Journal of the American Medical Informatics Association, 27(9), 2020, 1393–1400
doi: 10.1093/jamia/ocaa083
Advance Access Publication Date: 8 July 2020
Research and Applications

Research and Applications

Piloting a model-to-data approach to enable predictive analytics in health care through patient mortality prediction

Timothy Bergquist[1,*], Yao Yan[2,*], Thomas Schaffter[3], Thomas Yu[3], Vikas Pejaver[1], Noah Hammarlund[1], Justin Prosser[4], Justin Guinney[1,3], and Sean Mooney[1]

[1]Biomedical Informatics and Medical Education, University of Washington, Seattle, Washington, USA, [2]Molecular Engineering and Sciences Institute, University of Washington, Seattle, Washington, USA, [3]Sage Bionetworks, Seattle, Washington, USA[4]Institute for Translational Health Sciences, University of Washington, Seattle, Washington, USA

*These authors contributed equally.

Corresponding Author: Sean Mooney, PhD, Biomedical Informatics and Medical Education, University of Washington, Seattle, WA 98195, USA; sdmooney@uw.edu

Received 11 December 2019; Revised 16 April 2020; Editorial Decision 20 April 2020; Accepted 6 May 2020

ABSTRACT

Objective: The development of predictive models for clinical application requires the availability of electronic health record (EHR) data, which is complicated by patient privacy concerns. We showcase the "Model to Data" (MTD) approach as a new mechanism to make private clinical data available for the development of predictive models. Under this framework, we eliminate researchers' direct interaction with patient data by delivering containerized models to the EHR data.
Materials and Methods: We operationalize the MTD framework using the Synapse collaboration platform and an on-premises secure computing environment at the University of Washington hosting EHR data. Containerized mortality prediction models developed by a model developer, were delivered to the University of Washington via Synapse, where the models were trained and evaluated. Model performance metrics were returned to the model developer.
Results: The model developer was able to develop 3 mortality prediction models under the MTD framework using simple demographic features (area under the receiver-operating characteristic curve [AUROC], 0.693), demographics and 5 common chronic diseases (AUROC, 0.861), and the 1000 most common features from the EHR's condition/procedure/drug domains (AUROC, 0.921).
Discussion: We demonstrate the feasibility of the MTD framework to facilitate the development of predictive models on private EHR data, enabled by common data models and containerization software. We identify challenges that both the model developer and the health system information technology group encountered and propose future efforts to improve implementation.
Conclusions: The MTD framework lowers the barrier of access to EHR data and can accelerate the development and evaluation of clinical prediction models.

Key words: electronic health records, clinical informatics, data sharing, privacy, data science

INTRODUCTION

Electronic health records and the future of data-driven health care

Healthcare providers substantially increased their use of electronic health record (EHR) systems in the past decade.[1] While the primary drivers of EHR adoption were the 2009 Health Information Technology for Economic and Clinical Health Act and the data exchange capabilities of EHRs,[2] secondary use of EHR data to improve clinical decision support and healthcare quality also contributed to large-scale adoption.[3] EHRs contain a rich set of information about patients and their health history, including doctors' notes, medications prescribed, and billing codes.[4] The prevalence of EHR systems in hospitals enables the accumulation and utilization of large clinical data to address specific clinical questions. Given the size and complexity of these data, machine learning approaches provide insights in a more automated and scalable manner.[5,6] Healthcare providers have already begun to implement predictive analytics solutions to optimize patient care, including models for 30-day readmissions, mortality, and sepsis.[7] As hospitals improve data capture quality and quantity, opportunities for more granular and impactful prediction questions will become more prevalent.

Hurdles to clinical data access

Healthcare institutions face the challenge of balancing patient privacy and EHR data utilization.[8] Regulatory policies such as Health Insurance Portability and Accountability Act and Health Information Technology for Economic and Clinical Health Act place the onus and financial burden of ensuring the security and privacy of the patient records on the healthcare institutions hosting the data. A consequence of these regulations is the difficulty of sharing clinical data within the research community. Research collaborations are often bound by highly restrictive data use agreements or business associate agreements limiting the scope, duration, quantities, and types of EHR data that can be shared.[9] This friction has slowed, if not impeded, researchers' abilities to build and test clinical models.[9] While these data host-researcher relationships are important and lead to impactful collaborations, they are often limited to intrainstitution collaborations, relegating many researchers with no healthcare institution connections to smaller public datasets or inferior synthetic data. One exception to this is the patient-level prediction working group in the Observational Health Data Sciences and Informatics community, which developed a framework for building and externally validating machine learning models.[10] While the PLP group has successfully streamlined the process to externally validate model performance, there is still an assumption that the model developers have direct access to an EHR dataset that conforms to the Observational Medical Outcomes Partnerships (OMOP) Common Data Model (CDM),[11,12] on which they can develop their models. In order to support model building and testing more widely in the research community, new governance models and technological systems are needed to minimize the risk of reidentification of patients, while maximizing the ease of access and use of the clinical data.

Methods for sharing clinical data

De-identification of EHR data and the generation of synthetic EHR data are 2 solutions to enable clinical data sharing. De-identification methods focus on removing or obfuscating the 18 identifiers that make up the protected health information as defined by the Health Insurance Portability and Accountability Act.[13] De-identification reduces the risk of information leakage but may still leave a unique fingerprint of information that is susceptible to reidentification.[13,14] De-identified datasets like MIMIC-III are available for research and have led to innovative research studies.[15–17] However, these datasets are either limited in size (MIMIC-III [Medical Information Mart for Intensive Care-III] only includes 38 597 distinct adult patients and 49 785 hospital admissions), scope (MIMIC-III is specific to intensive care unit patients), and availability (data use agreements are required to use MIMIC-III).

