Received: 29 December 2020 | Revised: 9 November 2021 | Accepted: 6 February 2022

DOI: 10.1002/bimj.202000393

Biometrical Journal

RESEARCH ARTICLE

Missing data imputation in clinical trials using recurrent neural network facilitated by clustering and oversampling

Halimu N. Haliduola[1,2] | **Frank Bretz**[3,4] | **Ulrich Mansmann**[1]

[1]Institute for Medical Information Processing, Biometry and Epidemiology (IBE), LMU Munich, Munich, Germany

[2]Alvotech Germany GmbH, Jülich, Germany

[3]Novartis Pharma AG, Basel, Switzerland

[4]Section for Medical Statistics, Medical University of Vienna, Vienna, Austria

Correspondence
Ulrich Mansmann, Institute for Medical Information Processing, Biometry and Epidemiology (IBE), LMU Munich, 81377 Munich, Germany.
Email: mansmann@ibe.med.uni-muenchen.de

This article has earned an open data badge "**Reproducible Research**" for making publicly available the code necessary to reproduce the reported results. The results reported in this article could fully be reproduced.

Abstract

In clinical practice, the composition of missing data may be complex, for example, a mixture of missing at random (MAR) and missing not at random (MNAR) assumptions. Many methods under the assumption of MAR are available. Under the assumption of MNAR, likelihood-based methods require specification of the joint distribution of the data, and the missingness mechanism has been introduced as sensitivity analysis. These classic models heavily rely on the underlying assumption, and, in many realistic scenarios, they can produce unreliable estimates. In this paper, we develop a machine learning based missing data prediction framework with the aim of handling more realistic missing data scenarios. We use an imbalanced learning technique (i.e., oversampling of minority class) to handle the MNAR data. To implement oversampling in longitudinal continuous variable, we first perform clustering via k-mean trajectories. And use the recurrent neural network (RNN) to model the longitudinal data. Further, we apply bootstrap aggregating to improve the accuracy of prediction and also to consider the uncertainty of a single prediction. We evaluate the proposed method using simulated data. The prediction result is evaluated at the individual patient level and the overall population level. We demonstrate the powerful predictive capability of RNN for longitudinal data and its flexibility for nonlinear modeling. Overall, the proposed method provides an accurate individual prediction for both MAR and MNAR data and reduce the bias of missing data in treatment effect estimation when compared to standard methods and classic models. Finally, we implement the proposed method in a real dataset from an antidepressant clinical trial. In summary, this paper offers an opportunity to encourage the integration of machine learning strategies for handling of missing data in the analysis of randomized clinical trials.

KEYWORDS

clinical trial, k-mean clustering, missing data, oversampling, recurrent neural network

1 | INTRODUCTION

In clinical trials, missing data are the data that would be meaningful for the analysis but is not documented. Missing data, if not handled properly, will lead to lower statistical power as the sample size is reduced. In addition, dropouts from the trial may have poor outcomes or extreme values (e.g., treatment failure may lead to dropout). Therefore, the loss of these dropouts could lead to a bias in the estimated treatment effect (especially when the missing values are more likely in one treatment arm because it is not as effective as the other) and an underestimate of the variability. Missing data may also impact the external validity of the study outcome.

There are three types of missingness mechanisms (Rubin, 1976). (*i*) Missing completely at random (MCAR): if the probability of missingness does not depend on observed or unobserved measurements, for example, patients move to another city due to non-health related reasons. (*ii*) Missing at random (MAR): if the probability of missingness depends only on observed measurements conditional on the covariates in the model, for example, younger people may more likely to have blood pressure not measured. (*iii*) Missing not at random (MNAR): if the probability of missingness depends on unobserved measurements, for example, patients discontinue from the trial due to lack of efficacy, that is, patients do not come for the visit as the disease status worsening hence the data are missing.

The three types of missingness are clearly defined, but in practice it is typically not possible to be certain whether there is a relationship between missing values and the unobserved outcome variable. That is, it is not possible to ascertain whether the MAR/MCAR assumptions are appropriate in any practical situation. A mixed strategy may be considered. For example, assume that dropouts due to lack of efficacy are MNAR and loss to follow-up are MAR (European Medicines Agency, 2010). The consequence of MNAR is that the missing data cannot be simply predicted using observed data from that patient. In addition, the distribution of completers' data and MNAR data are different; thereby, it is not plausible to impute missing data using the completers' data. This is the central problem of missing data analysis in clinical trials (National Research Council of the National Academies, 2010).

Many established methods for handling missing data under the assumption of MAR are available. Under the alternative assumption of MNAR, the following classic models have been introduced as sensitivity analysis in the past decades: selection model (SM) (Heckman, 1976; Rubin, 1976), pattern mixture model (PMM) (Little, 1993, 1994, 1995), and shared parameter model (SPM) (Little, 1995). Let $(Y_{i,obs}, Y_{i,mis}, R_i)$ denote the data for ith patient, $Y_{i,obs}$ is for the observed component, $Y_{i,mis}$ is for the missing component, and R_i is the missingness indicator (1 = missing, 0 = observed). The full density function is described as $f(Y_{i,obs}, Y_{i,mis}, R_i|\Theta, \psi)$, where the parameters vectors Θ and ψ describe the response and missingness processes, respectively. The SM and PMM are developed by factorizing the full density function differently. The SM is based on the below factorization:

$$f(Y_{i,obs}, Y_{i,mis}, R_i|\Theta, \psi) = f(Y_{i,obs}, Y_{i,mis}|\Theta) f(R_i|Y_{i,obs}, Y_{i,mis}, \psi). \tag{1}$$

The first part is the marginal density of the response process and the second part is the density of the missingness process, conditional on the response. The PMM can be seen as a mixture of different populations, characterized by the observed pattern of missingness, it is based on the below factorization:

$$f(Y_{i,obs}, Y_{i,mis}, R_i|\Theta, \psi) = f(Y_{i,obs}, Y_{i,mis}|R_i, \Theta) f(R_i|\psi). \tag{2}$$

The SPM assumes that the response process Y_i and the missingness process R_i are conditionally independent of each other by sharing a random effect b_i. Therefore, the density function can be described as

$$f(Y_{i,obs}, Y_{i,mis}, R_i|\Theta, \psi) = \int f(Y_{i,obs}, Y_{i,mis}|b_i, \Theta) f(R_i|b_i, \psi) f(b_i) db_i. \tag{3}$$

As mentioned above, these models are introduced as sensitivity analysis assuming MNAR (i.e., assuming all missing data are MNAR alternatively). However, in reality, the composition of missing data may be more complex, for example, a mixture of MAR and MNAR. A relatively large number of empirical studies have examined the performance of the classic models. These studies suggest that those models can reduce or eliminate bias when their assumptions are met. However, in many realistic scenarios, the models can produce estimates that are even worse than those of MAR-based missing data-handling methods (Enders, 2010). Those models heavily rely on the underlying assumption; unfortunately, these assumptions are largely untestable, so there is no practical way to judge the model's performance in a real data analysis.

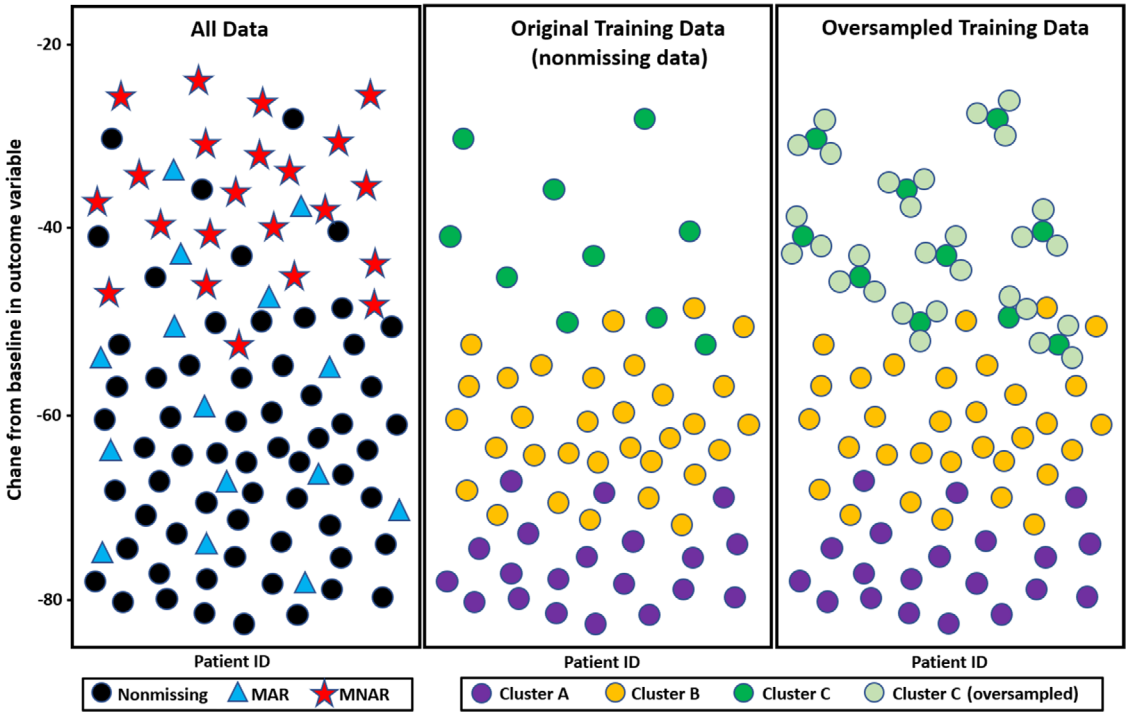

FIGURE 1 Demonstration of clustering and oversampling: cross-sectional of the longitudinal profiles – "change from baseline in outcome variable"

In this paper, with an aim of handling more realistic missing data scenarios (e.g., mixture of MAR and MNAR), a machine learning based missing data prediction framework is developed and evaluated using simulation data and real clinical trial data. According to Breiman (2001), in statistical learning "the goal is not interpretability, but accurate information." In line with this thinking, the proposed framework handles MNAR by emphasizing what is missing, while it also covers MAR by seeking for accurate individual information. The MNAR problem is treated as an imbalanced learning task,that is, the minority class oversampling (Weiss, 2013) is used to compensate for the MNAR data. To implement oversampling in longitudinal continuous data, clustering via k-mean trajectories (Gower, 1971) is performed first. Patients are clustered into "good responder," "medium responder," and "low responder," according to their longitudinal efficacy profiles. In our simulation study, to simplify the problem, we assume the dropouts due to lack of efficacy are MNAR. Therefore, those patients are mostly in the "low responders" class (of course it could be the other way round in reality, i.e., extreme good responders discontinue from the trial as they consider no need to continue the treatment). Depending on the proportion of MNAR data, the size of available data in the worst cluster can be smaller than the size of the other clusters (i.e., imbalanced distribution). In order to compensate for the MNAR in that cluster, and also to avoid the individual prediction being driven by the available data from the completers to an overall average level, random oversampling (with replacement) in the minority class is necessary. See Figure 1 for a display of the distribution of "change from baseline in an outcome variable" (cross-sectional of the longitudinal profiles) in a simulated example. The full data (including the nonmissing data and the values that are set to "missing" in the simulation) are presented in the left panel, black dot = nonmissing data, red star = MNAR, and blue triangle = MAR. The nonmissing data (i.e., the original training data) are repeated in the middle panel and clustered into three groups: green = "low responders," orange = "medium responder," and purple = "good responder." It is clear that the nonmissing data is not a good representative of the full data considering the MNAR data. The "low responders" (green dots) are relatively rare in the original training data (i.e., the minority class). When random oversampling is applied in the minority class, as shown in the right panel (the light green dots are the oversampled cases), there are more data points in the "low responders" area to compensate for the MNAR, that is, the distribution of green and light-green dots approximates the distribution of green and red dots.

