2021 43rd Annual International Conference of the
IEEE Engineering in Medicine & Biology Society (EMBC)
Oct 31 - Nov 4, 2021. Virtual Conference

Autopopulus: A Novel Framework for Autoencoder Imputation on Large Clinical Datasets

Davina J. Zamanzadeh[1] Panayiotis Petousis[2] Tyler A. Davis[1] Susanne B. Nicholas[3] Keith C. Norris[3] Katherine R. Tuttle[4,5] on behalf of the CURE-CKD Study team Alex A. T. Bui[6] Majid Sarrafzadeh[1]

***Abstract*— The adoption of electronic health records (EHRs) has made patient data increasingly accessible, precipitating the development of various clinical decision support systems and data-driven models to help physicians. However, missing data are common in EHR-derived datasets, which can introduce significant uncertainty, if not invalidating the use of a predictive model. Machine learning (ML)-based imputation methods have shown promise in various domains for the task of estimating values and reducing uncertainty to the point that a predictive model can be employed. We introduce *Autopopulus*, a novel framework that enables the design and evaluation of various autoencoder architectures for efficient imputation on large datasets. Autopopulus implements existing autoencoder methods as well as a new technique that outputs a range of estimated values (rather than point estimates), and demonstrates a workflow that helps users make an informed decision on an appropriate imputation method. To further illustrate Autopopulus' utility, we use it to identify not only which imputation methods can most accurately impute on a large clinical dataset, but to also identify the imputation methods that enable downstream predictive models to achieve the best performance for prediction of chronic kidney disease (CKD) progression.**

***Clinical relevance*— Enable investigation of autoencoders for imputation of large clinical datasets, and investigate the impact of imputation on downstream tasks instead of in isolation.**

I. INTRODUCTION

The widespread adoption of electronic health records (EHRs) has ushered in a new age of data-driven medicine, with a significant number of artificial intelligence (AI)-based methods being explored to provide new insights from the growing number of patient records. However, the variation in healthcare delivery complicates the analysis of such datasets, especially issues like missing data. Most predictive models are unable to handle missing values well, if at all. One way missing data are commonly dealt with is by removing the rows or features with missing data. Removing features with missing data is not always tenable as those features might be important predictive factors. Similarly, dropping observations with missing data poses issues such as: reducing the dataset size significantly, or introducing bias by limiting the dataset to observations with features that are well-populated but may not be randomly distributed. For example, in an EHR dataset, sicker patients require more tests and visits than healthier patients and are less likely to have missing values, so dropping patients with missing values might bias the dataset towards these less healthy patients.

Another common approach to handling missing data is substituting missing values with estimates on the missing values, which is known as *imputing*. While imputation seems more appealing than dropping potentially useful information, using the wrong imputation method may degrade the quality of prediction. To properly impute missing data, it is important that the method impose a reasonable assumption about the missingness mechanism for the data [1]: missing completely at random (MCAR), missing at random (MAR), and missing not at random (MNAR) [1]. Many imputation techniques exist and are frequently used. The simplest forms of imputation are those such as mean or mode, (stochastic) regression, nearest neighbor(s), and carry forward/backward imputation. While these are simple, fast, and easy to implement, they underestimate standard error by reducing variability and ultimately produce biased estimates. Another approach involves maximum likelihood and/or multiple imputation, (such as multiple imputation by chained equations, MICE). While these methods are guaranteed to produce unbiased estimates under MAR, they are computationally expensive and time consuming and will produce biased estimates under MNAR. Existing methods that can model data MNAR in addition to MAR, such as pattern mixture models [2], rely on Monte Carlo methods that are computationally slow.

As an alternative, we explore the use of autoencoders as a tool for imputation on EHR data. One benefit of autoencoders for imputation over the previously mentioned methods is their ability to rapidly impute on large datasets, such as EHR datasets, and quickly learn nonlinear relationships. However, despite the appeal of autoencoders for imputation, there are no theoretical proofs on their behavior. As such, we developed *Autopopulus* to empirically study the potential of autoencoders for imputation. Autopopulus allows us to efficiently compare different autoencoder methods in addition to assisting in identifying the best technique for the given dataset and task at hand, focusing on data MAR and MNAR—the most common scenarios for EHR data. We expand upon prior literature by taking a closer look at how accurately autoencoders can learn to impute an EHR dataset with missing values, and how that imputation affects downstream predictive performance. To demonstrate

