant performance differences were shown when truncating between 2500 and 6500 [25]. We have tested with CNN-based models as well as Bi-LSTM, RCNN, Multi-CNN, and TCN. For each model, we attached the attention mechanism on top of the model to add explainability to the model. The metrics used to compare the models are area under the curve (AUC), F1-score, and Precision. Furthermore, a 10-fold cross-validation approach was employed to evaluate the performance of our tested model thoroughly. The mean performance metrics were calculated over the 10 folds, providing a robust assessment. Tables 4 and 5 show the results for the different models, with and without the attention mechanism attached (denoted by +Att). Our Multi-CNN model, with the attention mechanism on top of it, outperformed other modes in all metrics in the top 50 and 30 ICD codes with very little variance in classification performance between the different folds. Additionally, we can see that the results are generally better for classifying the top 30 ICD codes than the top 50 ICD codes since there are fewer output classes.

D. DISEASE NAMED-ENTITY RECOGNITION

The initial dataset used for the model training of the disease name recognition task is the National Center for Biotechnology Information (NCBI) disease dataset [39]. However, we used for training the pre-processed version of the dataset provided by [30] and [40]. The word labels are encoded using the BIO scheme. Moreover, there are 5, 710 sentences and a total of 196, 282 words for training, 635 sentences and a total of 21, 807 words for validation, and 935 sentences and a total of 33, 391 words for testing. Additionally, we extracted 600 sentences and a total of 20, 321 words from the MIMIC-III dataset admission reports for cross-validation. We used a pre-trained BERT-tokenizer from HuggingFace [41] to tokenize words into subwords or tokens. We implemented our fine-tuned BERT model with BioBERT pre-trained weights [30] using the TensorFlow BERT library, which is provided by HuggingFace [41]. The output of the BioBERT model is linked to a fully connected layer to perform token classification. The input texts were truncated or padded with a ''PAD'' character to a max length of 128 tokens. We used the Adam optimizer with a learning rate of 3×10^{-5} to train our fine-tuned BioBERT model to minimize the masked cross-entropy loss, which will ignore the ''PAD'' character when calculating loss. Table 6 shows the validation results (mean) of the BioBERT model for the disease-named-entity recognition task on the preprocessed version of the NCBI disease dataset and our own MIMIC-III validation subset. We can see that the BioBERT model achieves satisfactory results on both datasets. Therefore, we used that trained model to extract disease names from the patients' admission reports in MIMIC-III.

TABLE 6. Classification results for the disease-named entity recognition task on the NCBI and the MIMIC-III datasets.

Class	F1-Score (NCBI)	F1-Score (MIMIC-III)
B	0.89	0.88
I	0.91	0.92
O	0.99	0.98
Macro-Average	0.93	0.92
Micro-Average	0.98	0.97

E. RECOMMENDER SYSTEM

The three inputs of our recommender system are the disease names, the demographics, and the ICD codes. The output of the recommender system is the top n recommended lab values. As we are treating our recommender system as a regression and ranking system, we would first predict the ratings for each lab value, and the rankings would be generated afterward. We mapped the disease names using pre-trained embeddings, which are the same as those used in the automatic ICD coding system. Then, we transformed the variable-sized disease names into a fixed length of disease name feature vectors by averaging them to feed them into the dense layer. All continuous features in demographics data, such as the patient's age and weight, were normalized and transformed at the same scale in the range of 0-1. The predicted ICD codes are either the top 50 or the top 30 codes. The difference in the recommender system performance between the top 30 and 50 was minor, so we chose the top

50 to include more patients. Moreover, the ICD codes were encoded using one-hot encoding so the sum vector of the predicted codes for a specific patient is a binary vector, which is the output of an *OR* operation between the one-hot encoded vector of each predicted ICD code. Finally, the patient profile is built by stacking the demographics, the one-hot encoded ICD codes vector, the disease names averaged embedding, and the binary vector representing the abnormal values.