One may question the oversampling applied here, arguing that the distribution of data has been changed due to the oversampling, and this will impact the treatment effect estimation. First, the oversampled dataset will never be used to estimate the treatment effect. All of the efforts taken here are to minimize the individual error in statistical learning. Once

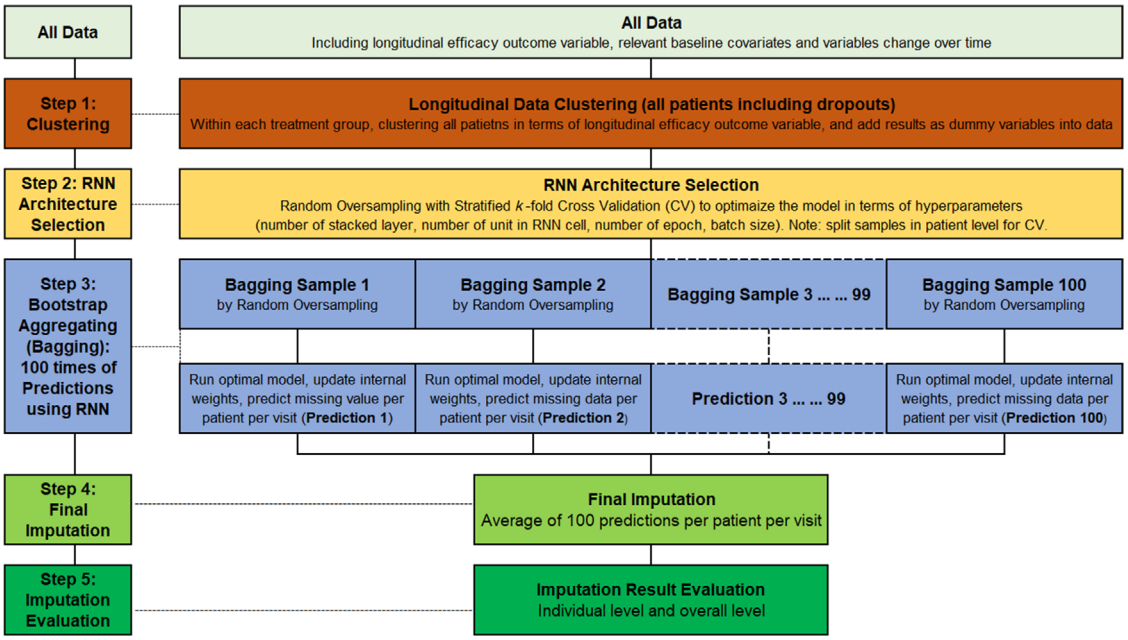

FIGURE 2 Overall workflow diagram: process to handle realistic missing data scenario in a longitudinal setting. Step 1: Structure the longitudinal efficacy response data by clustering; Step 2: select the optimal RNN architecture facilitated by oversampling and stratified K-fold CV; Step 3: perform multiple predictions for individual patients with missing data (facilitated by oversampling to avoid bias introduced by missing mechanisms); Step 4: create the final imputed dataset; Step 5: evaluate the imputation result

the prediction for each patient is optimized by minimizing the individual error, the treatment effect will be estimated based on the observed data plus the imputed data (i.e., no oversampling in the data analysis part).

We use recurrent neural networks (RNN) to model the longitudinal data. RNN is a type of neural network that can learn from the past to predict the future outcomes (Rumelhart et al., 1986; Schmidhuber, 1993). This allows us to exhibit temporal dynamic behavior for time sequence data, and thus it is a powerful tool for longitudinal clinical data modeling. RNN provides flexible nonlinear modeling without requiring any or much domain knowledge about the interrelationship between the variables, it learns automatically from the training data to estimate the weights and then predict the new data. Different RNN architectures are experimented to tune various hyperparameters, and the optimal model is selected via the bias-variance trade-off approach (Claesen & De Moor, 2015). To improve the accuracy of prediction and also to consider the uncertainty of a single prediction, bootstrap aggregating (bagging) is implemented. In light of the "evidence-based computational statistics" (Boulesteix et al., 2017, 2018), the proposed method is evaluated in the practically relevant simulation data and exemplified in a real clinical trial data. In the simulation data, the real-life plausibility of the simulation scenarios is emphasized. A mixed missing mechanism of MAR and MNAR is considered in the simulation. The real dataset is from an antidepressant clinical trial, which is one of the few publicly available datasets that can be used to demonstrate methods for handling missing data where a continuous outcome is measured repeatedly. The imputation results are evaluated at the individual patient level and the overall population level. Overall, the proposed methods provided plausible individual prediction for both of the MAR and MNAR data and reduced the bias of missing data in treatment effect estimation. Therefore, this paper offers an opportunity to encourage the integration of machine learning strategies for handling of missing data in the analysis of randomized clinical trials.

2 | METHODS

We propose a computational approach in this paper which comprises various individual components (see Figure 2 for the overall workflow). The proposed framework handles MNAR by emphasizing what is missing, while it also covers MAR by seeking for accurate individual information. The MNAR problem is treated as an imbalanced learning task, that is, the minority class oversampling is used to compensate for the MNAR data.

The first step is to cluster all patients (including the dropouts) according to their longitudinal efficacy profiles. Clustering structures the longitudinal data within each treatment group. It is a key concept in the proposed approach due to the following reasons: (i) the clusters (clustering results) are used in the stratified k-fold cross-validation (CV) and the random oversampling step as the "categorization" of the continuous target variable to balance the majority and minority clusters; (ii) the clusters are used (as dummy variable) in the RNN model to indicate the longitudinal pattern of patient efficacy profiles, which is important to borrow information from the similar patients (seeking for accurate individual information). Technical details about clustering are provided in Section 2.1.

The second step is to select the optimal RNN architecture. We use RNN to model the complex longitudinal data in a nonparametric manner (details about RNN are provided in Section 2.2). The RNN architectures feature a set of hyperparameters (e.g., number of units in each RNN cell, number of stacked layers, batch size, and number of epochs) that must be determined before training commences. In this step, the optimal RNN architecture (which can learn adequately from the training data and also performs equally well in the validation data) is selected via a bias-variance trade-off approach. Considering the data distribution with the presence of MNAR (as discussed in Section 1), stratified k-fold CV and oversampling of minority class are implemented in the RNN architecture selection process. Details are provided in Sections 2.3 and 2.4.

The third step is to generate, say, 100 bootstrap aggregating (bagging) samples with the minority classes oversampled, and to predict the missing data in each bagging sample. An ensemble method (i.e. bagging) is used to improve the prediction accuracy and also to consider the uncertainty of a single prediction. The optimal RNN model (in terms of hyperparameters) is executed 100 times, each time updating the internal weights of the RNN model and providing predictions for the missing data. In practice, the number of bagging can be even higher. We use 100 as a reasonable number in this paper as the proposed method is time consuming and computationally intensive. Within each bagging, minority classes are oversampled to compensate for the MNAR data.

The fourth step is to average all 100 predicted values for each patient at each visit. These are considered as final imputation.

The fifth step is to evaluate the imputation results at an individual patient level by visualizing the efficacy profiles. The treatment effect is estimated from the imputed data by commonly used statistical analysis methods.

The treatment effect estimated from the imputed data using the commonly used methods is compared with the treatment effect that was estimated using different methods including commonly used methods and the classic models (like SM, PMM, and SPM) that are applied without missing data imputation.

2.1 | Longitudinal data clustering

The k-mean clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. This is done by alternating an expectation phase and a maximization phase. In the expectation phase, the center of each cluster is determined, then in the maximization phase, each observation is assigned to its nearest cluster. This process is repeated until no changes in the clusters occur. For k-mean trajectories, different types of distance can be calculated. The `R package kml` is used in this paper (Genolini & Falissard, 2011). Considering the missing data, the classic Euclidian distance with Gower adjustment (Gower, 1971) is used. Hence the dropouts are also clustered based on their available data. Consider a set of n patients. The target variable is measured for each patient up to time t. Let Y_i denotes the patient i, and let Y_{ik} denotes the measurement for patient i at time k. The difference of the trajectories between two patients i and j can be calculated using the classic Euclidian distance with the Gower adjustment:

$$Dist_{GA}^{E}(Y_i, Y_j) = \sqrt{\frac{t}{\sum_{k=1}^{t}(\omega_{ijk})}\sum_{k=1}^{t}(Y_{ik} - Y_{jk})^2\omega_{ijk}}. \tag{4}$$

Here, ω_{ijk} equals 0 if Y_{ik} or Y_{jk} are missing, and 1 otherwise. Assuming the distribution of the target variable is different between the treatment group, we perform the clustering within each treatment group. The number of cluster should be decided on a case-by-case basis and should be prespecified. In this paper, we set the number of clusters to three within each treatment group with the idea of splitting the patients into "good response," "medium response," and "low response" categories according to their efficacy profile.