[1]Department of Computer Science, University of California, Los Angeles, Los Angeles, CA, United States
[2]Clinical and Translation Science Institute, University of California, Los Angeles, Los Angeles, CA, United States
[3]David Geffen School of Medicine, University of California, Los Angeles, Los Angeles, CA, United States
[4] Providence St. Joseph Health, Providence Medical Research Center, Spokane Washington
[5] University of Washington School of Medicine, Seattle Washington
[6]Medical & Imaging Informatics Group, University of California, Los Angeles, Los Angeles, CA, United States

Autopopulus and our approaches, we use the Center for Kidney Disease Research, Education and Hope (CURE-CKD) dataset [3], [4] to identify individuals at-risk for and with chronic kidney disease (CKD). We implemented several existing autoencoders for imputation, as well as our own approach for imputation that utilizes a completely discretized formulation of the data. We find that different imputation methods perform best under different missingness scenarios, although the overall downstream performance on the prediction task does not vary much for this dataset.

II. Related Work

There are many types of autoencoders. An autoencoder is undercomplete when the code (the output of the encoder) is smaller than the input, acting as lossy compression, and is overcomplete when the code is larger than the input. As opposed to a vanilla (i.e. regular) one, in a variational autoencoder (VAE), the encoder outputs two vectors: one of means and standard deviations, detailing a Normal distribution, rather than a single vector of raw values. Due to this nature, VAEs also require an additional loss term to penalize differences between the learned and true distribution. In a denoising autoencoder (DAE), the inputs are partially corrupted (e.g., set to zero or adding noise). In an imputation context, a DAE would treat missing values as noise.

Variational autoencoders [5] have seen success in imputation on data MNAR and MCAR in myriad domains including traffic forecasting [6], synthetic and simulated milling circuit data MCAR in McCoy et al.'s work [7], and facial image data MAR [8]. Unlike biomedical and EHR data, however, these datasets involve automated data collections systems and can be modeled with clear Gaussian distributions. Camino et al. [9] explored the effectiveness of variational autoencoders across tabular data in different domains such as breast cancer data, credit card data, and optical character recognition data. For high dimensional data, Chen et al. [10] proposed sparse convolutional denoising autoencoders to impute yeast and human genotypic data. They leveraged the added complexity of convolutional layers and a sparse weight matrix to make imputation of high dimensional data tractable. Gondara et al. proposed MIDA (Multiple Imputation using Denoising Autoencoders), an approach that uses an overcomplete denoising autoencoder with the assistance of simple imputation methods such as mean or mode imputation [11]. Beaulieu-Jones et al. proposed denoising autoencoders for the imputation of EHR data on patients diagnosed with Lou Gehrig's disease [12].

While the use of autoencoders for imputation has been previously explored, it is notable that their effect on downstream predictive performance has not been analyzed and the context in which any of these techniques may be optimal is unclear. This observation prompted our development of an open framework that would enable the rapid implementation of different autoencoder imputation methods on a dataset, comparing imputation performance and the sensitivity of a given predictive task relative to inferred missing values.

III. Methodology

With Autopopulus, we aim to provide an extensible framework for developing, testing, and ultimately comparing different autoencoder imputation methods. We show its usage by drawing on the work of McCoy et al., Gondara et al., and Beaulieu-Jones et al. in addition to implementing a novel autoencoder imputation technique. We designed Autopopulus so it can be used in the same fashion as a Scikit-learn [13] imputation model. This lends to easy and intuitive use for future datasets as it interfaces well with widely used tools with minimal overhead. The entire imputer training pipeline is shown in Fig. 1 and we describe key elements below.

a) Data processing: The input to our model is a dataset $\hat{X}$ and label X, which is the unaltered or true version. Autoencoders are not designed to handle missing values by default, so the first step taken is filling in missing values in $\hat{X}$, and also X if the true dataset also has missing values. For example, in McCoy et al. and Beaulieu-Jones et al. this is done by filling in missing values with 0, while in Gondara et al. it is done by a "warm start" with simple imputation.

b) Autoencoder architecture: As mentioned prior, autoencoders come in many flavors. Depending on the nature of the dataset or problem, we must specify a type of autoencoder. For example, McCoy et al. specify a VAE, while Beaulieu-Jones et al. use a DAE.

c) Loss: The reconstruction losses supported by Autopopulus are binary cross-entropy (BCE), mean squared error (MSE), and a combination that applies BCE only to categorical variables and MSE to continuous variables. When the autoencoder is not a VAE, loss is purely reconstruction-based. When the autoencoder is a VAE, we add an extra Kullback-Leibler (KL) divergence error term in addition to the reconstruction loss. For each implemented method we choose the loss employed in their respective paper.