Generated synthetic data attempt to preserve the structure, format, and distributions of real EHR datasets but do not contain identifiable information about real patients.[18] Synthetic data generators, such as medGAN,[16] can generate EHR datasets consisting of high-dimensional discrete variables (both binary and count features), although the temporal information of each EHR entry is not maintained. Methods such as OSIM2 are able to maintain the temporal information but only simulate a subset of the data specific to a use-case (eg, drug and treatment effects).[19] Synthea uses publicly available data to generate synthetic EHR data but is limited to the 10 most common reasons for primary care encounters and 10 chronic diseases that have the highest morbidity in the United States.[20] To our knowledge, no existing method can generate an entire synthetic repository while preserving complete longitudinal and correlational aspects of all features from the original clinical repository.

"Model to data" framework

The "Model to Data" (MTD) framework, a method designed to allow machine learning research on private biomedical data, was described by Guinney et al[21] as an alternative to traditional data sharing methods.The focus of MTD is to enable the development of analytic tools and predictive models without granting researchers direct, physical access to the data. Instead, a researcher sends a containerized model to the data hosts who are then responsible for running the model on the researcher's behalf. In contrast to the methods previously described, in which the shared or synthetic data were limited in both scope and size, an MTD approach grants a researcher the ability to use all available data from identified datasets, even as those data stay at the host sites, while not giving direct access to the researcher. This strategy enables the protection of confidential data while allowing researchers to leverage complete clinical datasets. The MTD framework relies on modern containerization software such as Docker[22] or Singularity[23] for model portability, which serves as a "vehicle," sending models designed by a model developer to a secure, isolated, and controlled computing environment where it can be executed on sensitive data. The use of containerization software not only facilitates the secure delivery and execution of models, but it opens up the ability for integration into cloud environments (eg, Amazon Web Services, Google Cloud) for cost-effective and scalable data analysis.

The MTD approach has been successful in a series of recent community challenges but has not yet been shown to work with large, EHR datasets.[24] Here, we present a pilot study of an MTD framework implementation enabling the intake and ingestion of containerized clinical prediction models by a large healthcare institution (the University of Washington health system, UW Medicine) to their on-premises secure computing infrastructure. The main goals of this pilot are to demonstrate (1) the operationalization of the MTD approach within a large health system, (2) the ability of the MTD framework to facilitate predictive model development by a researcher (here referred to as the model developer) who does not

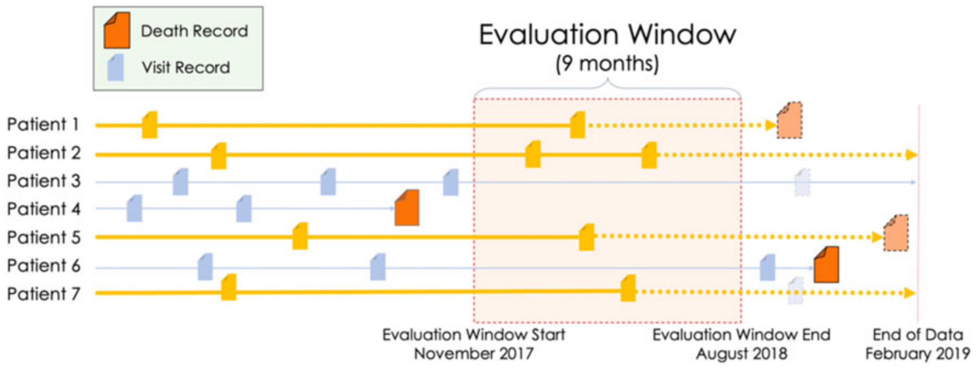

Figure 1. Defining the evaluation dataset. Any patient with at least 1 visit within the evaluation window was included in the evaluation dataset (gold). All other patient records were added to the training dataset (blue). Visits that were after the evaluation window end were excluded from the evaluation dataset and from the training dataset for patients who did not have a confirmed death (light/transparent blue). A 9-month evaluation window was chosen as the timeframe as that resulted in an 80/20 split between the training dataset and the evaluation dataset.

have direct access to UW Medicine EHR data, and (3) the feasibility of a MTD community challenge for evaluating clinical algorithms on remotely stored and protected patient data.

MATERIALS AND METHODS

Pilot data description

The UW Medicine enterprise data warehouse (EDW) includes patient records from medical sites across the UW Medicine system including the University of Washington Medical Center, Harborview Medical Center, and Northwest Hospital and Medical Center. The EDW gathers data from over 60 sources across these institutions including laboratory results, microbiology reports, demographic data, diagnosis codes, and reported allergies. An analytics team at the University of Washington transformed the patient records from 2010 to the present day into a standardized data format, OMOP CDM v5.0. For this pilot study, we selected all patients who had at least 1 visit in the UW OMOP repository, which represented 1.3 million patients, 22 million visits, 33 million procedures, 5 million drug exposure records, 48 million condition records, 10 million observations, and 221 million measurements.