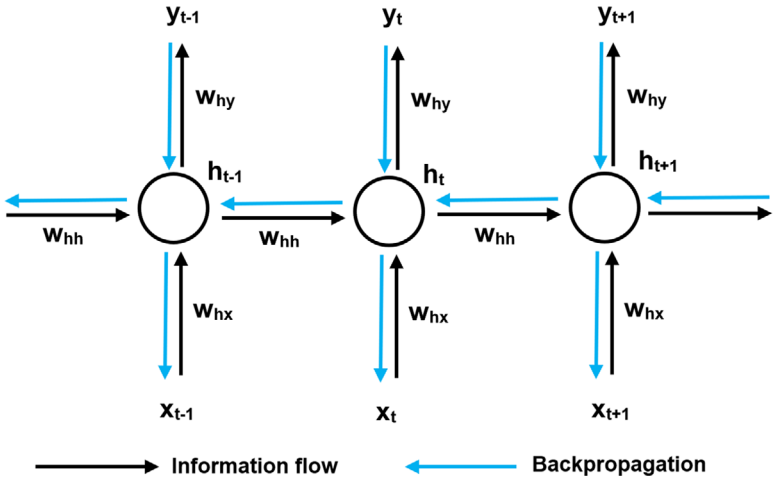

FIGURE 3 An RNN in time of the computation (modified based on LeCun et al., 2015)

2.2 | Recurrent neural network

An RNN is a type of neural network that can learn from the past to predict future outcomes. A basic RNN (Rumelhart et al., 1986; Schmidhuber, 1993) is shown in Figure 3. It uses hidden states (which temporarily store information about the past) to transfer information through time. At time t, the weighted sum of the input information x_t (e.g., the variables change over time) and the previous hidden state (h_{t-1}) is processed using an activation function (e.g., the weighted sum is squashed between −1 and 1 using a hyperbolic tangent (Tanh) function (see details of the Tanh function in Appendix Figure A.1). Then the processed information (hidden state h_t) is carried forward to the next time step. Meanwhile, the hidden state h_t can be output as prediction result y_t for time t using an appropriate activation function, for example, a Rectified Linear Unit (ReLU function; see Figure A.1) for a continuous outcome variable. In this way, an RNN can map an input sequence with elements x_t into an output sequence with elements y_t, with each y_t depending on all the previous $x_{t'}$ (for $t' \leq t$). The same internal weights (matrices w_{hx}, w_{hh}, w_{hy}) are used at each time step. For the internal weight optimization, a backpropagation algorithm can be applied to the computational graph of the unfolded network from the right to the left, that is, to compute the derivative of the error with respect to all the internal weights. The error, also called "loss" in machine learning, is calculated as the predicted value minus the observed value for a continuous outcome variable.

In addition to the basic RNN unit mentioned above (which contains a Tanh activation function), there are other types of RNN units. For example, the long short-term memory units (LSTM) (Gers et al., 1999; Hochreiter & Schmidhuber, 1997) and the gated recurrent unit (GRU) (Cho et al., 2014) are the most commonly used ones. Empirical evaluations show that LSTM and GRU perform superior over the basic RNN (Chung et al., 2014; Shewalkar et al., 2019). LSTM performs slightly better than GRU in terms of prediction accuracy (Shewalkar et al., 2019). We provide a basic introduction to LSTM in Appendix A.1. In this paper, LSTM is implemented using the `Keras library version 2.2.4` (Falbel et al., 2015) in `Python (version 3.6)`.

Neural networks use stochastic gradient descent which is an iterative method for optimizing an objective function with suitable smoothness properties. It can be regarded as a stochastic approximation of gradient descent optimization since it replaces the actual gradient (calculated from the entire dataset) with an estimate thereof (calculated from a randomly selected subset of the data, called "batch"). The machine learning algorithms consider the problem of minimizing an objective function that has the form of a sum:

$$Q(w) = \frac{1}{n}\sum_{i=1}^{n} Q_i(w), \tag{5}$$

where the internal weight w that minimizes $Q(w)$ is to be estimated. Each summand function Q_i is typically associated with the ith observation in the training dataset. When used to minimize the above function, a batch gradient descent method would perform the following iterations:

$$w^* = w - \eta\left(\frac{\partial Loss}{\partial w}\right), \tag{6}$$

where w^* is the new weight, w is the old weight, η is a step size (also called "learning rate"), the last part of this equation is the derivative of loss with respect to the weights. The `Keras library` implements the adaptive moment estimation (Adam) optimizer (Kingma & Ba, 2014). Empirical results demonstrate that Adam works well in practice and compares favorably to other stochastic optimization methods (Kingma & Ba, 2014). In Adam, the learning rate is initialized (e.g., default initial value = 0.001 in `Keras`) and then adapted automatically in training iterations.

There are three types of input variables in an RNN. (i) *Initial state*: the hidden state (h_0) at time step t_0. In practice, the default approach is to set the initial state as zero. However, if the impact of the initial state is not negligible, it makes sense to train the initial state as a variable. Thereby the model can start to learn from a good default state. In a clinical study, for the continuous outcome variable, the baseline value of the outcome variable can be considered as the initial state in an RNN. (ii) *Series input*: the variables change over time (x_t). (iii) *Static input*: for example, relevant demographics and baseline characteristics. In the `Keras library`, static input can be implemented by passing external constants to the RNN, there are no internal model weights learnt for static inputs. For all types of input data, the continuous variables need to be standardized before feeding into RNN to make the calculation faster.

When training the RNN, the nonmissing data from the dropouts should also be used as this particular part of the data is quite important to learn a certain pattern of those dropouts. This may be more important for the dropouts due to lack of efficacy (MNAR) as the trend of their efficacy profiles can be very different from the patients who completed the study. If possible, variables to indicate the missing mechanism (whether MNAR or not) need to be included in the model. Including the dropouts in the model will lead to different lengths of time sequence in data. Using an RNN, a fixed length of time series input is expected in the current available deep learning packages. An effective way to handle this problem is to use sample weight, that is, to create a metric per patient per time to indicate which time points to use in the learning. For example, the weights are set to 1 for nonmissing time steps, and 0 for missing time steps. These metrics are then multiplied by the loss (e.g., the difference between the predicted value and the actual value) per patient and per time before training the RNN. For example, for patient j at time t, the final loss is

$$loss_{jt} = loss'_{jt} w_{jt}, \tag{7}$$

where $loss'_{jt}$ is the loss calculated before using the sample weight metrics, w_{jt} is the sample weight for patient j at time t. By having such sample weight, the missing time steps will be ignored in learning.

2.3 | Minority class oversampling

As mentioned in Sections 1 and 2.1, we cluster patients into "good responder," "medium responder," and "low responder" according to their longitudinal efficacy profiles within each treatment group. Depending on the proportion of MNAR data, the size of available data in the worst cluster can be smaller than in the other clusters (i.e., the worst cluster is the minority class; see Section 1 for more details). To compensate for the MNAR in that cluster, and also to avoid the individual prediction being driven by the available data from the completers to an overall average level, random oversampling (with replacement) in the minority class is necessary. This process involves randomly selecting examples from the minority class, with replacement, and adding them to the training dataset. The amount of oversampling is a hyperparameter of the system (Chawla et al., 2002), it should be decided on a case-by-case basis. We describe two slightly different oversampling approaches in Section 3.3.2 (for the simulation studies) and Section 4 (for the real data implementation). Oversampling is implemented in both Step 2 (RNN architecture selection) and Step 3 (bootstrap aggregating) of the proposed framework to compensate for the MNAR data in both model selection and individual prediction processes.

2.4 | Stratified *k*-fold CV

In machine learning, hyperparameters are the parameters whose values are used to control the learning process (this is different from the model internal parameter or weight whose values need to be optimized during the training process). In general, for RNN, hyperparameter may include learning rate (see details in Section 2.2), number of epoch (defined as the number of times that the learning algorithm work through the entire training dataset), and batch size (defined as the number of samples to work through before updating the internal weights of the model). In addition to those general hyperparameters, the number of units in each RNN cell and number of stacked RNN layers are also needed to be specified before

training commences (as a more complex model may lead to overfitting). The choice of hyperparameters can significantly affect the resulting model's performance; hence, a disciplined, theoretically sound search strategy is essential (Claesen & De Moor, 2015). The bias-variance trade-off is the most commonly used approach for hyperparameter tuning with a goal of selecting a model that can learn adequately from the training data and also performs equally well in validation data.

The k-fold CV is the standard tool for hyperparameter tuning to address the overfitting/underfitting problem. In k-fold CV, the original sample is randomly split into k approximately equal-sized subsets. Of the k subsets, a single subset is retained as the validation data for testing the model and the remaining $k-1$ subsets are used as training data. The CV process is then repeated k times, with each of the k subsets used exactly once as the validation data. Ideally, all possible combinations of hyperparameters should be experimented using the CV approach. For each scenario, the loss over iteration history should be calculated and visualized (to facilitate the comparison) for both training data and validation data. The value of the hyperparameter that provides the least loss for both training data and validation data should be determined as the optimal value. Although the partition of the k fold is performed randomly, it does not guarantee to have a balanced distribution of the target outcome variable in each fold without supervision (especially with the presence of MNAR in the data). Stratified k-fold CV seeks to ensure that each fold is a representative subset of the whole data in terms of the variable of interest. This is very important for the MNAR as by stratification the dropouts due to lack of efficacy will be included in each fold equally, which means in each time when repeating the training process, the certain patterns of target variable in those dropouts (their nonmissing part) will be learnt adequately. Whenever the joint application of CV and oversampling concerns, the "overoptimism" issue should be emphasized and distinguished from overfitting (Santos et al., 2018). If the entire original data is oversampled first and then CV is performed later on, the same samples may appear in both of the training and validation partitions, thereby the model performs "very well" in both partitions. This is known as the "overoptimism" issue. Therefore, a better approach would be that the dataset is first divided into k stratified partitions and the oversampling happens in the training data part only. The validation data are never oversampled or seen by the model in the training stage, thereby allowing a proper evaluation of the model's performance for the generalization purpose.

3 | SIMULATION STUDY TO EVALUATE PERFORMANCE OF METHODS

We evaluate the proposed method by means of an extensive simulation study. In designing the simulation study, we used realistic missing data scenarios. We consider a longitudinal continuous clinical score as the efficacy variable. Further, we consider a mixed missing mechanism of MAR and MNAR in the simulation. One of the advantages of the simulation study in this context is that the "missing values" are known (as the complete data are generated first and some values are set to "missing"), and this can be used as a benchmark to evaluate the performance of the imputation methods (both at t the individual level and the overall level).