d) Training: For each implemented method we chose the optimizer employed in their respective paper. We tuned the number of layers and nodes per layer, learning rate, L2 penalty, maximum epochs, and early stopping patience on the validation set using an automatic sweep of hyperparameters with asynchronous hyberband scheduling (ASHA) [14].

e) Output: Once loss has been calculated, model output is further processed before computing imputation performance metrics. The same steps are employed when using the model for imputation after training. We apply the sigmoid function only to the categorical variables. Note that this might be slightly different in the experiments if implementing a method from the literature that otherwise applied a sigmoid at the output to all variables whether or not they were categorical, or if there are no categorical variables and are using MSE loss. Lastly, all methods are only used to fill in missing values, so the original values are otherwise kept.

A. Implementation

We implemented the system[1] using Python, Pandas [15], and NumPy [16], and PyTorch-Lightning [17].

[1]The implementation is available at https://github.com/davzaman/autopopulus.

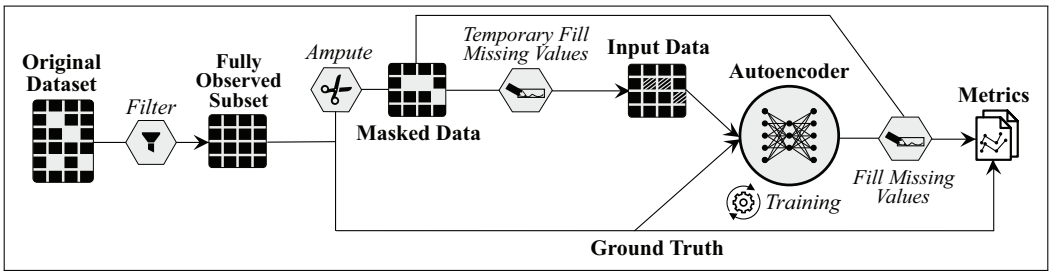

Fig. 1. The data pipeline for training the autoencoder. Data are either initially filled with a constant value such as 0, or data are discretized and a uniform distribution is imposed over missing values. Note if the latter is true, the ground truth is also discretized before being fed to the autoencoder, and the autoencoder output is un-discretized. After training the autoencoder, its output is only used to fill originally missing values.

B. Enabling Experiments to Compare Imputation Methods

Using Autopopulus, we can compare autoencoder-led imputation methods to baseline methods. Autopopulus supports comparing to some of the most popular imputation methods such as mean and mode, k-nearest neighbors, and MICE, and is readily extended to include further techniques.

In addition to allowing imputation on incomplete raw datasets, Autopopulus also supports experiments with controlled missingness scenarios, shown in the first steps of Fig. 1. Via Autopopulus, we can choose which features to ampute (i.e. mask) via which mechanism, and at what percentage. For data MCAR, we replace a value with NaN (not at number) uniformly at random for selected variables. For data MAR we create a cutoff, determined by the percentage of missingness specified, on an observed variable. If a given data entry falls above the cutoff, the value for the chosen missing variable will be NaN, simulating the scenario where the value is missing due to another observed variable. For data MNAR we create lower and upper cutoffs based on the variables that will be missing themselves (e.g., a historically normotensive patience opts to not have their blood pressure measured). If a patient falls inside the cutoff range, the value will be missing, simulating the scenario where the value is missing due to the value itself. Under MAR we can choose which observed features we would like to control the missingness of each of the missing features.

C. Adding New Imputation Methods

Alongside existing autoencoders used for imputation, we used Autopopulus to design a new technique for comparison, showing how the framework can be extended. In this method, the data are discretized (into one-hot features) and a uniform distribution imposed across the discretized variables wherever a value is missing as a data preprocessing step. The goal of the autoencoder is to take the uncertainty, modeled by a uniform probability, for any given missing variable and learn to shift weight onto the discrete bin that most likely represents the true value. For example, if age is discretized into 10 bins and its value is missing, then each bin has a 10% probability of being the "true" bin. We replace the missing value with this uniform probability. The data $\hat{X}$ are passed through the autoencoder model and the result is then compared to the label X according to a given loss function to train the model. We employ minimum description length (MDL) discretization via the Orange package to automatically create bins [18], [19]. MDL discretization is a supervised algorithm that recursively decides the split that minimizes entropy and minimum description length principles as well as the labels assigned to each sample. We chose this discretization method to be able to employ Autopopulus on a wide array of datasets quickly and because it has been used successfully on clinical data in the literature [20]. However, it is possible to use Autopopulus with manual or other means of discretization.