To train our recommender system, we need our three aforementioned inputs, and for the output, we need the past ratings for each lab value. The score or rating stands for the level of concern, with a higher score indicating a more important value, ranging from 1 (lowest importance) to 10 (highest importance). The extracted disease names and predicted ICD codes for each patient will also be shown as part of the generated smart summary.

Initially, we do not have reference lab value ratings for each patient. Therefore, we built the initial training dataset, as mentioned before, by randomly assigning a rating value for each lab value, giving a higher rating to abnormal values. Furthermore, we built a feedback dataset to examine our recommender system's learning capabilities. We used the *k-means* clustering algorithm to group similar patients by their disease names, demographics, and ICD codes [32]. The same scores were distributed to the patients in the same clusters, serving as our feedback dataset's target labels.

Since the main goal of our recommender system is to recommend top n lab values, it could be viewed as a ranking system. Therefore, we used *precision@n* to evaluate our system as used in the literature for such problems [42]. *Precision@n* is used to measure the proportion of the recommended n items that are indeed relevant to the users. Equation 1 shows how the *Precision@n* is computed.

$$\text{Precision@n} = \frac{\text{recommended n items} \cap \text{relevant items}}{\text{recommended n items}}$$

As discussed before, we trained the RS on two datasets: the initial dataset, which contains the random weights emphasizing the importance of abnormal values, and the synthetic feedback dataset, which contains the feedback ratings of the lab values. The initial dataset consists of 9, 528 training samples with no testing samples as there is no need for such. On the other hand, the feedback dataset consists of 2, 324 training samples and 9, 528 testing samples. We first trained the RS on the initial dataset for 50 epochs to warm-start the system to be able to recommend the abnormal lab values as the most important ones. Then, we trained on the feedback dataset for another 50 epochs. Since we emphasize the doctor's feedback, 1.5 weights were assigned to the feedback dataset. This gives more importance to these samples in the model's loss function and pushes the model learning further from the feedback dataset. Furthermore, the feedback weights can be adjusted based on the doctor's experience. For instance, feedback from less experienced doctors can be assigned lower weights compared to that from more experienced doctors, thereby minimizing the influence of less reliable samples on the learning process. However, the impact of medical errors and varying levels of doctor experience on the overall learning efficacy of the recommender system across different entities is beyond the scope of this work.

TABLE 7. Results of the RS on the top 10 recommended lab values in the testing set.

Number of groups/clusters	Precision@10	
	Before training on feedback dataset	After training on feedback dataset
5	0.21	**0.91**
10	0.21	**0.92**
15	0.28	**0.87**
20	0.26	**0.87**
25	0.21	**0.86**
30	0.18	**0.85**
40	0.21	**0.85**
50	0.21	**0.84**

We set a batch size of 64 for the training and used RMSprop with a learning rate of 0.01 to minimize the minimum squared error (MSE). The 5 dense layers in our RS model have 256, 256, 128, 128, and 128 neurons respectively. We ran the experiments 100 times with different dataset splits and different random initial datasets to gauge the statistical significance of the results. Finally, table 7 shows the mean *precision@10* of our RS on the test portion of the feedback dataset before and after training on the feedback training dataset. We can see that the results were sub-optimal before training on the feedback dataset, indicating that the system learned how to predict only abnormal lab values, which are not necessarily what the doctors want. However, after training on the feedback dataset, we can see that the system's performance improved significantly, indicating that our system learned how to recommend relevant lab values to patients with similar cases. Additionally, we can also notice how the performance of the RS deteriorates with the increase in the number of patient groups/clusters until it stabilizes at around 0.86. The cluster number must be chosen carefully according to the data. Methods like the elbow curve method or the silhouette method can be used to identify the best optimal number of clusters. For example, in the elbow method, the sum of squares at each number of clusters is calculated and graphed, and the user looks for a change of slope from steep to shallow (an elbow) to determine the optimal number of clusters [43], [44]. In our case, that number is 15 clusters. More than 15 clusters do not add meaningful values to the clustering. Additionally, the number of clusters could be attached to the number of main or sub-classes of a certain patient cohort. For example, the number of patient clusters can be chosen to represent the number of categories of mechanically ventilated patients [45].