3.1 | Design of simulation study

The patient baseline characteristics and longitudinal efficacy data are simulated assuming a two-arm parallel designed clinical trial. Each patient is designed to be treated and assessed biweekly from Week 0 (baseline) up to Week 16 ("primary endpoint"). Different sample sizes (300, 400, or 500 patients), overall dropout rate (20%, 30%, or 40%) and monotone missingness starting time-point (Week 8, Week 10, or Week 12) are considered in the simulation. In total, $3 \times 3 \times 3 = 27$ scenarios are simulated. In each scenario, the patient is randomly assigned to the treatment groups (Test : Control = 1 : 1). A longitudinal clinical score is considered as efficacy variable which decreases over time in general. To take account of the intrapatient correlation, the change from previous visit values in the score (per patient per visit) is modeled considering several fixed factors and a random effect. Similar idea as for the PMM (i.e., data patterns are different for MNAR and completers), a mixed missing mechanism of MAR (discontinue due to lost to follow-up), and MNAR (discontinue due to lack of efficacy) is considered in the simulation. The control group is more impacted by the missing data as the proportions of MNAR and MAR are much higher in the control group than in the test group. The MAR is influenced by the treatment group but not by any other covariates (i.e., within each treatment group, it is actually an MCAR scenario). The complete simulation data before setting the missing values are kept for all patients at all visits for the purpose of imputation result evaluation. The details about the data generation process are described in Appendix A.2.

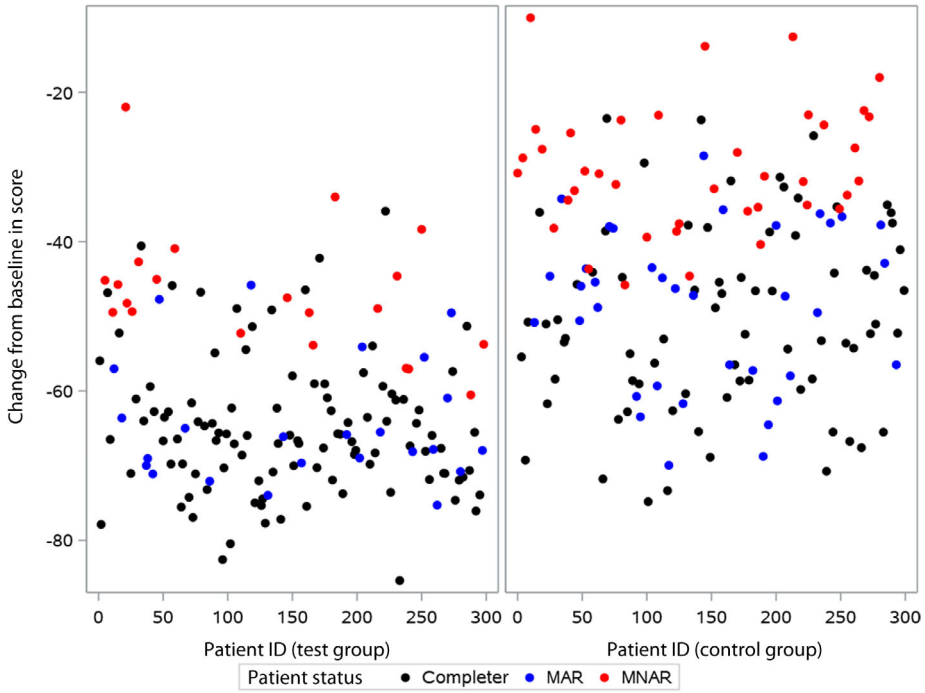

FIGURE 4 Simulation Scenario 300-40-10: scatter plot for "change from baseline in score at Week 16." The black dots are the completers, the blue dots are MAR, and the red dots are MNAR

In this paper, we report the simulation scenario with total sample size = 300, overall dropout rate = 40%, missingness starting at Week 10 (called "Scenario 300-40-10" for short) as an example, as it is one of the scenarios that is most impacted by MNAR. In the test group, 21(14.0%) and 24(16.0%) patients discontinued due to "lack of efficacy" and "lost to follow-up," respectively. In the control group, 39(26.0%) and 36(24.0%) patients discontinued due to "lack of efficacy" and "lost to follow-up," respectively. As a cross section of the longitudinal profiles, the distribution of "the primary endpoint: change from baseline in score at Week 16" is shown in Figure 4. It is clear that the MAR (blue dots) are randomly distributed over the whole data space, but the MNAR data (red dots) are mostly presented in the "low response" area. Therefore, realistic missing data scenarios are successfully "mimicked" in the simulation study. A proper missing data handling method should compensate for the MNAR data and also provide accurate prediction to the MAR data.

In addition, we also simulated the scenarios with only MNAR data, using data from the 27 scenarios mentioned above, but with the MAR values replaced by the known true values (i.e., only MNAR are present in the data). An expected result of these simulations is that the classic models (i.e. PMM, SM, and SPM) perform the best and the proposed approach also performs fairly well.

3.2 | Measuring performance of the proposed methods

To measure the performance of the methods at an overall population level, the treatment effect is estimated using different methods (as described below), and the results are compared in a forest plot. The treatment effect is defined as the difference of change from baseline in score at Week 16 between the treatment group.

Analysis methods

(i) A mixed model for repeat measurement (MMRM) was used for longitudinal data from Week 2 to Week 16. MMRM included treatment, visit, and treatment×visit as fixed effects, baseline score value as covariate, and subject as the random effect.
(ii) Analysis of covariance (ANCOVA) model used for the data at Week 16 only, ANCOVA included treatment as factor and baseline score value as covariate.
(iii) Classic models including PMM, SM, and SPM were used for longitudinal data from Week 2 to Week 16. Details about PMM, SM, and SPM are provided in the Supporting Information (with the `SAS` code and instructions).

Analysis dataset and the corresponding analysis method:

(i) The "true" treatment effect: the simulated complete efficacy data before setting any missing value are analyzed by MMRM and ANCOVA. The "true" treatment effect is used as a benchmark to evaluate the performance of the proposed method and other methods.
(ii) The imputed data (i.e., nonmissing data + data imputation by RNN prediction facilitated with clustering and oversampling) is analyzed by MMRM and ANCOVA models.
(iii) To illustrate the role of clustering and oversampling in handling of MNAR data, the imputed data (i.e., nonmissing data + data imputation by a straightforward RNN prediction without facilitation with clustering and oversampling) is also analyzed by MMRM and ANCOVA models.
(iv) The nonimputed data are analyzed by following commonly used methods: MMRM, ANCOVA, PMM, SM, and SPM.

In addition to the overall level treatment effect comparison, we are also interested in the prediction performance at the individual level. The patient profiles (the observed values and the 100 predicted values) are visualized for all dropouts. The mean value (final imputation) and variability of prediction (measured as the first and third quartiles) at each study week are provided in the plot. The "true" values of missing data (i.e., the simulated complete efficacy data before setting any missing value) are also provided for each patient to visually check the accuracy of the imputation.

3.3 | Simulation results

3.3.1 | Results of longitudinal clustering

As mentioned in Section 2, clustering is the first and very important step. Within each treatment group, patients are clustered into three categories: "good responder," "medium responder," and "low responder" according to their longitudinal efficacy profiles. Figure A.3 in the Appendix shows the individual patient profiles by cluster for Scenario 300-40-10. It is clear that the k-mean clustering captures the longitudinal data structure very well for all patients (including the dropouts). It must be noted that the interpretation of clusters has to be taken with caution as k-mean clustering is unsupervised learning, that is, the clusters/categories are not labeled in the input data. However, the clustering is helpful to structure the longitudinal profile patterns and to seek for similar patients. In addition, as mentioned in Section 1, to learn the pattern of minority clusters adequately in such setting (i.e., the worst clusters in each group with less completers compared to other clusters) and also to avoid the prediction that has been driven by the majority available data from the completers, it is necessary to oversample the small clusters with less nonmissing data.

3.3.2 | Tuning the RNN hyperparameters

Different RNN hyperparameters are tuned using stratified k-fold CV and oversampling (as described in Section 2.4). As mentioned above, it is necessary to oversample the small clusters with less available data. For the hyperparameter tuning purpose, to have a consistent oversampling approach that can be generalized in all simulation scenarios, the following rules are used: the "good" and "medium responder" clusters, and the completers from the "low responder" cluster are oversampled to the size of the largest cluster in that treatment group; in addition, to utilize the available data (nonmissing part) from the dropouts, the dropouts from the worst cluster are also 1:1 random sampled (with replacement). During the CV process, the loss (measured as mean squared error (MSE)) changing over iteration history from the fivefold CV and the MSE at the last training and validation iteration is compared and the appropriate values for the hyperparameters are selected. Based on a substantial number of experiments, the optimal model for the simulation data is determined as LSTM with one single layer and nine units in each cell, iteration epoch of 3000, and batch size of 50.

3.3.3 | Imputation result evaluation at the individual level

The optimal RNN model is performed 100 times. At each time, the data (including the available data from the dropouts) are randomly sampled (minority clusters are oversampled following the same approach as mentioned in Section 3.2.2), the