Scientific question for the pilot of the "model to data" approach

For this MTD demonstration, the scientific question we asked the model developer to address was the following: Given the past electronic health records of each patient, predict the likelihood that he/she will pass away within the next 180 days following his/her last visit. Patients who had a death record and whose last visit records were within 180 days of the death date were defined as positives. Negatives were defined as patients whose death records were more than 180 days away from the last visit or who did not have a death record and whose last visit was at least 180 days prior to the end of the available data.

We selected all-cause mortality as the scientific question due to the abundance and availability of patient outcomes from the Washington state death registry. As UW has linked patient records with state death records, the gold standard benchmarks are not constrained to events happening within the clinic. Moreover, the mortality prediction question has been thoroughly studied.[25–27] For these reasons, patient mortality prediction represents a well-defined proof-of-concept study to showcase the potential of the MTD evaluation platform.

Defining the training and evaluation datasets

For the purpose of this study, we split the data into 2 sets: the training and evaluation datasets. In a live healthcare setting, EHR data is constantly changing and evolving along with clinical practice, and prospective evaluation of predictive models is important to ensure that the clinical decision support recommendations generated from model predictions are robust to these changes. We defined the evaluation dataset as patients who had more recently visited the clinic prior to our last death record and the training dataset as all the other patients. This way the longitudinal properties of the data would be approximately maintained.

The last death record in the available UW OMOP repository at the time of this study was February 24, 2019. Any record or measurement that was found after this date was excluded from the pilot dataset and this date was defined as "end of data." When building the evaluation dataset, we considered the date 180 days prior to the end of data (August 24, 2018) the end of the "evaluation window" and the beginning of the evaluation window to be 9 months prior to the evaluation window start (November 24, 2017). We chose a 9-month evaluation window size because this resulted in an 80/20 split between the training and evaluation datasets. We defined the evaluation window as the period of time in which, if a patient had a visit, we included that patient and all their records in the evaluation dataset. Patients who had visits outside the window, but none within the window, were included in the training data. Visit records that fell after the evaluation window end were removed from the evaluation dataset (Figure 1, patient 7) and from the training dataset for patients who did not have a confirmed death (Figure 1, patient 3). We only defined the true positives for the evaluation dataset and created a gold standard of these patients' mortality status based on their last visit date and the death table. However, we gave the model developer the flexibility to select prediction dates for patients in the training dataset and to create corresponding true positives and true negatives for training purposes. See the Supplementary Appendix for additional information.

Model evaluation pipeline

Docker containerized models

Docker is a tool designed to facilitate the sharing of software and dependencies in a single unit called an image.[22] These images make

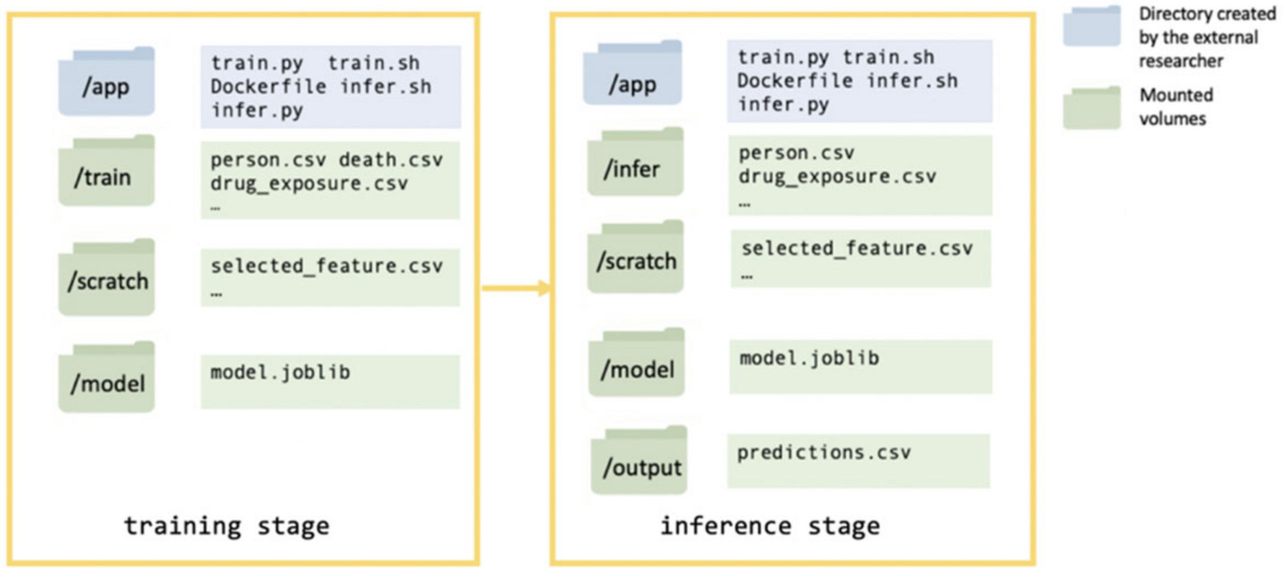

Figure 2. Schema showing the Docker container structure for the training stage and inference stage of running the Docker image.

package dependency, language compilation, and environmental variables easier to manage. This technology enables the simulation of an operating system that can be run on any computer that has the Docker engine or compatible container runtime installed. These containers can also be completely isolated from the Internet or the server on which they are hosted, an important feature when bringing unknown codes to process protected data. For this study, the model developer built mortality prediction Docker images, which included dependencies and instructions for running models in the Docker container.