Here, we choose a vanilla autoencoder with BCE loss for the reconstruction error and train using the Adam optimizer [21]. We use BCE instead of mean-squared error because our inputs are either binary due to discretization, or continuous between 0 and 1 after imposing a uniform distribution across bins for a missing variable. Imposing a uniform probability across missing values maximizes entropy for the missing value, which is then penalized by the BCE loss, forcing the autoencoder to focus on correcting for missing values.

In order to allow a more direct comparison to other imputation methods, we first "un-discretize" the output. To un-discretize the data, we map the originally-continuous variables down to the mean of their most likely bin (the one with the highest score). For example, if age was discretized into 10 bins, and the autoencoder yielded the highest score for the range (50,60], then the patient's age would be estimated to be 55 years.

IV. EXPERIMENTAL DESIGN

A. Data

a) CKD cohort: We evaluated Autopopulus using the CURE-CKD Registry [3], [4]. CURE-CKD comprises a comprehensive, real-world, longitudinal dataset of EHR-derived data from two large healthcare systems (Providence St. Joseph Health and University of California, Los Angeles Health; UCLA) over a period of 12 years (2006-2017). Patients included are adults over the age of 18 categorized into one of two cohorts: diagnosed with CKD or flagged as "at-risk" for CKD (diagnosed with diabetes (DM), hypertension (HTN), or pre-DM), based on diagnostic administrative codes, laboratory data, vital signs, and medications. Patients entered and exited the registry that generated the dataset at different points over the 12-year period, resulting in different

history lengths for each patient. For this study, we limited our cohort to patients with CKD only.

The prevailing method for testing for CKD and tracking disease progression is measuring the estimated glomerular filtration rate (eGFR) [22]. Predictive models are being developed to identify individuals who would experience rapid kidney function decline, defined as a 40% decline in eGFR over two years [23]. There are $4,067$ patients in the positive class (experiencing rapid decline), which make up 4.59% of the $88,560$ patients diagnosed with CKD. There are $9,062$ (10.23%) patients that are not missing any data.

b) Variables and Missingness: We define study-entry as the date of the first serum creatinine measurement during the observation period for a patient. We define time-zero entries as the mean of measurements in the first 90 days after the first serum creatinine measurement. We use the study-entry and time-zero entries (eGFR, A1c, systolic blood pressure, and number of ambulatory and inpatient visits) in addition to demographic information (age, sex, ethnicity, and rurality status) and risk factor information (diagnosis of HTN, DM, or pre-DM, and use of angiotensin-converting enzymes inhibitors (ACEIs) and angiotensin receptor inhibitors (ARBs)) to predict rapid kidney function decline in patients in the two years from registry entry. Notably, 74.5% of A1c entries are missing at study-entry and 71.9% at time-zero. 60% of systolic blood pressure entries are missing at study-entry and 60.5% at time-zero.

c) Data Preprocessing: All continuous variables are standardized using a min-max scaler, and all categorical variables are one-hot encoded before being handled by Autopopulus.

B. Experiments

Two sets of experiments were conducted to answer two investigative questions: the ability for various imputation methods to create an accurate representation of the dataset (Fig. 1); and the impact of missingness and imputation on downstream prediction (Fig. 2).

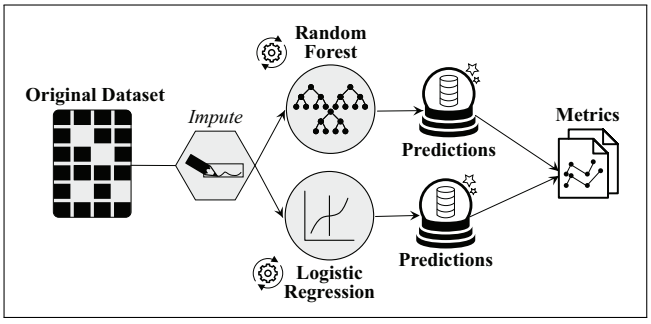

Fig. 2. The data flow for the predictive task. Imputation is done with either baseline methods such as k-nearest neighbors (KNN), or an autoencoder-led method. If using an autoencoder-led method, the autoencoder is trained first and then used to impute the dataset.