F. DISTRIBUTED RECOMMENDER SYSTEM

To test our DRS, we have implemented the setup shown in Fig. 11. The system consists of 4 clients and a server. The clients and the server are simulated on general-purpose

TABLE 8. The precision @10, communication overhead, and computation overhead of our DRS on one client after training for 50 epochs on the feedback dataset.

Configuration	Average Dismissal Rate/Epoch (%)	Average Data Transfer/Epoch (MB)	Data Transfer Reduction (%)	Precision @10	GFLOPS at Client/Batch	GFLOPS Reduction (%)
Basic SL	0	2230	0	86%	0.542	0
SL + AE	0	1226	45	85%	0.553	-2.1
SL + AE + ATM (sigmoid)	49.2	624	72	83.8%	0.297	45.2

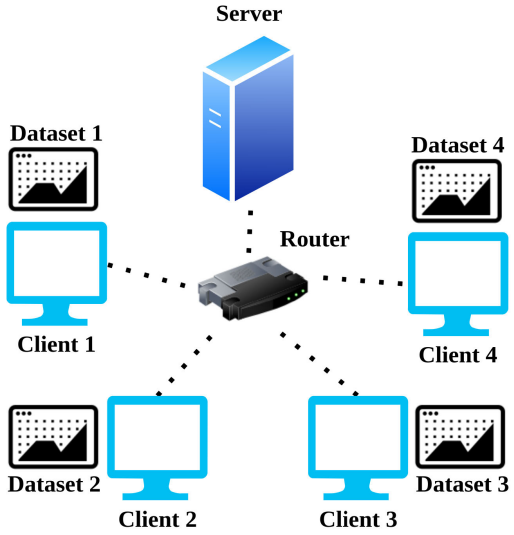

FIGURE 11. The distributed recommender system test setup. We have four clients, each with a local data split and one server.

desktop computers. Each client has the same client model as in Fig. 10. The encoder consists of two dense layers with 128 and 32 neurons, respectively. On the other hand, the decoder consists of two dense layers with 32 and 128 neurons, respectively. The autoencoder is trained with the rest of the RS network when it is trained on the initial dataset that emphasizes the abnormal values. Then, the autoencoder will stop training and be used as non-trainable layers when training on the feedback dataset. Moreover, a sigmoid function is used during backward propagation to prevent sending insignificant updates to the client, as mentioned in our previous work [11].

The feedback dataset was uniformly distributed among the 4 clients. Each client has 581 samples for training and validation. The testing dataset is the same for all clients and consists of 9, 528 samples. We used weight sharing between clients to ensure that all clients will learn from others during training and that their performance will be equal afterward. Table 8 shows the communication overhead, computation overhead, and the *precision@10* for one client with different system configurations. First, we can notice that adding the AE to the basic DRS with SL reduces the data transferred between the client and the server by approximately 45%. All while reducing the *precision@10* by only 1% and increasing the computations at the client by 2.1%. Second, adding the ATM (sigmoid) to the DRS alongside the AE allowed us to further reduce the communication overhead by 72% compared to the basic DRS with SL leading only to 2.2% reduction in *precision@10*. Additionally, ignoring insignificant updates at the client led to a reduction in client computations by 45.2%. This means that the client (hospital, clinic, etc..) could run the training faster using less energy. All while keeping patients' data private. This is the overall system performance we were allowed to achieve on our developed DRS. Moreover, our enhanced split learning mechanism, as demonstrated in our prior research [11], exhibits scalability to a significantly larger client base. Consequently, the recommendation system (RS) stands to gain from accessing extensive training patient data across numerous hospitals/clinics, ensuring privacy is upheld throughout.

G. EXPLAINABILITY

In healthcare, there is an increasing desire for AI techniques that are effective and explainable to the user [46]. Therefore, we have ensured explainability both at the level of the RS and the level of the sub-modules that make up the RS. First, for the ICD code predictions, we used the Multi-CNN module with the attention mechanism, which makes it possible to provide an intuitive explanation through the visualization of the attention weights. Second, for the DNER module, we can show the full admission note and the extracted disease name/s, which can be checked directly by the medical staff. Finally, if the system has not yet been trained on the feedback dataset, it will prompt the users and draw their attention to the fact that the system is recommending only abnormal lab values. On the other hand, if the system has been trained on doctors' feedback, it will show, for a certain patient, similar patients who already received feedback from doctors. This will help the medical staff to understand why the system recommended a certain output. The test RS, alongside the explainability modules integrated into it, can be seen on the paper's Github repository [47].