Synapse collaboration platform

Synapse is an open-source software platform developed by Sage Bionetworks (Seattle, WA) for researchers to share data, compare and communicate their methodologies, and seek collaboration.[28] Synapse is composed of a set of shared REST (representational state transfer)-based web services that support both a website to facilitate collaboration among scientific teams and integration with analysis tools and programming languages to allow computational interactions.[29]The Synapse platform provides services that enable submissions of files or Docker images to an evaluation queue, which have previously been used to manage containerized models submitted to DREAM challenges.[28] We use an evaluation queue to manage the model developer's Docker image submissions.

Submission processing pipeline

To manage the Docker images submitted to the Synapse Collaboration Platform, we used a Common Workflow Language (CWL) pipeline, developed at Sage Bionetworks. The CWL pipeline monitors an evaluation queue on Synapse for new submissions, automatically downloading and running the docker image when the submission is detected. Executed commands are isolated from network access by Docker containers run on UW servers.

UW on-premises server infrastructure

We installed this workflow pipeline in a UW Medicine environment running Docker v1.13.1. UW Research Information Technology uses CentOS 7 (Red Hat Linux) for their platforms. The OMOP data were stored in this environment and were completely isolated behind UW's firewalls. The workflow pipeline was configured to run up to 4 models in parallel. Each model had access to 70 GB of RAM, 4 vCPUs, and 50 GB of SSD.

Institutional review board considerations

We received an institutional review board (IRB) nonhuman subjects research designation from the University of Washington Human Subjects Research Division to construct a dataset derived from all patient records from the EDW that had been converted to the OMOP v5.0 Common Data Model (institutional review board number: STUDY00002532). Data were extracted by an honest broker, the UW Medicine Research IT data services team, and no patient identifiers were available to the research team. The model developer had no access to the UW data.

RESULTS

Model development, submission, and evaluation

For this demonstration, a model developer built a dockerized mortality prediction model. The model developer was a graduate student from the University of Washington who did not have access to the UW OMOP clinical repository. This model was first tested on a synthetic dataset (SynPUF),[30] by the model developer to ensure that the model did not fail when accessing data, training, and making predictions. The model developer submitted the model as a Docker image to Synapse, via a designated evaluation queue, in which the Docker image was uploaded to a secure Docker Hub cloud storage service managed by Sage Bionetworks. The CWL pipeline at the UW secure environment detected this submission and pulled the image into the UW computing environment. Once in the secure environment, the pipeline verified, built, and ran the image through 2 stages, the training and inference stages. During the training stage, a model was trained and saved to the mounted volume "model" and during the inference stage a "predictions.csv" file was written to the mounted volume "output" with mortality probability scores (between 0 and 1) for each patients in the evaluation dataset (Figure 2). Each stage

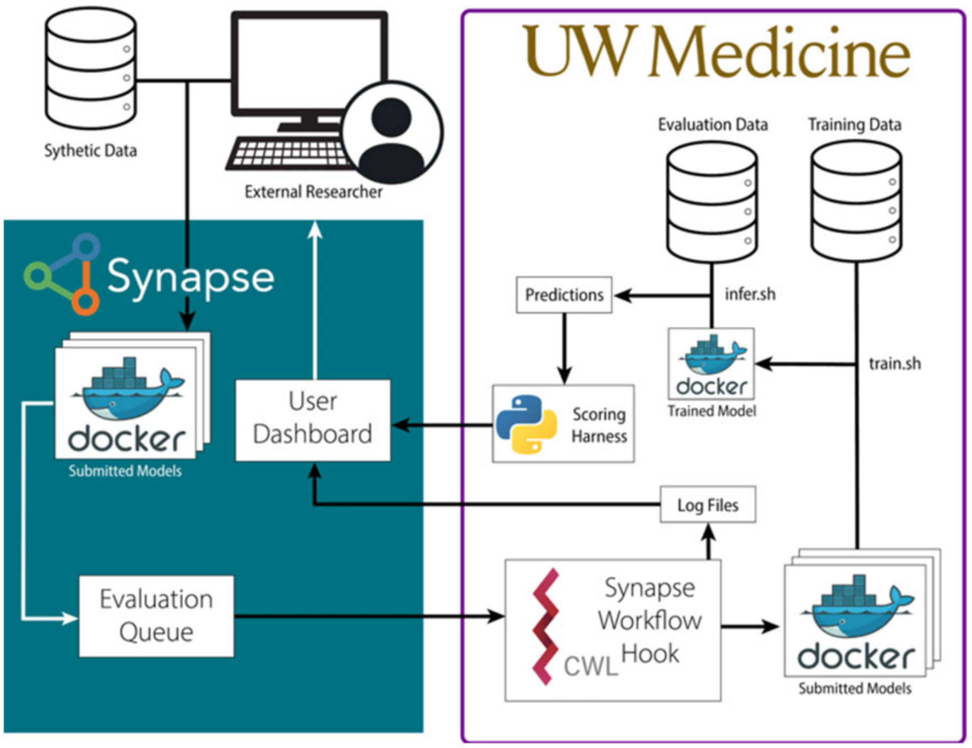

Figure 3. Diagram for submitting and distributing containerized prediction models in a protected environment. Dockerized models were submitted to Synapse by a model developer to an evaluation queue. The Synapse Workflow Hook pulled in the submitted Docker image and built it inside the protected University of Washington (UW) environment. The model trained on the available electronic health record data and then made inferences on the evaluation dataset patients, outputting a prediction file with mortality probability scores for each patient. The prediction file was compared with a gold standard benchmark. The model's performance, measured by area under the receiver-operating characteristic curve, was returned to the model developer. CWL: Common Workflow Language.

had a mounted volume "scratch" available for storing intermediate files such as selected features (Figure 2). The model developer specified commands and dependencies (eg, python packages) for the 2 stages in the Dockerfile, train.sh, and infer.sh. The training and evaluation datasets were mounted to read-only volumes designated "train" and "infer" (Figure 2).