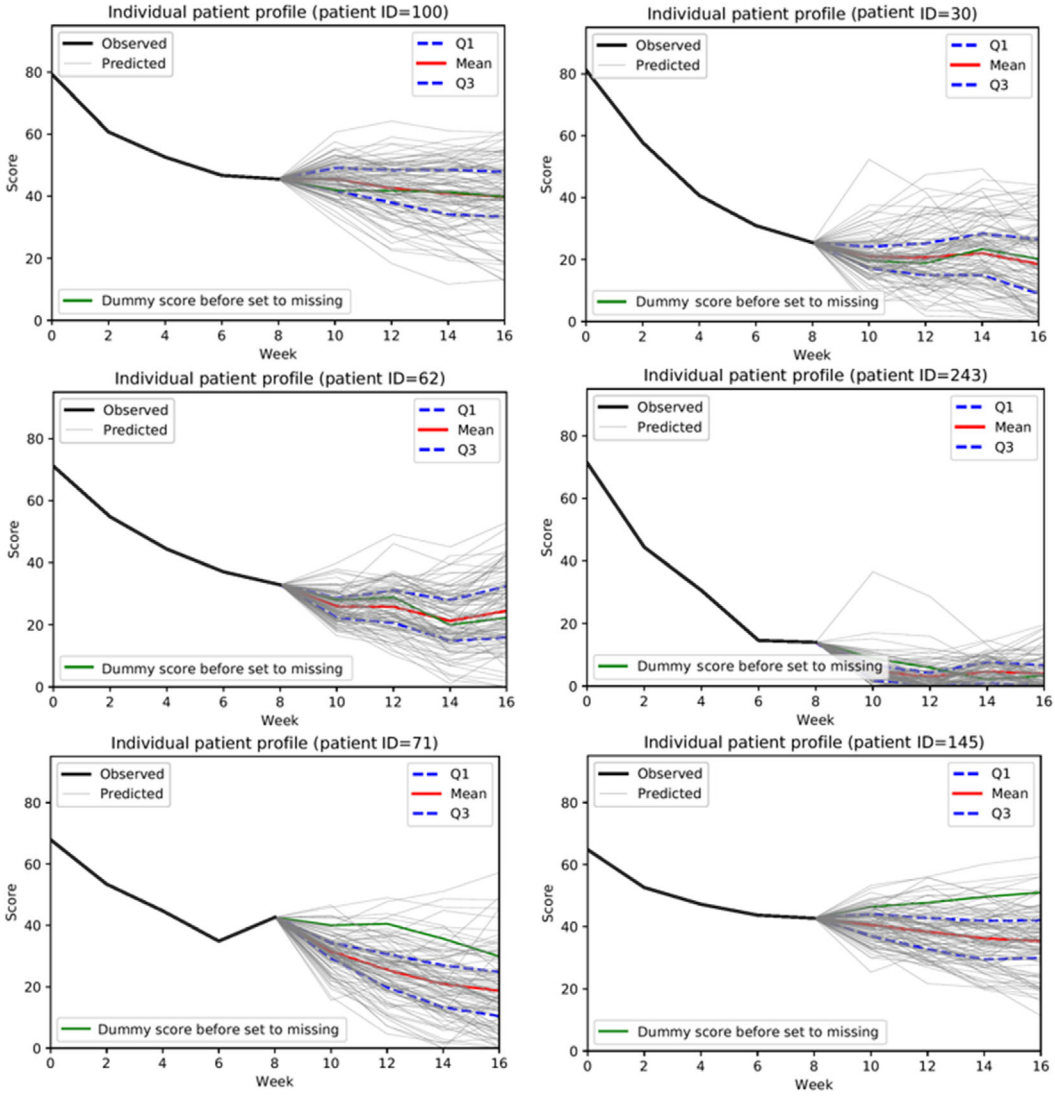

FIGURE 5 Simulation data: examples of individual prediction compared with actual values. The solid black lines are the observed data, each dashed gray line is the prediction from each bagging, and the mean (red lines) and quartiles (blue lines) from the 100 predictions are also provided. The solid green lines are the complete data before setting the missing values (i.e., the true known values)

internal weights are updated and predictions for the missing data are provided within each bagging. For each patient, at each time point, the average of all 100 predicted values is considered as the final imputation. Some examples of the individual patient profile are provided in Figure 5. For the majority of the dropouts, the predictions are close to the actual values (e.g., the first four patients in the figure). The proposed methods provided fairly good predictions for "good," "medium," and "low responders." For some dropouts (< 15% of dropouts), when the intrapatient variability is large or the scores are extremely high at the end of the trial, the imputations are not good as expected (e.g., patient numbers 71 and 145 in Figure 5). Another reason for such bad prediction could be a lack of relevant predictors in the data. Due to the difficulties in the longitudinal data simulation, only a few relevant variables are generated in the simulation data (see details of data generation process in Appendix A.2).

3.3.4 | Imputation result evaluation at the overall population level

The results from different methods (as described in Section 3.2) are presented in a forest plot (Figure 6) for Scenario 300-40-10. The proposed imputation method (i.e., RNN imputation facilitated by clustering and oversampling) + standard analysis method (MMRM or ANCOVA) provided the best estimation of the true treatment effect (i.e., the estimates are the closest

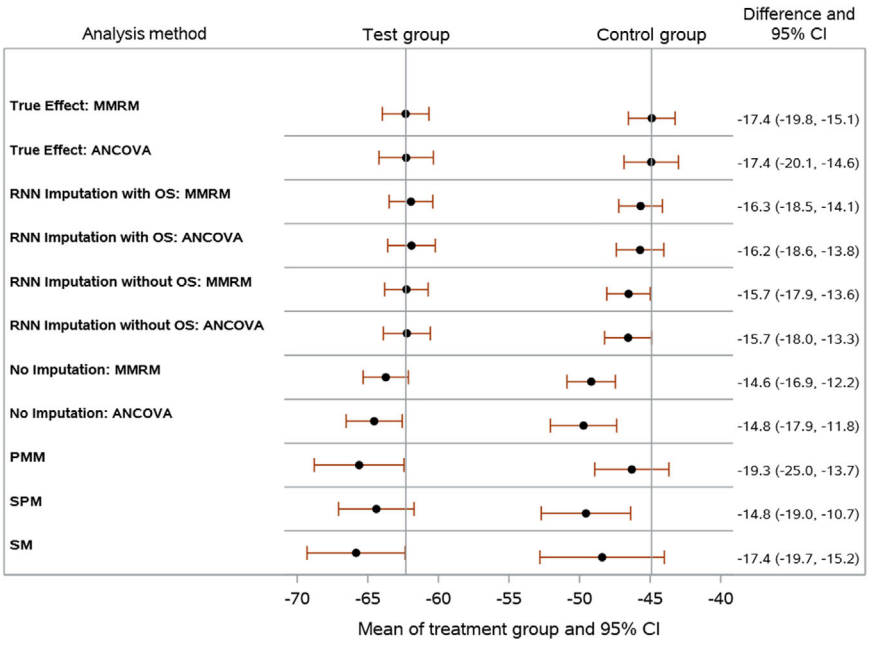

FIGURE 6 Forest plot for analysis results of "change from baseline in score at Week 16" using different methods. True effect: estimation from the complete efficacy data before setting any missing value. RNN imputation: nonmissing data + imputed data by the proposed method. OS = oversampling. No imputation: nonmissing data only. PMM, pattern mixture model; SPM, shared parameter model; SM, selection model

to the true effect in both treatment groups). RNN imputation without clustering and oversampling also provided good estimation (only with a slight bias in the control group), hinted that if the impact of MNAR is considered as ignorable, a simply RNN imputation (without clustering and oversampling) can also provide accurate imputation. Since MMRM and ANCOVA assume MAR, therefore, it is not surprising to have a considerable bias when the MMRM and ANCOVA are applied without imputing the missing value (given the presence of both MNAR and MAR in the data). The bias is larger in the control group where the impact of the MNAR data is much heavier. There is a systematic bias in the results from the classic models in both treatment groups (i.e., the models including PMM, SPM, and SM that are applied without imputing the missing value). Similar to MMRM and ANCOVA that are applied without data imputation, these models tend to overestimate the treatment effect when both MAR and MNAR are present in the data. Since those models heavily rely on the underlying assumption (e.g., a pure MNAR), when a mixture of MAR and MNAR is present in the data, the impact of missing data is somehow not properly handled by these models. The proposed method performed equally well in all 27 simulated scenarios. Similar patterns (as discussed above) are observed in all other simulation scenarios. The analysis results for other 26 scenarios are provided in the Supporting Information. In general, based on the simulation study, the impact of missing data on the treatment effect estimation in a realistic scenario (e.g., mixture of MAR and MNAR) is properly handled by the proposed method by providing accurate individual predictions for both MAR and MNAR data.

In addition, scenarios with only MNAR data are also simulated, see the example of Scenario 300-40-10 in Figure 7 (in this case, the actual proportion of missing data is 20% due to the absence of MAR). In general, as expected, the classical models (i.e., PMM, SM, and SPM) perform better than MMRM and ANCOVA when only MNAR is present in the data, and the proposed approach also performs fairly well.

4 | REAL DATA EXAMPLE

The proposed method is implemented in a real dataset from an antidepressant clinical trial. Original data are from an antidepressant clinical trial with four treatments; two doses of an experimental medication, a positive control, and a placebo (Goldstein et al., 2004). Hamilton 17-item rating scale for depression (HAMD17) is observed at baseline and weeks 1, 2, 4, 6, and 8. To mask the real data, Week 8 observations are removed. Two arms are created: the control group (original placebo arm, $N = 88$) and a test group created by randomly selecting patients from the three nonplacebo arms ($N = 84$). There are 21(25.0%) and 23(26.1%) dropouts in test group and control group, respectively. Within each treatment group, patients are clustered into three subgroups in terms of their HAMD17 score profile. The individual patient profiles by cluster are provided in Figure 8. It is clear that the "good responder" and "low responders" are relatively small clusters. Small clusters are randomly oversampled to the size of the largest cluster ("medium responder") within each treatment group. The oversampling process in the real datasetting is much simpler than the ones used in the simulation data given

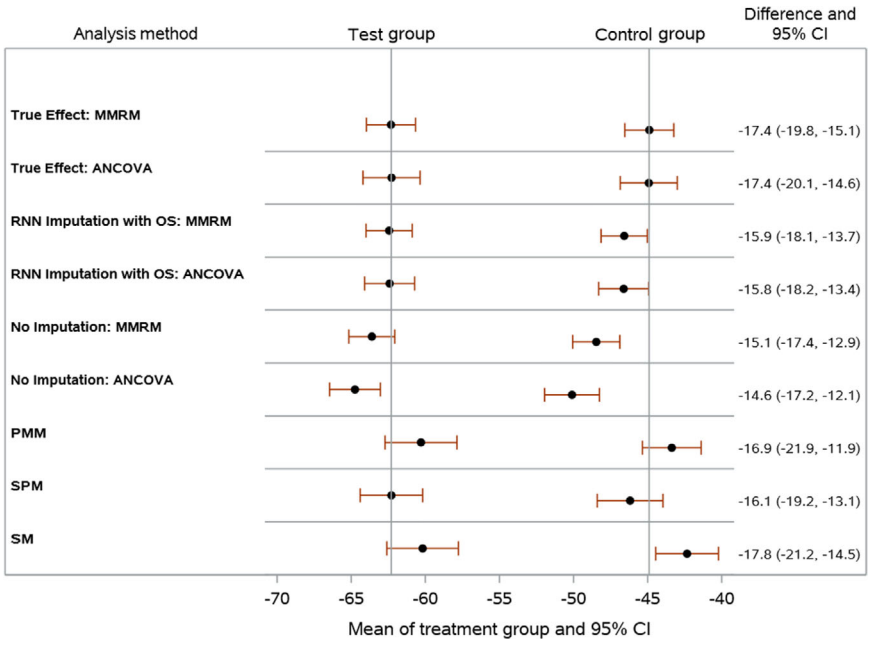

FIGURE 7 Forest plot for analysis results of "change from baseline in score at Week 16" using different methods in the scenario with MNAR only in the data. True effect: estimation from the complete efficacy data before setting any missing value. RNN imputation: nonmissing data + imputed data by the proposed method. OS, oversampling; No imputation: nonmissing data only; PMM, pattern mixture model; SPM, shared parameter model; SM, selection model