IV. CONCLUSION

In this paper, we propose a private and efficient distributed recommender system for patients in the intensive care unit that recommends relevant lab values, highlighting the ones that are predicted to be abnormal in the next 4 hours. The system has three sub-modules that read the patients' data and produce the patient profile for the RS. First, the system reads the patient's previous lab values and classifies which lab values are predicted to be abnormal in the next 4 hours. We tested multiple models and chose the Multi-CNN model, which performed best with an F1-score of 0.88 on the test dataset. Second, the system predicts the patients'

ICD codes based on the subjective part of the admission notes, which can help the medical staff automatically classify the patient's cases. We have tested multiple models on the MIMIC-III dataset and developed a model based on the Multi-CNN with attention architecture that scored an F1-score of 0.71 (micro-average) on the top 30 codes dataset, outperforming existing models. Third, the system reads the admission notes and extracts entities referring to disease names. We have used a SOTA model called BioBERT for this task, achieving an F1-score of 0.97 (micro average) on our testing subset from MIMIC-III. Finally, our RS uses the data generated from the different modules to recommend the most relevant lab values to a particular patient, highlighting predicted abnormalities in lab values. Doctors can also provide feedback on the output of the RS to improve its future output for similar patients. We use K-means clustering to group similar patients and to create a feedback dataset to mimic doctors' behavior by giving the same rating to the patients in the same cluster or group. Our RS achieved a *Precision@10* score of 0.86 when trained on our feedback dataset (with 25 groups of patients), meaning the system learned from the feedback and gave better recommendations on the test dataset. Moreover, our RS runs in a distributed manner using our novel modified split learning mechanism, where a small part of the model is implemented at each client (hospital, clinic, etc.), and the rest of the model is hosted on the server. Using a test system with four clients, our system achieved a *Precision@10* score of 0.838 while reducing communication and computation overhead by 72% and 45.2%, respectively. This means that our RS system achieves privacy while being faster and more energy efficient. There are several promising avenues for future work. Firstly, enhancing model robustness and interpretability by including additional patient data sources, such as genetics or environmental factors, could be explored. Secondly, adapting the recommendation process in real-time based on patient feedback and evolving medical guidelines could optimize patient care. Additionally, integrating advanced machine learning techniques like reinforcement learning may further improve recommendation accuracy, especially when receiving suboptimal feedback from inexperienced doctors or in the case of a medical error. Lastly, conducting rigorous clinical validation studies to assess real-world impact on patient outcomes and healthcare workflow efficiency would be essential for widespread adoption.

V. CODE AVAILABILITY

The full code used to produce this work is available via Github https://github.com/a-ayad/smartsummary [47].

REFERENCES

[1] A. Hall and G. Walton, "Information overload within the health care system: A literature review," *Health Inf. Libraries J.*, vol. 21, no. 2, pp. 102–108, Jun. 2004.

[2] I. Klerings, A. S. Weinhandl, and K. J. Thaler, "Information overload in healthcare: Too much of a good thing?" *Zeitschrift Evidenz, Fortbildung Qualität Gesundheitswesen*, vol. 109, nos. 4–5, pp. 285–290, 2015.

[3] F. O. Isinkaye, Y. O. Folajimi, and B. A. Ojokoh, "Recommendation systems: Principles, methods and evaluation," *Egyptian Informat. J.*, vol. 16, no. 3, pp. 261–273, Nov. 2015.

[4] M. Wiesner and D. Pfeifer, "Health recommender systems: Concepts, requirements, technical basics and challenges," *Int. J. Environ. Res. Public Health*, vol. 11, no. 3, pp. 2580–2607, Mar. 2014.