After checking that the "predictions.csv" file had the proper format and included all the patients in the evaluation dataset, the pipeline generated an area under the receiver-operating characteristic curve (AUROC) score and returned this to the model developer through Synapse. When the Docker model failed, a UW staff member would look into the saved log files to assess the errors. Filtered error messages were sent to the model developer for debugging purposes. See Figure 3 for the full workflow diagram.

Model developer's perspective

The model developer built and submitted models, using 3 sets of features: (1) basic demographic information (age on the last visit date, gender, and race); (2) basic demographic information and binary indicators for 5 common chronic diseases (cancer, heart disease, type 2 diabetes, chronic obstructive pulmonary disease, and stroke)[31]; and (3) the 1000 most common concept_ids selected from the procedure_occurrence, condition_occurrence, and drug_exposure domains in the OMOP dataset. For model 2, the developer used the OMOP vocabulary search engine, Athena ("Athena" n.d.), to identify 404 clinical condition_concept_ids associated with cancer, 76 condition_concept_ids with heart disease, 104 condition_concept_ids with type 2 diabetes, 11 condition_concept_ids with chronic obstructive pulmonary disease, and 153 condition_concept_ids with stroke (Table 1). Logistic regression was used on the 3 sets of features respectively. All model scripts are available online (https://github.com/yy6linda/Jamia_ehr_predictive_model).

Model performance

The submitted models were evaluated at the University of Washington by comparing the output predictions of the models to the true 180-day mortality status of all the patients in the evaluation dataset. The implementation of the logistic regression model, Model 1, using only demographic information, had an AUROC of 0.693. Model 2, using demographic information and 5 common chronic diseases, yielded an AUROC of 0.861. Model 3, using demographic information and the most common 1000 condition/drug/procedure concepts, yielded an AUROC of 0.921 (Figure 4).

Benchmarking the capacity of fixed computing resources for running predictive models

We tested the capability of running models through the pipeline on increasingly large feature sets using 2 machine learning algorithms: logistic regression and neural networks. The models ran under fixed computational resources: 70 GB of RAM and 4 vCPUs (a quarter of the total available UW resources made available for this project). This tested the feasibility of running multiple (here, 4) concurrent, high-performance models on UW infrastructure for a community challenge. A total of 6934 of the features used for this scalability test were selected from condition_concept_ids that have more than 20 occurrences within 360 days from the last visit dates of patients in the training dataset. The 2 selected algorithms were applied to a sub-

Table 1. Number of patients in the University of Washington Medicine Observational Medical Outcomes Partnerships repository who have been diagnosed with cancer, heart disease, type 2 diabetes, or chronic obstructive pulmonary disease

	Training set (n = 956 212)	Evaluation set (336 548)
Patients with cancer	66 203 (6.9)	42 195 (12.5)
Patients with heart disease	31 352 (3.3)	23 108 (6.9)
Patients with type 2 diabetes	40 938 (4.3)	28 234 (8.4)
Patients with chronic obstructive pulmonary disease	13 777 (1.4)	8302 (2.5)
Patients with stroke	5216 (0.6)	3927 (1.2)
Other patients	834 591 (87.3)	257 884 (76.6)

Values are n (%).

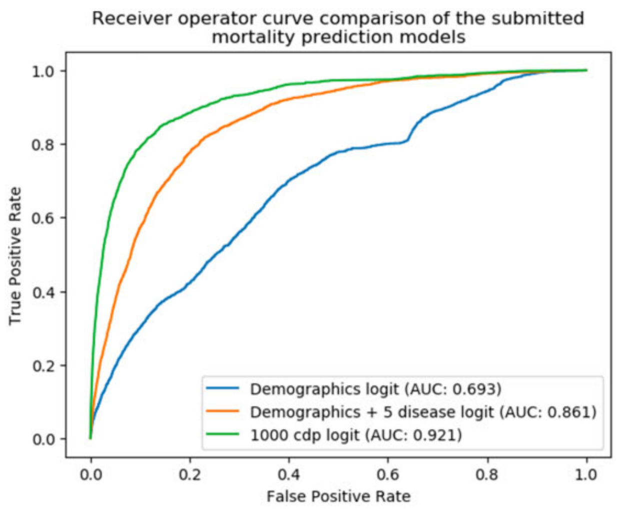

Figure 4. A comparison of the receiver-operating characteristic curves for the 3 mortality prediction models submitted, trained, and evaluated using the "Model to Data" framework. AUC: area under the curve; cdp: condition/procedure/drug.