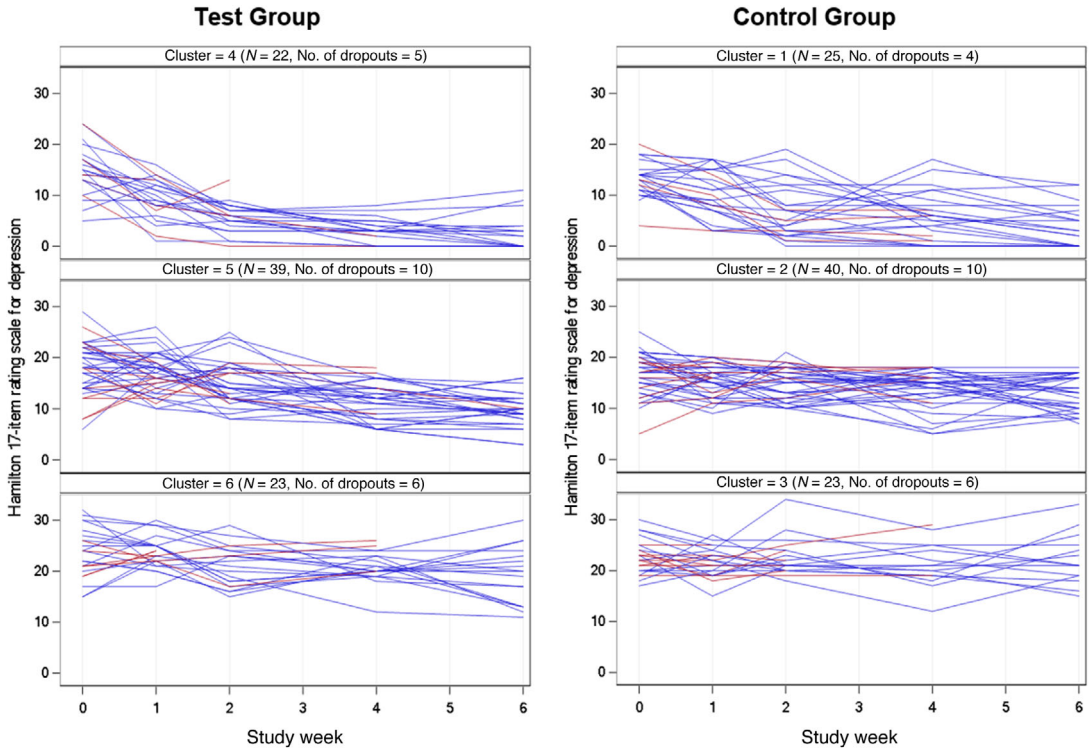

FIGURE 8 Real data: patient profile by the treatment group and cluster. Blue lines are for completers, and red lines are for dropouts

the nature of the data. The dropout reasons are not available in the published real dataset; this makes it difficult to make an assumption about the missing mechanism. Based on the clustering results, the dropouts in "low" and "good" responder clusters could be considered as MNAR, as they may discontinue from the trial due to lack of efficacy or they responded so well before completing the trial and considered that no need to continue with the treatment. Especially the dropouts in the "low responder" clusters (6/23 in the test group vs. 9/23 in the control group), without proper handling of the missing data, their impact on the treatment effect estimation can be nonnegligible.

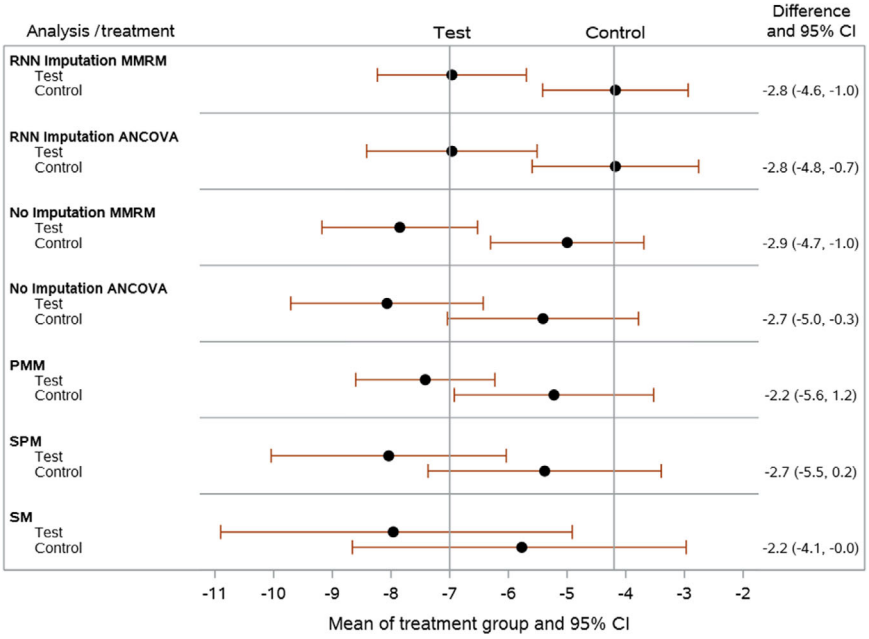

FIGURE 9 Real data: forest plot for analysis results of change from baseline in HAMD17 score at Week 6 using different methods. RNN imputation with OS: nonmissing data + imputed data by the proposed method. No imputation: nonmissing data only. PMM, pattern mixture model; SPM, shared parameter model; SM, selection model

All available variables are used in the RNN model, HAMD17 score as the outcome variable, input variables including gender, treatment, baseline HAMD17 value, HAMD Total score, Patient Global Impression of Improvement (PGI-I), and cluster result (as dummy variables). Based on many experiments, the optimal RNN model for the real data is determined as LSTM with one single layer and seven units in each cell, iteration epoch of 3000, and batch size of 60. The optimal model has been run 100 times, within each bagging, data (including the available data from dropouts) is randomly sampled (smaller clusters are oversampled as mentioned above), the internal weights are updated, and predictions for the missing data are provided within each bagging. The average of 100 predictions is considered as the final prediction. The detailed outputs for clustering and individual prediction are available in the Supporting Information.

The change from baseline in the HAMD17 score is analyzed using the different methods as described in Section 3.4, and the results are presented in the forest plot (Figure 9). The proposed imputation method (i.e., RNN imputation facilitated by clustering and oversampling) + standard analysis method (MMRM and ANCOVA) provided the most conservative estimation for the treatment effect in both treatment groups. Considering the dropouts in the "low responder" clusters (as mentioned above), such conservative estimates may make sense to take into account for the potential impact of MNAR data. Similar to what was observed in the simulation data, there is a systematic bias in the results from the other methods (i.e., MMRM, ANCOVA, PMM, SPM, and SM that are applied without imputing the missing value). Although there are no considerable discrepancies in the point estimates for the difference between treatment groups (maybe due to the dropout rates are similar between treatment groups), the estimates for each treatment group are quite different from the estimates using the proposed method. In general, compared with the proposed method, other methods tend to be optimistic, which may lead to aggressive estimation and hence introduce bias in the study conclusion (especially in the cases when the dropout rate or the efficacy pattern of dropouts are not comparable between the treatment group).

5 | DISCUSSION

As mentioned in the Introduction, it is not possible to ascertain whether the MAR assumptions are appropriate in any practical situation. Therefore, at least a sensitivity analysis to evaluate the impact of MNAR should be warranted if the assumption of MAR cannot be fully justified. In this paper, a machine learning based missing data prediction framework has been developed for longitudinal clinical data with an aim of handling more realistic missing data scenarios. Overall, based on the simulation study, the proposed method provided accurate prediction for both MAR and MNAR data and reduces the bias of missing data in treatment effect estimation. RNN demonstrates the powerful predictive capability for longitudinal data and unrestricted flexibility for nonlinear modeling. Even without being facilitated by any other manner, a straightforward implementation of RNN can provide a fairly good prediction for longitudinal data if there are no severe

MNAR issues in the data. The k-mean trajectory clustering is a crucial step in the proposed method, not only because it facilitates the oversampling and stratified k-fold CV in longitudinal continuous data, but also it is important to borrow information from similar patients by including the cluster information in the RNN model to indicate the longitudinal pattern of efficacy profile. The classic Euclidian distance with Gower adjustment is used in this paper, more insights are needed in the future for other feasible distance metrics and their impact on the imputation results. As the fundamental principle for the imbalanced learning, balancing the classes is key to improve the prediction accuracy for the MNAR data, especially in clinical trials in which the low responders are relatively less and part of them leave the trial due to lack of efficacy. Oversampling the minority class will ensure the efficacy pattern of the low responders been adequately learnt by the model and will also avoid the individual prediction driven by the majority of patients who completed the trial to the overall average level. A simple random oversampling approach (e.g., equalizing the clusters) is taken in this paper, and more insights are needed in the future for other imbalanced learning techniques like different types of oversampling, undersampling, and the combination of both. In addition, it should be noted that the variability of prediction at each study week (measured by the quartiles) is increasing over time (as shown in Section 3.3.4). This hints that the prediction may be less confident for the distant time steps than the near ones. Therefore, when using this method, one should be cautious for the too early dropouts (i.e., the patients with only limited profile available).

In contrast, the commonly used analysis methods (like MMRM and ANCOVA that are applied without imputing the missing value) and classic models (like PMM, SM, and SPM) did not perform as well as the proposed method and showed systematic bias in the treatment effect estimation. These methods tend to overestimate the treatment effect when MNAR is present in the data. This finding is supported in real trial data, that is, a similar pattern of the systematic bias is observed in the real data from an antidepressant clinical trial with a dropout rate of 25%. The performance of classic models in missing data context might need more insights from the practical point of view. Those models heavily rely on the underlying assumption, for example, assuming only MNAR in the data. This kind of assumption can be violated in reality (e.g., missing data can be a mixture of MAR and MNAR), hence leading to suboptimal performance of the model. Additionally, unlike the implementation of joint modeling in complete data (where the binary variable is useful to define the conditional distribution of continuous variable), in the context of missing data, the binary variable only provides the information about missing yes or no. This information actually can also be gained from the continuous variable itself (if its value missing or not). Without providing further information about any feature of the missing data (like potential patterns of efficacy profile or similar patient), the value of such the second process is weakened.