In our first set of experiments (Experiment Set 1), we filter the dataset down to the fully observed subset of data and ampute according to different mechanisms and at different missingness rates. In this set of experiments, we test the mean arctangent absolute percent error (MAAPE) and root mean squared error (RMSE) by amputing at both a low and high percentage of missingness (33% and 66%) across all three missingness mechanisms (i.e., MCAR, MAR, and MNAR). These experiments explore an imputation method's ability to create an accurate representation of the data and can control for performance given different levels of missingness and mechanisms. In the second set of experiments (Experiment Set 2), we use the entire dataset "as is" to explore the effect of imputation on predictive performance via the Brier score (calibration), precision-recall area under the curve (PR-AUC), and receiver operating characteristics area under the curve (ROC-AUC) for classifying rapid decline. We used a train, validation, test split of 60%, 20%, and 20% respectively both for training the autoencoder and for training the predictive models.

We selected systolic blood pressure and A1c to be missing for the following settings as they are also missing in the original dataset. For data MAR we use the observed eGFR variable to create a cutoff. Thus, if a patient falls above the cutoff, their systolic blood pressure and A1c will be missing. For data MNAR, if a patient falls inside the cutoff range, their systolic blood pressure and A1c will be missing.

1) Imputation Comparison (Experiment Set 1): We compared MIDA [11]; a denoising autoencoder proposed by Beaulieu-Jones et al. [12] (DAE); a variational autoencoder proposed by McCoy et al. [7] (VAE); and our new method based on a vanilla autoencoder (*APnew*) (Section III-C). Architectures, losses, optimizers, and data preprocessing all match their original descriptions. The baseline methods include simple, k-nearest neighbors (KNN), and MICE imputation. We refer to simple imputation as mean imputation for continuous variables and mode imputation for categorical variables.

2) Predictive Task (Experiment Set 2): Here, our task involves using the imputed data to predict if a patient diagnosed with CKD will experience rapid kidney function decline in the next two years. In order to explore whether linear or non-linear predictive models are better suited for the predictive task, we trained both a logistic regression model and a random forest model on top of the autoencoder outputs. We trained the models to be sensitive to the positive class due to the dataset being highly imbalanced by taking advantage of class weights. Hyperparameters for the logistic regression and random forest models were tuned during validation automatically via Scikit-learn. Each predictive model was trained on 100 stratified bootstrapped samples. The logistic regression and random forest models were implemented via Scikit-learn [13]. We tuned using the Tune framework [24]. The 95% confidence intervals and means for each classification performance metric, such as PR-AUC, were produced from this bootstrapping process.

V. RESULTS

We used Autopopulus to compare autoencoder imputations methods.

A. Experiment Set 1: Imputation Accuracy

We report metrics computed only where the data were originally missing for imputation performance. Values that were not originally missing are kept at the end of the pipeline, so we are largely interested in the performance on originally missing values only. Fig. 3 shows that different imputation methods create more accurate representations of the missing data under different missingness mechanisms, and those patterns remain similar across the amount of data missing. All models achieve lower error under MNAR compared to MAR and MCAR. Under MAR, the RMSE under 33% missing is minutely larger than 66% missing. Under MCAR and MAR, MICE achieves the lowest MAAPE. However, under data MNAR, the VAE proposed by McCoy et al. achieves the lowest error. In general, under MNAR the autoencoder imputation models tend to achieve lower error than the baseline models, aside from APnew. APnew produces a large error across all missingness scenarios.

APnew uniquely discretizes and then undiscretizes the data. Therefore, only for this approach are we interested in the amount of times the autoencoder correctly guesses a bin for an originally continuous feature before undiscretizing. We evaluate this accuracy over the bins for missing values only in Table I. We observe that the autoencoder is fairly accurate under MCAR and MNAR for both percentages of missingness, but struggles more with data MAR.

Fig. 3. Imputation performance captured via MAAPE and RMSE for each imputation method across missingness scenarios, computed only on the missing values. Every column is a different missingness mechanism, and each row is a single imputation metric. The y-axis range differs between MAAPE and RMSE for a closer inspection of the performance under each metric. Note that MAAPE is a percentage reported as a decimal value, while RMSE is a non-negative decimal score.

B. Experiment Set 2: Predictive Performance

Fig. 4 shows that the logistic regression model is in general less calibrated than the random forest model regardless of which imputation method is used. Though the difference is small, the autoencoder-based imputation methods tend to be slightly less calibrated than the baseline methods; however, APnew is the best calibrated autoencoder imputation method.