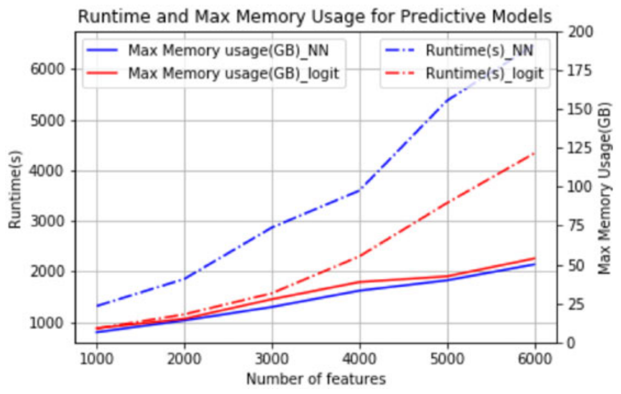

Figure 5. Runtime and max memory usage for training predictive models in the benchmarking test.

set of the features of 1000, 2000, 3000, 4000, 5000, and 6000 selected condition_concept_ids. We used the python sklearn package to build a logistic regression model and keras frameworks to build a 3-layer neural network model (dimension 25 * 12 * 2). For both models, we trained and inferred using the 6 different feature set sizes. We report here the run times and max memory usage (Figure 5). While run times scale linearly with the number of features, maximum memory usage scales in a slightly superlinear fashion.

DISCUSSION

In this pilot project, we implemented the MTD framework in the context of an institutional enterprise data warehouse and demonstrated how a model developer can develop clinical predictive models without having direct access to patient data. This MTD evaluation platform relied on a mutually agreed-upon set of expectations between the data-hosting institution and the model developer, including the use of (1) a common data model (here, OMOP), (2) a standard containerization platform (here, Docker), (3) predetermined input and output file formats, (4) standard evaluation metrics and scripts, and (5) a feedback exchange mechanism (here, Synapse). While we focused on the specific task of mortality status prediction in this pilot study, our platform would naturally be generalizable to other prediction questions or data models. A well-documented common data model (here, OMOP v5.0) is essential to the successful operation of the MTD approach. This framework, however, is not limited to the designated OMOP version, nor the OMOP CDM, and could be expanded to the PCORnet Common Data Model,[32] i2b2,[33] or any other clinical data model. The focus of the MTD framework is to deliver containerized algorithms to private data, of any standardized form, without exposing the data. With increased computational resources, our platform could scale up to handle submissions of multiple prediction models from multiple researchers. Our scalability tests show that complex models on wide feature sets can be trained and evaluated in this framework even with limited resources (70 GB per submission). These resources, including more RAM, CPUs, and GPUs, could be expanded in a cloud environment and parallelized across multiple models. This scalability makes the MTD approach particularly appealing in certain contexts as discussed in the following sections.

MTD as a mechanism to standardize sharing, testing, and evaluation of clinical prediction models

Typically, most clinical prediction models have been developed and evaluated in isolation on site-specific or network-specific datasets, without additional validation on external health record data from other sites.[27] By implementing an evaluation platform for common clinical prediction problems, it would be possible to compare the performance of models implementing different algorithms on the same data and to test the robustness of the same model across different sites, assuming those sites are using the same common data model. This framework also motivates researchers to containerize models for future reproduction. In the long term, we envision that the MTD approach will enable researchers to test their predictive models on protected health data without worrying about identification of patients and to inspire the ubiquitous use of dockerized containers as a standard means to deliver and customize predictive models across institutions. The proposed pipeline is not dependent on the clinical question under investigation nor whether the study question involves all steps of model development (training, inference/test, open-ended prospective, and non-performance-related evaluation).

MTD as a mechanism for enabling community challenges

Community challenges are a successful research model where groups of researchers develop and apply their prediction models in response to a challenge question(s), for which the gold standard truth is known only to the challenge organizers. There have been a large number of successful biomedical community challenges including DREAM,[28] CAFA,[34,35] CAGI,[36] and CASP.[37] A key feature of some of these challenges is the prospective evaluation of prediction models, an often unmet need in clinical applications. The MTD approach uniquely enables such an evaluation on live EDW data. Based on our observations in this pilot study, we will scale up our platform to initiate an EHR mortality community challenge at the next stage, in which participants from different backgrounds will join us in developing mortality prediction models.

Lessons learned and limitations

During the iterative process of model training and feedback exchange with the model developer, we discovered issues that will have to be addressed in future implementations. First, the model developer had multiple failed submissions due to discrepancies between the synthetic data and real data. Devoting effort toward improving the similarity between the synthetic data and the UW data will help alleviate this barrier. Correcting differences in data type, column names, and concept availability would allow model developers to catch common bugs early in the development process. Second, providing manually filtered log files (filtered by UW staff) that are generated by the submitted models when running on UW data as an iterative process can be cumbersome. We propose that prior to running submitted models on the UW data, models should first be run on the synthetic dataset hosted in an identical remote environment that would allow the return of all log files to support debugging. This would allow any major errors or bugs to be caught prior to the model running on the real data. Third, inefficiently written prediction models and their containers burdened servers and system administrators. The root cause of this issue was the model developer's difficulty in estimating the computing resources (RAM, CPU) and time needed to run the submitted models. We can use the same synthetic data environment as solution 2 to estimate run time and RAM usage on the full dataset prior to running the model on the real data. Fourth, the model developer was unaware of the data distributions or even the terminologies for certain variables making feature engineering difficult. Making a data dictionary with the more commonly used concept codes from the UW data available to the model developer will enable smarter feature engineering.