The computational approach comprises necessary components to handle the problem: Step 1: Clustering structures the longitudinal data within each treatment group; Step 2: the RNN models the complex longitudinal data, and the optimal RNN architectures are selected via stratified k-fold CV; Step 3: individual prediction is based on bagging, and the minority class oversampling provides the necessary database for honest predictions; Step 4: average of the bagging predictions is considered as final imputation; Step 5: the imputation results are evaluated at the different levels. Steps 1–3 are reflecting the MNAR problem for longitudinal data with monotones missing patterns. It fits the definition of MNAR, where the missingness depends on the unobserved profile. It is obvious that the proposed methods also incorporate the MAR mechanism by seeking for accurate individual information. The limitation of this paper consists in its special setting studied: monotones missing patterns, continuous longitudinal outcome, three cluster approach with a quite standard metric, and a fixed percentage of MAR and MNAR observations. It is not clear how the proposed strategy will behave in settings that deviate from our assumptions. Therefore, the paper offers an opportunity to encourage the integration of machine learning strategies for handling of missing data in the analysis of randomized clinical trials.

ACKNOWLEDGMENTS

The authors thank Prof. Anne-Laure Boulesteix and Lara Donik for their valuable contribution to this work. The authors also thank the two anonymous reviewers, the associate editor and the editor for their generous and constructive detailed comments that helped us to improve the paper.

CONFLICT OF INTEREST

The authors have declared no conflict of interest.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are openly available in Supporting Information at https://doi.org/10.1002/bimj.202000393.

OPEN RESEARCH BADGES

This article has earned an Open Data badge for making publicly available the digitally-shareable data necessary to reproduce the reported results. The data is available in the Supporting Information section.

This article has earned an open data badge "**Reproducible Research**" for making publicly available the code necessary to reproduce the reported results. The results reported in this article could fully be reproduced.

ORCID

Halimu N. Haliduola https://orcid.org/0000-0002-9157-8187
Frank Bretz https://orcid.org/0000-0002-2008-8340
Ulrich Mansmann https://orcid.org/0000-0002-9955-8906

REFERENCES

Boulesteix, A. L., Wilson, R., & Hapfelmeier, A. (2017). Towards evidence-based computational statistics: Lessons from clinical research on the role and design of real-data benchmark studies. *BMC Medical Research Methodology*, *17*(1), 138. https://doi.org/10.1186/s12874-017-0417-2

Boulesteix, A. L., Binder, H., Abrahamowicz, M., & Sauerbrei, W. (2018). On the necessity and design of studies comparing statistical methods. *Biometrical Journal*, *60*, 216–218. https://doi.org/10.1002/bimj.201700129.

Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical Science*, *16*(3), 199–215. https://doi.org/10.1214/ss/1009213726.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*(2002), 321–357. https://doi.org/10.1613/jair.953

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv:1406.1078 [cs.CL].

Chung, J., Gulcehre, C., Cho, K. & Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. In NIPS 2014 Workshop on Deep Learning, December 2014. Neural Information Processing System Foundation.

Claesen, M. & De Moor, B. (2015). Hyperparameter search in machine learning. arXiv:1502.02127 [cs.LG].

Enders, C. K. (2010). *Applied missing data analysis* (pp. 295–301). Guilford Press.

European Medicines Agency. (2011). *Guideline on missing data in confirmatory clinical trials*. (11–12). EMA.

Falbel, D., Allaire, J., Chollet, F., RStudio, Google, Tang, Y., Van Der Bijl, W., Studer, M., & Keydana, S. (2015). *Keras*. GitHub. https://github.com/fchollet/keras

Genolini, C., & Falissard, B. (2011). KmL: A package to cluster longitudinal data. *Computer Methods and Programs in Biomedicine*, *104*(3), e112–21. https://doi.org/10.1016/j.cmpb.2011.05.008.

Gers, F. A., Schmidhuber, J., Cummins, F. (1999). Learning to forget: Continual prediction with LSTM. In *9th International Conference on Artificial Neural Networks: ICANN '99* (pp. 850–855). https://doi.org/10.1049/cp:19991218.

Goldstein, D. J., Lu, Y., Detke M. J., Wiltse, C., Mallinckrodt, C., & Demitrack, M. A. (2004). Duloxetine in the treatment of depression: A double-blind placebo-controlled comparison with paroxetine. *Journal of Clinical Psychopharmacology*, *24*, 389–399.

Gower, J. C. (1971). A general coefficient of similarity and some of its properties. *Biometrics*, *27*(4), 857–871. https://doi.org/10.2307/2528823

Heckman, J. J. (1976). The common structure of statistical models of truncation, sample selection and limited dependent variables and a simple estimator for such models. *Annals of Economic and Social Measurement*, *5*(4), 475–492. http://www.nber.org/chapters/c10491

Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.

Kingma, D. P. & Ba, J. (2014). Adam: A method for stochastic optimization. Paper presented at The Third International Conference for Learning Representations, San Diego, 2015. arxiv:1412.6980 [cs.LG].

LeCun, Y., Bengio, Y., & Hinton, G. E. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Little, R. J. A. (1993). Pattern-mixture models for multivariate incomplete data. *Journal of the American Statistical Association*, *88*, 125–134.

Little, R. J. A. (1994). A class of pattern-mixture models for normal incomplete data. *Journal Biometrika*, *81*(3), 471–483. https://doi.org/10.2307/2337120.

Little, R. J. A. (1995). Modeling the drop-out mechanism in repeated-measures studies. *Journal of the American Statistical Association*, *90*, 1113–1121.

National Research Council of the National Academies. (2010). *The prevention and treatment of missing data in clinical trials*. (53–54). National Academies Press.

Olah, C. (2015). Understanding LSTM networks. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

Rubin, D. B. (1976). Inference and missing data. *Biometrika*, *63*(3), 581–592. https://doi.org/10.1093/biomet/63.3.581

Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, *323*(6088), 533–536.

Santos, M. S., Soares, J. P., Abreu, P. H., & Araujo, H. J. (2018). Cross-validation for imbalanced datasets: Avoiding overoptimistic and overfitting approaches. *IEEE Computational Intelligence Magazine*, *13*(4), 59–76. https://doi.org/10.1109/mci.2018.2866730

Schmidhuber, J. (1993). Netzwerkarchitekturen, Zielfunktionen und Kettenregel [Network architectures, objective functions, and chain rule] [Habilitation (postdoctoral thesis)]. Institut für Informatik, Technische Universität München.

Shewalkar, A., Nyavanandi, D., & Ludwig, S.A. (2019). Performance Evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU. *JAISCR: Journal of Artificial Intelligence and Soft Computing Research*, *9*, 235–245.
Weiss, G. M. (2013). *Imbalanced learning: Foundations, algorithms and applications*. Wiley-IEEE Press.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: Haliduola, H. N., Bretz, F., & Mansmann, U. (2022). Missing data imputation in clinical trials using recurrent neural network facilitated by clustering and oversampling. *Biometrical Journal*, *64,* 863–882. https://doi.org/10.1002/bimj.202000393

APPENDIX A

A.1 | A short introduction to LSTM

The LSTM unit is more complex than the basic RNN unit. It contains the following elements: two states (cell state and hidden state), input, output, and three gates (forget gate, input gate, and output gate; see Figure A.1). The cell state is the key to LSTM, the horizontal line running through the top of the diagram. The cell state runs straight down the entire chain, with only some minor linear interactions. This allows the LSTM to have the ability to remove or add information to the cell state, carefully regulated by the gates which in a way optionally let information through. The gates are composed of a sigmoid function (see details in Figure A.1), and an element-wise multiplication operation. The sigmoid function squashes information between 0 ("let nothing through") and 1 ("let everything through"). In practice, the learning capability of LSTM can be improved by including more than one unit in each cell (i.e., one LSTM cell can contain several concatenated LSTM units).

As shown in Figure A.1, at each time step, first, for the information that comes from current input vectors (x_t) and the hidden state at the previous time step (h_{t-1}), the forget gate decides what information to throw away from the cell state. The forget gate is expressed as

$$f_t = \sigma_{sig}(w_{fx}x_t + w_{fh}h_{t-1} + b_f), \tag{A.1}$$

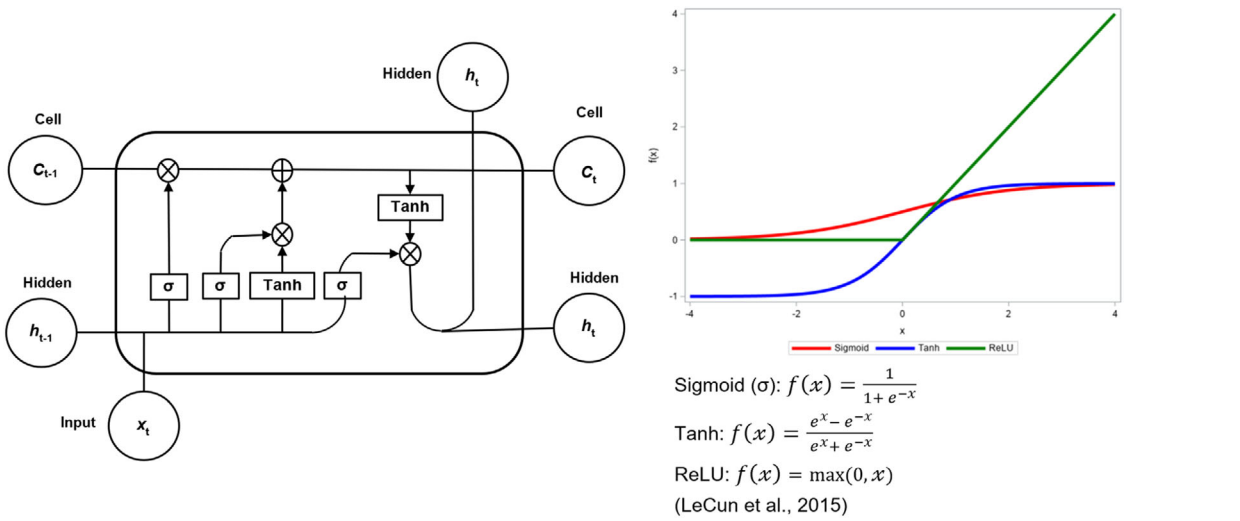

FIGURE A.1 One LSTM unit at time t (reproduced based on Olah, 2015) and the activation functions used in this paper. Note: c_{t-1} = cell state at the previous time step, h_{t-1} = hidden state at the previous time step, x_t = current input vectors, f_t = forget gate, i_t = input gate, Tanh = Tanh activation function, o_t = output gate, h_t = hidden state at current time step, c_t = cell state at current time step, $\otimes$ = element-wise multiplication, $\oplus$ = vector addition

where σ_{sig} is the sigmoid function; w (weights) and b (bias or intercept) is the parameter matrices to be learned. The next step is to decide what new information to store in the cell state. This has two parts. The first part is an input gate i_t, which decides what values to update:

$$i_t = \sigma_{sig}(w_{ix}x_t + w_{ih}h_{t-1} + b_i). \tag{A.2}$$

The second part is a Tanh function (where information is squashed between −1 and 1), which creates a vector of new candidate values which could be added to the cell state, it is expressed as

$$Tanh_t = \sigma_{Tanh}(w_{Tanhx}x_t + w_{Tanhh}h_{t-1} + b_{Tanh}). \tag{A.3}$$