The imputation method used did not have a large impact on the PR-AUC and ROC-AUC when using a random forest model. But in general, the baseline predictive methods produced better results when using a logistic regression. Markedly, the PR-AUC and ROC-AUC for MIDA almost exactly mirrors simple imputation no matter which predictive model is used, which calls into question whether MIDA offers much advantage beyond simple imputation for the downstream task on this dataset. The remaining three autoencoder-based imputation methods perform similarly across both predictive models. The most calibrated predictive models were trained on data imputed with either MICE or APnew. Across all imputation methods the logistic regression model offers tighter confidence intervals and better Recall but overall poorer performance regarding calibration (Brier score), Precision, PR-AUC, and ROC-AUC. Do note that the Brier score acts as a loss, where a larger value means poorer calibration. MICE imputation produces a very similar PR-AUC whether using a logistic regression or a random forest model for prediction. KNN imputation produces a very similar ROC-AUC whether using a logistic regression or a random forest model for prediction. Another point of interest is the relatively poor PR-AUC across all imputation methods and predictive models. Upon closer inspection, we notice that the precision is extremely poor while recall is more acceptable. Both the logistic regression and random forest struggle with false positives on this dataset.

TABLE I
ACCURACY OVER BINS (ONLY APNEW)

Mechanism	Percent	Accuracy Over Bins
MAR	33.0	0.626
	66.0	0.699
MCAR	33.0	0.804
	66.0	0.842
MNAR	33.0	1.000
	66.0	0.864

C. Discussion

Autopopulus enabled us not only to easily implement multiple autoencoders with a consistent interface and unified training pipeline, but also allowed us to explore their performance on the same dataset and under different missingness scenarios, including MAR and MNAR. Autopopulus enabled the use of data with mixed feature types (continuous and categorical), such as the CURE-CKD dataset, instead of assuming one type. We can identify patterns across missingness mechanisms, on the hypothesis that each mechanism behaves differently. For example, when comparing imputation metrics across methods we saw better imputation performance on data MNAR over data MAR across imputation methods, even though MNAR is famously difficult to handle. In this case, this observation may be due to the steps Autopopulus takes to simulate MNAR, which may be too simplistic compared

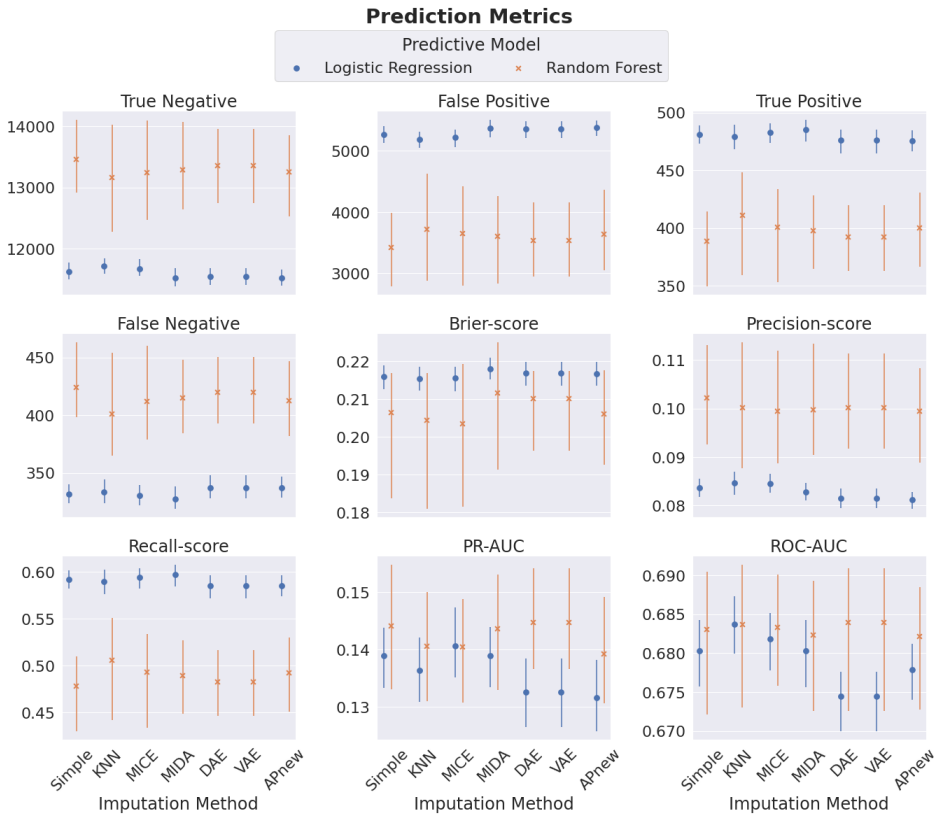

Fig. 4. Average predictive performance for each imputation method on the entire dataset as-is for the predictive models across bootstraps. The true negatives, false positives, true positives, and false negatives are all reported as counts. The remaining metrics are percentages reported as decimal values. Note that the Brier score is in fact a loss, where smaller is better.

