Bioinformatics, 37(20), 2021, 3538–3545
doi: 10.1093/bioinformatics/btab322
Advance Access Publication Date: 13 May 2021
Original Paper

Systems biology

Evaluation of categorical matrix completion algorithms: toward improved active learning for drug discovery

Huangqingbo Sun [1] **and Robert F. Murphy** [1,2,3,4,*]

[1]Computational Biology Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA, [2]Department of Biological Sciences, Carnegie Mellon University, Pittsburgh, PA 15213, USA, [3]Department of Biomedical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213, USA and [4]Machine Learning Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA

*To whom correspondence should be addressed.

Associate Editor: Jinbo Xu

Received on August 28, 2020; revised on April 5, 2021; editorial decision on April 25, 2021; accepted on April 29, 2021

Abstract

Motivation: High throughput and high content screening are extensively used to determine the effect of small molecule compounds and other potential therapeutics upon particular targets as part of the early drug development process. However, screening is typically used to find compounds that have a desired effect but not to identify potential undesirable side effects. This is because the size of the search space precludes measuring the potential effect of all compounds on all targets. Active machine learning has been proposed as a solution to this problem.

Results: In this article, we describe an improved imputation method, Impute by Committee, for completion of matrices containing categorical values. We compare this method to existing approaches in the context of modeling the effects of many compounds on many targets using latent similarities between compounds and conditions. We also compare these methods for the task of driving active learning in well-characterized settings for synthetic and real datasets. Our new approach performed the best overall both in the accuracy of matrix completion itself and in the number of experiments needed to train an accurate predictive model compared to random selection of experiments. We further improved upon the performance of our new method by developing an adaptive switching strategy for active learning that iteratively chooses between different matrix completion methods.

Availability and implementation: A Reproducible Research Archive containing all data and code is available at http://murphylab.cbd.cmu.edu/software.

Contact: murphy@cmu.edu

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Drug development is an expensive and lengthy process. Potential drugs can be initially identified by screening on a specific biological target, but these drugs often have unfavorable side effects due to the complex network interactions within cells and tissues (Lounkine *et al.*, 2012). These are typically not discovered until later in the drug development process because the throughput even of automated systems is not sufficient to screen all potential drugs for all potential effects. The use of active machine learning has been proposed as a potential solution to this combinatorial problem (Murphy, 2011). Active learning is the subject of extensive previous research (Balcan *et al.*, 2009; Cohn *et al.*, 1995) has been successfully used for large scale problems in experimental sciences (Coley *et al.*, 2020; Hinkson *et al.*, 2020; Reker, 2020).

In the context of drug screening, active learning has been shown to reduce the amount of screening needed when considering a single target (Warmuth *et al.*, 2003). It can also provide dramatic reductions in the number of required experiments for the larger problem of many drugs and many targets (Kangas *et al.*, 2014; Naik *et al.*, 2013), and for cases in which the possible phenotypes are not known in advance (Naik *et al.*, 2016). The underlying computational problem in this application is matrix completion, the prediction of unmeasured elements in a matrix of drugs and targets. There are two settings in which this may take place. In the first, numerical features are available that are believed to be accurate descriptors of the properties of both the drugs and the targets [the setting explored in Kangas *et al.* (2014)]. There are a number of approaches for this setting (Chiang *et al.*, 2015; Huang *et al.*, 2017; Wang and Elhamifar, 2018). The second, more difficult setting, is when such features are not available. For example, in the setting explored by Naik *et al.* (2016), the targets were different fluorescently-tagged clones of the same cell line for which there was no way to measure similarity in advance. In this case, imputation must be based upon similarities in the observations among drugs or targets. Within this setting, the observations may either be numerical or categorical. For numerical values, a number of methods have been described (Candes and Plan, 2010; Candes and Tao, 2010; Mazumder *et al.*, 2010). For categorical values, alternative methods are needed since there are no mathematical relationships between the categorical variables (Cao and Xie, 2015; Davenport *et al.*, 2014). For the drug-target (or condition-target) problem, Naik *et al.* (2013, 2016) introduced methods based on grouping independent variables to predict consistent categorical phenotypes. This approach finds 'Target-types' that have a similar response to conditions and 'Condition-types' that have similar effects on targets. For unmeasured experiments, predictions are made by considering the measured experiments that have the same 'Target-type' or 'Condition-type'. While these methods in combination with active learning were demonstrated to reduce the number of experiments needed to achieve high accuracy, there is significant room for improvement, especially when limited data have been acquired. Chen *et al.* (2020) adapted the SOFT IMPUTE approach (Mazumder *et al.*, 2010) to the categorical framework and showed improved active learning performance for both synthetic data and the data acquired by Naik *et al.* (2016).