The presented pilot predictive models are relatively simple. However, the MTD framework is also compatible with more complicated machine learning algorithms and feature engineering. Future researchers can dockerize their complicated predictive models with more advanced feature engineering and send them through our pipeline as docker images. Our pipeline is able to execute these docker images on real data and return scores. Model interpretation, such as feature importance scores, is also feasible under this framework if the feature importance calculation is embedded in the docker models and output to a designated directory in the docker container. After checks for information leakage, the UW IT would be able to share that information for the model developer to further interpret their models. However, the remote nature of the MTD framework limits the opportunities for manual hyperparameter tuning which usually requires direct interaction with data. Hyperparameters are model parameters predefined before the models' training stages (eg, learning rate, number of layers in neural networks, etc.). However, automated methods to tune the hyperparameters work with the proposed pipeline. The emergence of AutoML, as well as other algorithms including grid and random search, reinforcement learning, evolutionary algorithms, and Bayesian optimization, allows hyperparameter optimization to be automated and efficient.[38]

CONCLUSION

We demonstrate the potential impact of the MTD framework to bring clinical predictive models to private data by operationalizing this framework to enable a model developer to build mortality prediction models using protected UW Medicine EHR data without gaining access to the dataset or the clinical environment. This work serves as a demonstration of the MTD approach in a real-world clinical analytics environment. We believe this enables future predictive analytics sandboxing activities and the development of new clinical predictive methods safely. We are extending this work to enable the EHR DREAM Challenge: Patient Mortality Prediction as a further demonstration.

FUNDING

This work was supported by the Clinical and Translational Science Awards Program National Center for Data to Health funding by the National Center for Advancing Translational Sciences at the National Institutes of Health (grant numbers U24TR002306 and UL1 TR002319). Any opinions expressed in this document are those of the Center for Data to Health community and the Institute for Translational Health Sciences and do not necessarily reflect the views of National Center for Advancing Translational Sciences, team members, or affiliated organizations and institutions. TB, YY, SM, JG, TY, TS, and JP were supported by grant number U24TR002306. TB, JP, and SM were supported by grant number UL1 TR002319. VP is supported by the Washington Research Foundation Fund for Innovation in Data-Intensive Discovery and the Moore/Sloan Data Science Environments Project at the University of Washington and the National Institutes of Health grant number K99 LM012992.

AUTHOR CONTRIBUTIONS

TB managed and curated the Observational Medical Outcomes Partnerships repository, implemented the "Model to Data" (MTD) framework, and was a major contributor in writing the paper. YY built the mortality prediction models, tested the MTD framework, and was a major contributor in writing the paper. TS was a major contributor to the design and management of the study and in writing the paper. VP and NH were contributors in writing the paper. JP and TY were contributors in maintaining and implementing the MTD pipeline and were contributors in writing the paper. SM and JG conceived of the project with TB and YY as well as funding and overseeing scientific progress. All authors read and approved the final paper.

SUPPLEMENTARY MATERIAL

Supplementary material is available at *Journal of the American Medical Informatics Association* online.

ACKNOWLEDGMENTS

We would like to thank Drs. Gang Luo, Kari Stephens, Martin Gunn, Aaron Lee, Meliha Yetisgen, and Su-In Lee for their advice and efforts in planning this project.

CONFLICT OF INTEREST STATEMENT

The authors have no competing interest to declare.