to real-world series of events. Autopopulus also allowed us to easily test which imputation methods would perform best in different missingness scenarios. Per prior findings in the literature, under MCAR and MAR, MICE consistently performs best, though sometimes only marginally. However, under MNAR the VAE performed best. For the CURE-CKD dataset, this finding may be due to how we simulate MNAR by specifying lower- and upper-bound cutoff thresholds on the missing variables themselves. The encoder portion of a variational autoencoder produces means and standard deviations, defining a normal distribution for each input. This likely made it "easier" for the VAE to guess that the missing values were in the tails of the distributions, if its inferred means and standard deviations were accurate enough.

Through Autopopulus, we were also able to investigate how the different imputation methods might affect a predictive model's ability to handle class imbalance. In Section IV-A we outlined that only 4.59% of the dataset contains samples from the positive class (experiencing rapid kidney decline). We can use Autopopulus to easily compare each imputation method on the downstream PR-AUC versus ROC-AUC. Though we attempt to deal with class imbalance in the predictors by weighting samples, we do see the imbalance severely affecting performance. It is likely the ROC-AUC is significantly larger than the PR-AUC due to the large number of samples in the negative class (not experiencing rapid kidney decline). Overall, we can see that no one imputation method significantly helped remedy this problem.

Markedly, by comparing methods we identified the imputation method with the consistently largest errors, which turned out to be APnew. This result is to be expected, as the bins produced by automated MDL discretization on this dataset were wide. Though this finding might make us question the utility of APnew, one advantage of APnew is that unlike all of the previous autoencoder-led imputation methods, it provides a *range* of possible values rather than one or more point estimates for imputation. Arguably, point estimates produced by the compared autoencoder methods can be troublesome as they are biased. We can further explore the usefulness of APnew via the experiments with Autopopulus. Despite performing poorly on imputation metrics, models trained on data imputed with APnew perform quite similarly to the alternative methods under many predictive metrics. Table I demonstrates that APnew does a relatively good job of inferring the correct bin for missing values under MCAR and MNAR. We believe this indicates that APnew provides unique insight and can grow into a powerful addition to Autopopulus as we grow this framework.

1) Future Work: Future work includes repeating analysis on an updated version of the dataset that covers a longer follow-up period and is more feature-rich. We also intend to attempt pre-training on various missingness scenarios to

train the model to learn those mechanisms, and then fine-tune the autoencoder imputer model on the dataset or task at hand. We plan to analyze if the techniques differ in strengths depending on characteristics of the data, in particular, data distribution. In order to expand on the novel method, we plan to analyze how discretization bin sizes affect the behavior and performance of the imputation method, and how it compares to other methods. One improvement that could be made to the proposed method would be to account for the longitudinal aspect of EHR data by combining the autoencoder with a long-short term memory model. Other improvements include trying end-to-end training and including more computationally expensive but effective predictor models such as XGboost or a shallow neural network.

Aside from the motivation for using autoencoders for imputation (Section I), autoencoders provide another unexplored advantage. In the process of learning to impute a given dataset, autoencoders also learn to encode the data into latent representations via the encoder. We plan to explore how using latent representations of the data generated from an autoencoder optimized to impute a dataset may affect downstream tasks. We also plan to explore the performance of downstream predictive models when they are directly fed these latent representations, specifically whether higher performance can be achieved than on the imputed data.

VI. CONCLUSIONS

Autopopulus enabled us to more easily investigate different autoencoder-based imputation methods for our dataset and task of interest. Building our system so that it abstracts away the implementation details of specific variations of autoencoders resulted in a highly extensible framework, enabling rapid and wide experimentation with different autoencoder flavors. From our experiments we were able to see that no one imputation method is definitively better than all others, as the performance of each method varies across missingness mechanism. As such, it is important to always evaluate the performance of a range of imputation methods on one's dataset, as restricting an analysis to a single method may lead to sub-optimal results. Our analysis shows that while autoencoders can be effectively used for imputation in a clinical setting, given the unique nature of each feature and missingness pattern, a combination of imputation methods might be most effective. In addition to presenting Autopopulus, we also explored a novel autoencoder imputation technique that, though it struggled with this dataset, shows promise in future iterations and extensions to other datasets. In general, the differences in performance on the downstream task were rather minuscule for this dataset. We believe this work will enable and encourage practitioners to more thoroughly investigate autoencoder-based imputation and also aid them in selecting the right autoencoder for their task, much as one would tune any hyperparameter.