In the current work, we introduce an alternative 'lazy learning' method for categorical matrix completion, Impute by Committee (IBC). Lazy learning refers to deferring the construction of a predictive model until it is needed. In this case, rather than grouping drugs or targets and constructing models to make predictions for each entire group, our algorithm builds separate imputation models for each unmeasured experiment using whatever existing experimental measurements are relevant. We evaluate this approach and three previously described algorithms both for the task of categorical matrix completion and in the context of active learning of drug-target models using both synthetic and real datasets. The results demonstrate that overall IBC provides better performance for both the completion and active learning tasks. We also describe an adaptive switching strategy which estimates the properties of partially observed matrices and uses them to pick an appropriate algorithm for each round of experiments. This results in improved performances on both synthetic and real datasets over using a single algorithm.

2 Materials and methods

2.1 Problem definition

Following Naik *et al.* (2016), we use the term 'condition' as a general term that not only includes drugs and small molecule compounds but also manipulations like gene knockouts or knockdowns, addition of hormones or growth factors, or changes in experimental environment. Similarly, 'targets' is considered to include any biological entity, such as proteins, cells or tissues. We use a finite categorical set T that includes all targets under investigation, $T = \{t_i, i = 1, 2, \ldots, m\}$, where m is the total number of targets being considered; similarly, C is a finite categorical set $C = \{c_j, j = 1, 2, \ldots, n\}$, where n is the total number of conditions being considered. We use the term 'phenotype' to describe the effect for a drug on a particular target, and define the collection of all the phenotypes possible from any combination of condition and target as $P = \{p_{i,j}\}$, for condition$_j$ and target$_i$. Thus condition and target are independent variables, and phenotype is the dependent variable. The experimental space is $E = T \times C$, $E = \{e_{i,j}\}$. Here, $e_{i,j}$ refers to the experiment on t_i under c_j. We define a function $P(e) = p$ which maps an element in E to its phenotype.

The problem is to iteratively create an imputation model for missing phenotypes and use them to choose experiments to perform in order to improve the model. The premise of the imputation, as with all matrix completion methods, is that latent similarities exist in the matrix (i.e. within the experimental space). That is, we assume that there are similarities between targets in T and between conditions in C. This makes sense from a biological perspective: a given drug may perturb multiple targets in a similar manner, and a family of compounds may affect a given target in a similar way. As discussed in Section 1, we consider the case in which there are no features available to permit a comparison of targets (or conditions) to each other. Thus the phenotypes are the only basis for comparison; we say that two experiments are similar if they share the same phenotype. In the learning process, a set of observed experiments, O, is defined as a set of all observed experiments in the experimental space formed by the independent variables. The similarity between conditions themselves or targets themselves can be estimated if they have co-observed experiments under the same target or condition. That is, we consider two targets to be similar if the phenotypes of the experiments of these two targets under the same condition are the same. The more co-observed experiments have the same phenotype, the higher the similarity between these two targets. Similarly, we estimate the similarity of two conditions from co-observed experiments on the same target.

2.2 Imputation methods

Below we describe a novel categorical matrix completion algorithm, IBC [briefly introduced in Sun and Murphy (2020)] and briefly summarize three other previously described categorical matrix completion algorithms. Two rely on mapping the categorical matrix to a real valued matrix and solving the real valued matrix by SOFT IMPUTE (Chen *et al.*, 2020; Mazumder *et al.*, 2010) or nuclear norm regularized log-likelihood function maximization algorithm (NNRLLFM) (Cao and Xie, 2015; Davenport *et al.*, 2014; Klopp *et al.*, 2015; Lafond *et al.*, 2014). The other, like IBC, relies on learning a partition of E based on similarities between conditions or targets, and imputing the missing value with the partitioned clusters.

2.2.1 Imputation by committee

Here, we describe a committee voting imputation method for making predictions of unobserved e in E. Previous work by Naik *et al.* (2013) employs a condition-specific distribution for imputation in the training step; our current model adopts instead a lazy learning method that introduces no bias before making a prediction for an individual experiment. By lazy learning, a full use of the observed information is achieved and the model is allowed to be robust to the noise in observed data.