[5] T. N. T. Tran, A. Felfernig, C. Trattner, and A. Holzinger, "Recommender systems in the healthcare domain: State-of-the-art and research issues," *J. Intell. Inf. Syst.*, vol. 57, no. 1, pp. 171–201, Aug. 2021.

[6] R. Lafta, J. Zhang, X. Tao, Y. Li, and V. S. Tseng, "An intelligent recommender system based on short-term risk prediction for heart disease patients," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol. (WI-IAT)*, vol. 3, Dec. 2015, pp. 102–105.

[7] I. Kulev, C. Valk, Y. Lu, and P. Pu, "Recommender system for responsive engagement of senior adults in daily activities," *J. Population Ageing*, vol. 13, no. 2, pp. 167–185, Jun. 2020.

[8] F. Rustam, Z. Imtiaz, A. Mehmood, V. Rupapara, G. S. Choi, S. Din, and I. Ashraf, "Automated disease diagnosis and precaution recommender system using supervised machine learning," *Multimedia Tools Appl.*, vol. 81, no. 22, pp. 31929–31952, Sep. 2022.

[9] G. S. Aujla, A. Jindal, R. Chaudhary, N. Kumar, S. Vashist, N. Sharma, and M. S. Obaidat, "DLRS: Deep learning-based recommender system for smart healthcare ecosystem," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–6.

[10] Q. Zhang, G. Zhang, J. Lu, and D. Wu, "A framework of hybrid recommender system for personalized clinical prescription," in *Proc. 10th Int. Conf. Intell. Syst. Knowl. Eng. (ISKE)*, Nov. 2015, pp. 189–195.

[11] A. Ayad, M. Frei, and A. Schmeink, "Efficient and private ECG classification on the edge using a modified split learning mechanism," in *Proc. IEEE 10th Int. Conf. Healthcare Informat. (ICHI)*, Jun. 2022, pp. 1–6.

[12] R. Bhardwaj and D. Datta, "Development of a recommender system healthmudra using blockchain for prevention of diabetes," in *Recommender System With Machine Learning and Artificial Intelligence: Practical Tools and Applications in Medical, Agricultural and Other Industries*. Hoboken, NJ, USA: Wiley, 2020, pp. 313–327. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119711582.ch16

[13] S. Pinon, S. Jacquet, C. Bulcke, E. Chatzopoulos, X. Lessage, and R. Michel, "Federated health recommender system," in *Proc. 16th Int. Joint Conf. Biomed. Eng. Syst. Technol.*, 2023, pp. 439–444.

[14] A. Peine, A. Hallawa, J. Bickenbach, G. Dartmann, L. B. Fazlic, A. Schmeink, G. Ascheid, C. Thiemermann, A. Schuppert, R. Kindle, L. Celi, G. Marx, and L. Martin, "Development and validation of a reinforcement learning algorithm to dynamically optimize mechanical ventilation in critical care," *NPJ Digit. Med.*, vol. 4, no. 1, pp. 1–12, Feb. 2021.

[15] A. Ayad, A. Hallawa, A. Peine, L. Martin, L. B. Fazlic, G. Dartmann, G. Marx, and A. Schmeink, "Predicting abnormalities in laboratory values of patients in the intensive care unit using different deep learning models: Comparative study," *JMIR Med. Informat.*, vol. 10, no. 8, Aug. 2022, Art. no. e37658.

[16] J. J. Frassica, "Frequency of laboratory test utilization in the intensive care unit and its implications for large-scale data collection efforts," *J. Amer. Med. Inform. Assoc.*, vol. 12, no. 2, pp. 229–233, Nov. 2004.

[17] B. Clouzeau, M. Caujolle, A. San-Miguel, J. Pillot, N. Gazeau, C. Tacaille, V. Dousset, F. Bazin, F. Vargas, G. Hilbert, M. Molimard, D. Gruson, and A. Boyer, "The sustainable impact of an educational approach to improve the appropriateness of laboratory test orders in the ICU," *PLoS ONE*, vol. 14, no. 5, May 2019, Art. no. e0214802.