ACKNOWLEDGMENT

This work was funded by the UCLA CTSI grant (UL1 TR001881), NIH/NIMHD grant R01 MD014712, and NIH/NIBIB T32 training grant (T32 EB016640). Thanks to Anders O. Garlid, Henry Zheng, and David Gordon for assisting with edits.

REFERENCES

[1] Donald B. Rubin. Inference and missing data. *Biometrika*, 63(3):581–592, 12 1976.

[2] Mallorie H. Fiero, Chiu-Hsieh Hsu, and Melanie L. Bell. A pattern-mixture model approach for handling missing continuous outcome data in longitudinal cluster randomized trials. *Statistics in medicine*, 36(26):4094–4105, 2017.

[3] Keith C. Norris, O. Kenrik Duru, Radica Z. Alicic, et al. Rationale and design of a multicenter Chronic Kidney Disease (CKD) and at-risk for CKD electronic health records-based registry: CURE-CKD. *BMC nephrology*, 20(1):416, 11 2019.

[4] Katherine R. Tuttle, Radica Z. Alicic, O. Kenrik Duru, et al. Clinical Characteristics of and Risk Factors for Chronic Kidney Disease Among Adults and Children: An Analysis of the CURE-CKD Registry. *JAMA Network Open*, 2:e1918169–e1918169, 2019.

[5] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2013.

[6] Guillem Boquet et al. Missing Data in Traffic Estimation: A Variational Autoencoder Imputation Method. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2882–2886, 05 2019.

[7] John T. McCoy, Steve Kroon, and Lidia Auret. Variational Autoencoders for Missing Data Imputation with Application to a Simulated Milling Circuit. *IFAC-PapersOnLine*, 51(21):141–146, 01 2018.

[8] Christopher K. I. Williams, Charlie Nash, and Alfredo Nazábal. Autoencoders and Probabilistic Inference with Missing Data: An Exact Solution for The Factor Analysis Case. 02 2019.

[9] Ramiro D. Camino, Christian A. Hammerschmidt, and Radu State. Improving Missing Data Imputation with Deep Generative Models. 02 2019.

[10] Junjie Chen and Xinghua Shi. Sparse Convolutional Denoising Autoencoders for Genotype Imputation. *Genes*, 10(9):652, 08 2019.

[11] Lovedeep Gondara and Ke Wang. MIDA: Multiple Imputation using Denoising Autoencoders. 02 2018.

[12] Brett K. Beaulieu-jones and Jason H. Moore. Missing Data Imputation In The Electronic Health Record Using Deeply Learned Autoencoders. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, 22:207–218, 2016.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine Learning in Python . *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] Lisha Li, Kevin Jamieson, Afshin Rostamizadeh, et al. Massively parallel hyperparameter tuning, 2018.

[15] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[16] Travis E Oliphant. *A guide to NumPy*, volume 1. Trelgol Publishing USA, 2006.

[17] WA Falcon and .al. Pytorch lightning. *GitHub. Note: https://github.com/PyTorchLightning/pytorch-lightning*, 3, 2019.

[18] U. Fayyad and K. Irani. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *IJCAI*, 1993.

[19] Janez Demšar, Tomaž Curk, Aleš Erjavec, et al. Orange: Data mining toolbox in python. *Journal of Machine Learning Research*, 14:2349–2353, 2013.

[20] David M Maslove, Tanya Podchiyska, and Henry J Lowe. Discretization of continuous features in clinical datasets. *Journal of the American Medical Informatics Association : JAMIA*, 20(3):544–553, 2013.

[21] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[22] Classification Systems for Acute Kidney Injury: Background, RIFLE Classification, Acute Kidney Injury Network. 2, 03 2012.

[23] Hiddo J. Lambers Heerspink, Hocine Tighiouart, Yingying Sang, et al. GFR Decline and Subsequent Risk of Established Kidney Outcomes: A Meta-analysis of 37 Randomized Controlled Trials. *American Journal of Kidney Diseases*, 64(6):860–866, 2014.

[24] Richard Liaw, Eric Liang, Robert Nishihara, et al. Tune: A research platform for distributed model selection and training. *arXiv preprint arXiv:1807.05118*, 2018.