We use the term 'condition vector' to refer to the series of experiments under one particular condition, that is, $\hat{c}_j = \{e_{1,j}, \ldots, e_{m,j}\}$. Similarly, we define 'target vector' as the series of experiments for one particular target, $\hat{v}_i = \{e_{i,1}, \ldots, e_{i,n}\}$. Here, we define the conflict ζ and consistency ρ between two vectors $\hat{v}_1, \hat{v}_2$ as:

$$\zeta(\hat{v}_1, \hat{v}_2) = \sum_i \mathbb{I}(\widehat{v_{1,i}} \in O)\mathbb{I}(\widehat{v_{2,i}} \in O)\mathbb{I}(P(\widehat{v_{1,i}}) \neq P(\widehat{v_{2,i}})), \quad (1)$$

$$\rho(\hat{v}_1, \hat{v}_2) = \sum_i \mathbb{I}(\widehat{v_{1,i}} \in O)\mathbb{I}(\widehat{v_{2,i}} \in O)\mathbb{I}(P(\widehat{v_{1,i}}) = P(\widehat{v_{2,i}})). \quad (2)$$

Here, $\mathbb{I}(*)$ is the indicate function.

As discussed in the Section 2.1, conditions and targets in our problem setting are assumed to have some relationships to each other. Therefore, imputing the unobserved e can be realized from these relationships. The algorithms are shown in Algorithms 1 and 2 and illustrated in Figure 1. First, we consider imputation from conditions by calling impute$_{\text{all}}$(cond$\hat{\text{i}}$tions), with a return set predictions$_{\text{from}}$condition. Here, cond$\hat{\text{i}}$tions is the set of condition vectors. For a given vector $\hat{v}$, we define the collection of vectors with n conflicts to be $\hat{v}$ as committee $\mathcal{C}_{\hat{v}}(n)$. Imputation for a given vector is done by constructing a committee with similar vectors and then the prediction for each unobserved element in the given vector is chosen by voting among the observed elements in the corresponding position (elements with the same index) of the member vectors $\hat{m}$ in the committee. We define a function to evaluate the similarity of two given vectors that penalizes predictions that are derived from conflicts

$$\text{score}(\hat{v}_1, \hat{v}_2) = \rho(\hat{v}_1, \hat{v}_2) - pnl \times \phi \times \zeta(\hat{v}_1, \hat{v}_2), \quad (3)$$

where ϕ is the fraction of experimental space that has been observed. As this penalty increases, we expect the behavior to

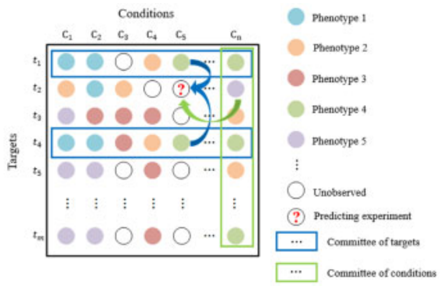

Fig. 1. Illustration of the Imputation by Committee algorithm. The experimental space is represented as a categorical matrix. Unobserved phenotypes are predicted from similar targets and conditions. Specifically, to predict the phenotype of an experiment (a combination of target and condition), we create a cluster (committee) of targets that are similar to the target, and a cluster (committee) of conditions that are similar to the condition. The predicted phenotype is selected by voting from the corresponding observed phenotype in the committees

Algorithm 1. impute_all($\mathrm{vector_{set}}$)

```
1: predictions ← $\varnothing$
2: for $\hat{v}$ in $\mathrm{vector_{set}}$ do                                    #
   $\mathrm{vector_{set}}$ can be the set of target vectors or condition vectors in E
3:   $n_{\mathrm{conflict}} \leftarrow 0$
4:   $U = \{e | e \in \hat{v}, e \notin O\}$
5:   while $O \neq \varnothing$ do
6:     $\mathcal{C}(n_{\mathrm{conflict}}) \leftarrow \{\hat{m}_1, \hat{m}_2, \ldots\}$
7:     $\mathrm{pred}_e F \leftarrow \mathrm{infer}(e, \hat{v}, \mathcal{C}(n_{\mathrm{conflict}}))$
8:     if $\mathrm{pred}_e \neq \varnothing$ then
9:       remove $e$ out of $U$
10:       predictions $\cup\ \mathrm{pred}_e$
11:     end if
12:     $n_{\mathrm{conflict}} \leftarrow n_{\mathrm{conflict}} + 1$
13:   end while
14: end for
15: return predictions
```