[18] R. Butler, M. Monsalve, G. W. Thomas, T. Herman, A. M. Segre, P. M. Polgreen, and M. Suneja, "Estimating time physicians and other health care workers spend with patients in an intensive care unit using a sensor network," *Amer. J. Med.*, vol. 131, no. 8, pp. 972.e9–972.e15, Aug. 2018.

[19] K. Chang, P. Singh, P. Vepakomma, M. G. Poirot, R. Raskar, D. L. Rubin, and J. Kalpathy-Cramer, "Privacy-preserving collaborative deep learning methods for multiinstitutional training without sharing patient data," in *Artificial Intelligence in Medicine*. Amsterdam, The Netherlands: Elsevier, 2021, pp. 101–112.

[20] A. Ayad, M. Renner, and A. Schmeink, "Improving the communication and computation efficiency of split learning for IoT applications," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2021, pp. 01–06.

[21] *ACP Reference Ranges*. Accessed: May 5, 2022. [Online]. Available: https://annualmeeting.acponline.org/educational-program/handouts/reference-ranges-table
[22] *International Classification of Diseases: Ninth Revision, Basic Tabulation List With Alphabetic Index*, World Health Org., Geneva, Switzerland, 1978, p. 331.
[23] R. Kavuluru, A. Rios, and Y. Lu, "An empirical evaluation of supervised learning approaches in assigning diagnosis codes to electronic medical records," *Artif. Intell. Med.*, vol. 65, no. 2, pp. 155–166, Oct. 2015.
[24] P. Xie and E. Xing, "A neural architecture for automated ICD coding," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 1066–1076.
[25] F. Li and H. Yu, "ICD coding from clinical text using multi-filter residual convolutional neural network," in *Proc. AAAI Conf. Artif. Intell.*, 2020, vol. 34, no. 5, pp. 8180–8187.
[26] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.
[27] M.-T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," 2015, *arXiv:1508.04025*.
[28] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein, "Explainable prediction of medical codes from clinical text," 2018, *arXiv:1802.05695*.
[29] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. Conf. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.
[30] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, "BioBERT: A pre-trained biomedical language representation model for biomedical text mining," *Bioinformatics*, vol. 36, no. 4, pp. 1234–1240, Feb. 2020.
[31] X. Yu, W. Hu, S. Lu, X. Sun, and Z. Yuan, "BioBERT based named entity recognition in electronic medical record," in *Proc. 10th Int. Conf. Inf. Technol. Med. Educ. (ITME)*, Aug. 2019, pp. 49–52.
[32] E. W. Forgy, "Cluster analysis of multivariate data: Efficiency versus interpretability of classification," *Biometrics*, vol. 21, no. 3, pp. 768–769, 1965.
[33] Y. Gao, M. Kim, S. Abuadbba, Y. Kim, C. Thapa, K. Kim, S. A. Camtepe, H. Kim, and S. Nepal, "End-to-end evaluation of federated learning and split learning for Internet of Things," 2020, *arXiv:2003.13376*.
[34] P. Vepakomma, T. Swedish, R. Raskar, O. Gupta, and A. Dubey, "No peek: A survey of private distributed deep learning," 2018, *arXiv:1812.03288*.
[35] A. E. W. Johnson, T. J. Pollard, L. Shen, L.-W.-H. Lehman, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. Anthony Celi, and R. G. Mark, "MIMIC-III, a freely accessible critical care database," *Sci. Data*, vol. 3, no. 1, May 2016, Art. no. 160035.
[36] T. J. Pollard, A. E. W. Johnson, J. D. Raffa, L. A. Celi, R. G. Mark, and O. Badawi, "The eICU collaborative research database, a freely available multi-center database for critical care research," *Sci. Data*, vol. 5, no. 1, pp. 1–13, Sep. 2018.
[37] J. Mullenbach, S. Wiegreffe, J. Duke, J. Sun, and J. Eisenstein, "Explainable prediction of medical codes from clinical text," in *Proc. NAACL*, 2018, pp. 1101–1111.
[38] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," 2013, *arXiv:1301.3781*.
[39] D. L. Wheeler et al., "Database resources of the national center for biotechnology information," *Nucleic Acids Res.*, vol. 33, pp. D39–D45, Dec. 2004.
[40] X. Wang, Y. Zhang, X. Ren, Y. Zhang, M. Zitnik, J. Shang, C. Langlotz, and J. Han, "Cross-type biomedical named entity recognition with deep multi-task learning," 2018, *arXiv:1801.09851*.
[41] T. Wolf et al., "HuggingFace's transformers: State-of-the-art natural language processing," 2019, *arXiv:1910.03771*.
[42] N. Craswell, *Precision at N*. Cham, Switzerland: Springer, 2009, pp. 2127–2128.
[43] D. M. Saputra, D. Saputra, and L. D. Oswari, "Effect of distance metrics in determining K-value in K-means clustering using elbow and silhouette method," in *Proc. Sriwijaya Int. Conf. Inf. Technol. Appl. (SICONIAN)*, 2020, pp. 341–346.
[44] M. A. Syakur, B. K. Khotimah, E. M. S. Rochman, and B. D. Satoto, "Integration K-means clustering method and elbow method for identification of the best customer profile cluster," *IOP Conf. Ser., Mater. Sci. Eng.*, vol. 336, Apr. 2018, Art. no. 012017.
[45] R. Sammouda and A. El-Zaart, "An optimized approach for prostate image segmentation using K-means clustering algorithm with elbow method," *Comput. Intell. Neurosci.*, vol. 2021, pp. 1–13, Nov. 2021.
[46] A. Holzinger, G. Langs, H. Denk, K. Zatloukal, and H. Müller, "Causability and explainability of artificial intelligence in medicine," *WIREs Data Mining Knowl. Discovery*, vol. 9, no. 4, p. e1312, Jul. 2019.
[47] A. Ayad, H. Yu, and A. Schmeink. (2023). *Smart Summary*. [Online]. Available: https://github.com/a-ayad/smartsummary