REFERENCE LIST

1. Charles D, Gabriel M, Searcy T, et al. Adoption of electronic health record systems among US non-federal acute care hospitals: 2008–2014. ONC data brief 2015; 23. https://www.healthit.gov/sites/default/files/data-brief/2014HospitalAdoptionDataBrief.pdf Accessed September 05, 2018.
2. Heisey-Grove D, Patel V. *Physician Motivations for Adoption of Electronic Health Records*. Washington, DC: Office of the National Coordinator for Health Information Technology; 2014.
3. Birkhead GS, Klompas M, Shah NR. Uses of electronic health records for public health surveillance to advance public health. *Annu Rev Public Health* 2015; 36 (1): 345–59.
4. Jones SS, Rudin RS, Perry T, *et al*. Health information technology: an updated systematic review with a focus on meaningful use. *Ann Intern Med* 2014; 160 (1): 48–54.
5. Xiao C, Choi E, Sun J. Opportunities and challenges in developing deep learning models using electronic health records data: a systematic review. *J Am Med Inform Assoc* 2018; 25 (10): 1419–28.
6. Miotto R, Wang F, Wang S, *et al*. Deep learning for healthcare: review, opportunities and challenges. *Brief Bioinform* 2018; 19 (6): 1236–46.
7. Kaji DA, Zech JR, Kim JS, *et al*. An attention based deep learning model of clinical events in the intensive care unit. *PLoS One* 2019; 14 (2): e0211057.
8. Abouelmehdi K, Beni-Hessane A, Khaloufi H. Big healthcare data: preserving security and privacy. *J Big Data*2018; 5: 1. doi:10.1186/s40537-017-0110-7.
9. Allen C, Des Jardins TR, Heider A, *et al*. Data governance and data sharing agreements for community-wide health information exchange: lessons from the beacon communities. *EGEMS (Wash DC)* 2014; 2 (1): 1057.
10. Reps JM, Schuemie MJ, Suchard MA, *et al*. Design and implementation of a standardized framework to generate and evaluate patient-level prediction models using observational healthcare data. *J Am Med Inform Assoc* 2018; 25: 969–75.
11. Hripcsak G, Duke JD, Shah NH, *et al*. Observational Health Data Sciences and Informatics (OHDSI): opportunities for observational researchers. *Stud Health Technol Inform* 2015; 216: 574–8.
12. Klann JG, Joss MAH, Embree K, *et al*. Data model harmonization for the All Of Us Research Program: Transforming i2b2 data into the OMOP common data model. *PLoS One* 2019; 14 (2): e0212463.
13. Garfinkel SL. *De-Identification of Personal Information*. Gaithersburg, MD: National Institute of Standards and Technology; 2015. doi:10.6028/NIST.IR.8053.
14. Malin B, Sweeney L, Newton E. *Trail re-identification: learning who you are from where you have been*. Pittsburgh, PA: Carnegie Mellon University, Laboratory for International Data Privacy; 2003.
15. Desautels T, Calvert J, Hoffman J, *et al*. Prediction of sepsis in the intensive care unit with minimal electronic health record data: a machine learning approach. *JMIR Med Inform* 2016; 4 (3): e28.
16. Choi E, Biswal S, Malin B, *et al*. Generating multi-label discrete patient records using generative adversarial networks. *arXiv*:1703.06490. 2017.
17. Johnson AEW, Pollard TJ, Shen L, *et al*. MIMIC-III, a freely accessible critical care database. *Sci Data* 2016; 3 (1): 160035.
18. Foraker R, Mann DL, Payne P. Are synthetic data derivatives the future of translational medicine? *J Am Coll Cardiol Basic Trans Science* 2018; 3 (5): 716–8.
19. Murray RE, Ryan PB, Reisinger SJ. Design and validation of a data simulation model for longitudinal healthcare data. *AMIA Annu Symp Proc* 2011; 2011: 1176–85.
20. Walonoski J, Kramer M, Nichols J, *et al*. Synthea: An approach, method, and software mechanism for generating synthetic patients and the synthetic electronic health care record. *J Am Med Inform Assoc* 2018; 25: 230–8.
21. Guinney J, Saez-Rodriguez J. Alternative models for sharing confidential biomedical data. *Nat Biotechnol* 2018; 36 (5): 391–2.
22. Docker. https://www.docker.com/ Accessed August 9, 2019.
23. Sylabs.io. Singularity. https://sylabs.io/ Accessed 18 November, 2019.
24. Ellrott K, Buchanan A, Creason A, *et al*. Reproducible biomedical benchmarking in the cloud: lessons from crowd-sourced data challenges. *Genome Biol* 2019; 20 (1): 195.
25. Ge W, Huh J-W, Park YR, *et al*. An interpretable ICU mortality prediction model based on logistic regression and recurrent neural networks with LSTM units. *AMIA Annu Symp Proc* 2018; 2018: 460–9.
26. Avati A, Jung K, Harman S, *et al*. Improving palliative care with deep learning. *BMC Med Inform Decis Mak* 2018; 18 (Suppl 4): 122.
27. Goldstein BA, Navar AM, Pencina MJ, *et al*. Opportunities and challenges in developing risk prediction models with electronic health records data: a systematic review. *J Am Med Inform Assoc* 2017; 24 (1): 198–208.
28. Saez-Rodriguez J, Costello JC, Friend SH, *et al*. Crowdsourcing biomedical research: leveraging communities as innovation engines. *Nat Rev Genet* 2016; 17 (8): 470–86.
29. Omberg L, Ellrott K, Yuan Y, *et al*. Enabling transparent and collaborative computational analysis of 12 tumor types within The Cancer Genome Atlas. *Nat Genet* 2013; 45 (10): 1121–6.
30. Lambert CG, Amritansh Kumar P. Transforming the 2.33M-patient Medicare synthetic public use files to the OMOP CDMv5: ETL-CMS software and processed data available and feature-complete. Albuquerque, NM: Center for Global Health, University of New Mexico; 2016.
31. Weng SF, Vaz L, Qureshi N, *et al*. Prediction of premature all-cause mortality: A prospective general population cohort study comparing machine-learning and standard epidemiological approaches. *PLoS One* 2019; 14 (3): e0214365.
32. Fleurence RL, Curtis LH, Califf RM, *et al*. Launching PCORnet, a national patient-centered clinical research network. *J Am Med Inform Assoc* 2014; 21 (4): 578–82.
33. Murphy SN, Weber G, Mendis M, *et al*. Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2). *J Am Med Inform Assoc* 2010; 17 (2): 124–30.
34. Radivojac P, Clark WT, Oron TR, *et al*. A large-scale evaluation of computational protein function prediction. *Nat Methods* 2013; 10 (3): 221–7.
35. Jiang Y, Oron TR, Clark WT, *et al*. An expanded evaluation of protein function prediction methods shows an improvement in accuracy. *Genome Biol* 2016; 17 (1): 184.
36. Cai B, Li B, Kiga N, *et al*. Matching phenotypes to whole genomes: Lessons learned from 4 iterations of the personal genome project community challenges. *Hum Mutat* 2017; 38 (9): 1266–76.
37. Moult J. A decade of CASP: progress, bottlenecks and prognosis in protein structure prediction. *Curr Opin Struct Biol* 2005; 15 (3): 285–9.
38. He X, Zhao K, Chu X. AutoML: a survey of the state-of-the-art. *arXiv*:198.00709. 2019.