Then the input gate and the outcome of the Tanh function are combined (element-wise multiplication) as an input section to create an update to the state, the input section is expressed as

$$\text{Tanh}_t \otimes i_t. \tag{A.4}$$

In the next step, the cell state at the previous time step c_{t-1} is updated to the current cell state c_t. This is done by adding the element-wise product of the c_{t-1} and the forget gate f_t (i.e., forgetting the things are decided to forget earlier) and the input section (i.e., adding the new candidate values which are scaled by how much that decided to update the cell state). The current cell state is expressed as

$$c_t = c_{t-1} \otimes f_t + \text{Tanh}_t \otimes i_t. \tag{A.5}$$

The final step is to decide what to store in the current hidden state. This has two parts. First, for the information that comes from current input vectors and previous hidden states, the output gate decides what parts to output. The output gate is expressed as

$$o_t = \sigma_{sig}(w_{ox}x_t + w_{oh}h_{t-1} + b_o). \tag{A.6}$$

The second part is the current cell state going through a Tanh function. Then the element-wise product of these two parts is stored as hidden state for the current time step, expressed as

$$h_t = o_t \otimes \sigma_{\text{tanh}}(c_t). \tag{A.7}$$

The current hidden state will be carried forward for the next time step, and it can also be gained as prediction result for the current time step by using an appropriate activation function, for example, a ReLU function for the continuous outcome variable, in that case, the output vector can be expressed as

$$y_t = ReLU(w_{yh}h_t + b_y). \tag{A.8}$$

A.2 | Details on the data generation process used in the simulation study

The patient baseline characteristics and longitudinal efficacy data are simulated assuming a parallel designed clinical trial. Each patient is designed to be treated and assessed biweekly from Week 0 (baseline) up to Week 16. In total, $3 \times 3 \times 3 = 27$ scenarios are simulated. In each scenario, the patient is randomly assigned to the treatment group (test group : control group = 1 : 1). The longitudinal clinical score decreases over time in general. To take account of the intrapatient correlation, and to reflect the influence of the relevant covariates (including the baseline variables and the variables change over time) on the longitudinal profile, the change from previous visit values (per patient per visit) is modeled using several fixed factors and a random effect (more details are provided in Figure A.2). Similar ideas as for the pattern mixture model (i.e., data patterns are different for MNAR and completers), a mixed missing mechanism of MAR (lost to follow-up) and MNAR (dropout due to the lack of efficacy) is considered in the simulation. For the completers and MAR, a completely random effect is considered. For the MNAR, the change from previous visit values shrinks over time (by using the absolute value of the random normal function) so that their scores decrease less or even increase over time. The change from baseline value is calculated by summing up the change from previous visit values up to certain visits within each patient.

Steps	Data Structure	Data Content
Step a: **Patient level information**	One record per patient	**Baseline characteristics and treatment variables** Treatment: test=1 (50%), control=0 (50%); Gender: 0 = female (50%), 1 = male (50%); Body weight (kg) ~ N(75, 10**2); Baseline clinical score ~ N(70, 5**2); **Discontinuation flags**: randomly assign certain number of patients as dropouts (MCAR or MNAR), rules are described as below: **(a)** the proportion of any missing data in Control group is larger than the Test group (i.e., the difference of proportion between treatment group is 1/2 * dropout rate); **(b)** in **Test** group, the proportion of MNAR is slightly lower than MCAR; **(c)** in **Control** group, the proportion of MNAR is higher than MCAR.
Step b: **Longitudinal struction frame**	One record per visit per patient	**Visit:** Week 0, 2, 4, 6, 8, 10, 12, 14, and 16; **Missing record flag**: for the dropouts, the **monotone** missingness started from Week 8, Week 10, or Week 12 based on simulation scenario. **Concomitant Medication flags**: randomly flag 20% records as taken ConMed, and 80% as not taken ConMed.
Step c: **Complete longitudinal score data**	One record per visit per patient	**Change from previous visit (CFPV) values are modeled** as described in below text; **Change from baseline** (CFB) = cumulative CFPV; **Absolute score** = baseline + CFB
Step d: **Create missing value in longitudinal score data**	One record per visit per patient	**Set absolute score and CFB to missing value** if visit is flagged as missing visit at Step b

FIGURE A.2 Data generation process in simulation study

The control group is more impacted by the missing data as the proportions of MNAR and MAR are much higher in the control group than in the test group. The MAR is influenced by the treatment group but not by any other covariates (i.e., within each treatment group, it is actually an MCAR scenario). The complete simulation data before setting the missing values are kept for all patients at all visits (for the purpose of prediction evaluation). The data generation process is shown in Figure A.2.

Longitudinal clinical score is generated as follows:

(i) the score with a range of 0–100, the higher the score the worse the disease status. At baseline, the score from all patients follow a normal distribution of mean = 70, SD = 5; body weight follow normal distribution of mean = 75 kg, SD = 10 kg; treatment: test group = 1, control group = 0, that is, the average treatment effect difference = $\frac{4}{3} - 1$ (test - control); gender: 0 = female, 1 = male.

(ii) at postbaseline visits (from Week 2 to Week 16), the score decreases over time for most cases, the changes from the previous visit (per visit per patient) are modeled as follows:
Change from the previous visit in Score =

$$\frac{75}{\text{BodyWeight}} \frac{\text{BaselineScore}}{70} \frac{(3+\text{Treatment})}{3} \frac{(\text{Gender}+b)(1+\text{ConMed})}{b} \frac{30}{\sqrt{1+Week^2}} + \beta f(\varepsilon) \qquad \text{(A.9)}$$

where ConMed (concomitant medication) changes over time randomly: 1 = Yes (20%), 0 = No (80%), b is the coefficient to determine the importance of gender and ConMed, here $b = 5$ which is chosen empirically; Week = 2, 4, 6, 8, 10, 12, 14, 16, the coefficient of 30 for week is chosen empirically to determine the magnitude of the change from previous visit data; the error function $f(\varepsilon)$ follows a standard normal distribution, β is the coefficient of error term which determines the data pattern, that is, for MNAR (dropout due to lack of efficacy) $\beta = -3.5$ (and the absolute value of $f(\varepsilon)$ is used); for the completers or MAR (dropout due to lost to follow-up), $\beta = 5$ (without an absolute function), the values for coefficients are chosen empirically.

(iii) The change from baseline at each visit is calculated as accumulation of all changes from previous visit; the absolute score at postbaseline visits is calculated as baseline + change from baseline. For the dropouts, the monotone missingness started from Week 8, Week 10, or Week 12 according to the simulation scenario.
`SAS version 9.4` is used for the data generation.

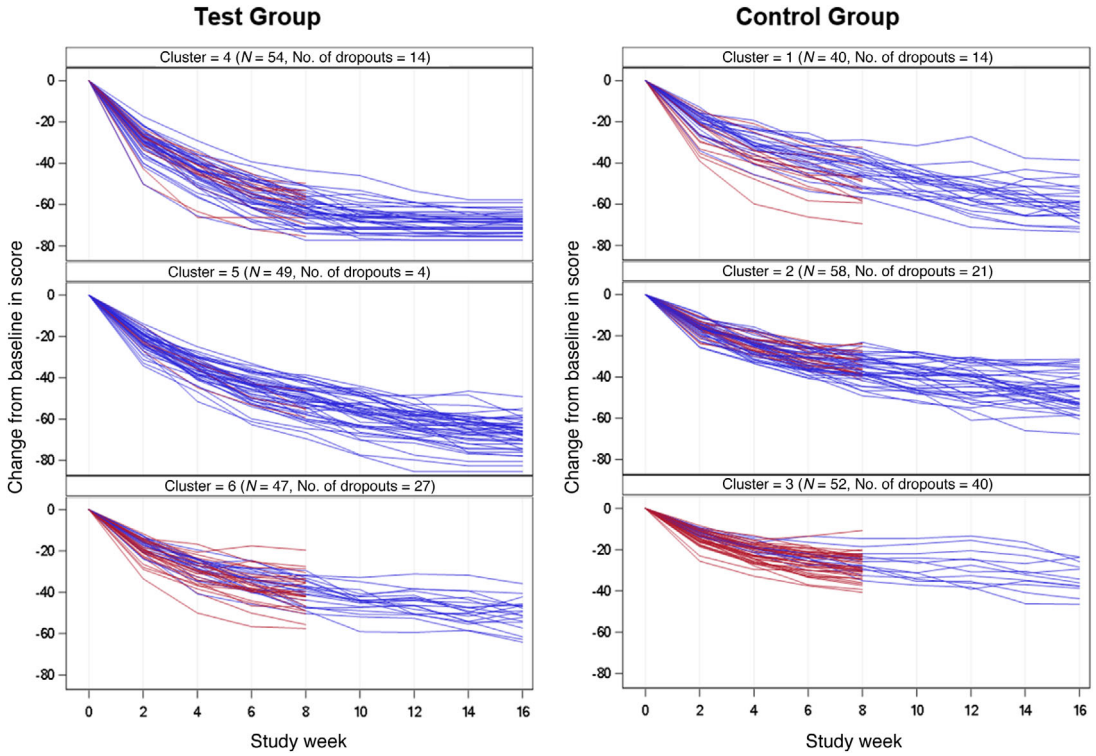

FIGURE A.3 Simulation data: patient profile by the treatment group and cluster (Scenario 300-40-10). Blue lines are for completers, and red lines are for dropouts

A.3 | Longitudinal clustering results in the simulation study

Within each treatment group, patients are clustered into three categories: "good responder," "medium responder," and "low responder" according to their longitudinal efficacy profiles. The individual patient profiles by cluster are provided in Figure A.3 for simulation Scenario 300-40-10.