AHMAD AYAD (Member, IEEE) received the B.S. degree in electrical engineering from The University of Jordan, Jordan, in 2015, and the M.S. degree in communications engineering from RWTH Aachen University, Germany, in 2019, where he is currently pursuing the Ph.D. degree in electrical engineering. His research interests include distributed machine learning and artificial intelligence applications in medicine.

YU-HSUAN TAI received the B.S. degree in electrical engineering from National Taiwan Normal University, Taiwan, in 2018, and the M.S. degree in electrical and computer engineering from RWTH Aachen University, Germany, in 2022. His research interests include machine learning and applications of artificial intelligence in medicine.

GUIDO DARTMANN (Senior Member, IEEE) received the Diploma and Dr.-Ing. degrees from RWTH Aachen University, in 2007 and 2013, respectively. Since 2016, he has been a Professor of distributed systems with Umwelt-Campus Birkenfeld. He is currently a member of the Institute for Software Systems (ISS) Board of Directors. He is the coauthor of more than 100 publications and an Editor of the books *Big Data Analytics for Cyber-Physical Systems: Machine Learning for the Internet of Things* and *Smart Transportation: AI Enabled Mobility and Autonomous Driving*. His research interests include distributed systems, artificial intelligence, cyber-physical systems, and machine learning.

ANKE SCHMEINK (Senior Member, IEEE) received the Diploma degree in mathematics with a minor in medicine and the Dr.-Ing. degree in electrical engineering and information technology from RWTH Aachen University, Germany, in 2002 and 2006, respectively. She was a Research Scientist with Philips Research before joining RWTH Aachen University. She spent several research visits with The University of Melbourne and the University of York. She is currently leading the Chair of Information Theory and Data Analytics, RWTH Aachen University, Germany. She is the coauthor of more than 270 publications and an Editor of the books *Big Data Analytics for Cyber-Physical Systems: Machine Learning for the Internet of Things* and *Smart Transportation: AI Enabled Mobility and Autonomous Driving*. Her research interests include information theory, machine learning, data analytics, and optimization, focusing on wireless communications and medical applications.