Algorithm 2. infer(e,$\hat{v}$,$\mathcal{C}$)

```
1: $\mathrm{pred}_e \leftarrow \varnothing$
2: for $\hat{m}$ in $\mathcal{C}$ do
3:   if $m_* \in O$ then                                                    #
     here, $m_*$ is the element in $\hat{m}$ with the same index as $e$ in $\hat{v}$
4:     if $((P(m_*), \text{anyscore}) \notin \mathrm{pred}_e$ then                         #
       P(*) is the function mapping $e$ to its phenotype
5:       $\mathrm{pred}_e \cup \{(P(m_*), \mathrm{score}(\hat{v}, \hat{m}))\}$                      #
       'anyscore' can be any value, we only care about $P(m_*)$
6:     else
7:       replace $(P(m_*), \text{anyscore})$ with $(P(m_*), \text{anyscore} + \mathrm{score}(\hat{v}, \hat{m}))$
8:     end if
9:   end if
10: end for
11: return $\mathrm{pred}_e$
```

approach the Greedy Merge (GM) algorithm (see below). We explored values of this penalty from 0.5 to 5 and observed only small changes (typically differences of only 1 or 2 rounds required to achieve 90% accuracy). We therefore set it to 1 for all studies reported here.

Symmetrically, imputation can also be done by predicting the unobserved experiments from the targets by calling $\mathrm{impute_{all}}(\widehat{\mathrm{targets}})$, with a return set $\mathrm{predictions_{from}target}$. Here, $\widehat{\mathrm{targets}}$ is the set of target vectors. Both imputations are combined with equal weight. That is, we merge the set $\mathrm{predictions_{from}condition}$ and the set $\mathrm{predictions_{from}target}$; the predictive candidate phenotypes for some experiments may appear in both imputation sets, the predictive score for these phenotypes is the sum of the corresponding scores from the two sets. For convenience, we define Ps_e as the set of all candidate phenotypes of e. For each unobserved e, the predictive decision is realized by:

$$P(e) = \underset{p \in Ps_e}{\operatorname{argmax}}\, \mathrm{score}_e(p), \tag{4}$$

where $\mathrm{score}_e(p)$ is the corresponding predictive score of the candidate phenotype p for e.

2.2.2 Soft IMPUTE-based method

This algorithm (SIB) was first proposed by Chen *et al.* (2020). The idea is to encode the E, a categorical matrix with K phenotypes, into K 'affiliation' matrices using a 'one-versus-rest' fashion (Aly, 2005). The values in each 'affiliation' matrix are 1 for observed 'on' elements, 0 for observed 'off' elements and 0.5 for unobserved elements, defined in equation (3) in Chen *et al.* (2020). The 'affiliation' matrix completion is realized by the SOFT IMPUTE algorithm implemented with an iterative and singular value thresholding fashion (Cai *et al.*, 2010).

2.2.3 NNRLLFM-based method

The NNRLLFM was first proposed by Davenport *et al.* (2014) for binary categorical matrix completion. The scope of this algorithm was extended to matrices with more than two categories (Cao and Xie, 2015; Klopp *et al.*, 2015; Lafond *et al.*, 2014). NNRLLFM relies on mapping a real value matrix X to a corresponding partial observed categorical matrix M with K link functions, f_k, $k = 1, \ldots, K$, which returns the probability of the element $M_{i,j}$ being the kth category, p_k. Here, $M_{*,*}$ refers to one of the unobserved elements in M. Formally, it can be written as

$$p_k = f_k(X_{i,j}). \tag{5}$$

The completion of M is achieved by solving the constraint optimization problem shown in (6)

$$\hat{M} = \operatorname{argmax}_X \sum_{(i,j)\in\Omega} \sum_{k=1}^{K} \mathbb{I}(M_{i,j} = k) \log(f_k(X_{i,j})), \tag{6}$$

s.t. $\|X\|_* \leq \alpha\sqrt{rd_1 d_2}$ and $\|X\|_\infty \leq \alpha$.
Here, $\mathbb{I}(*)$ refers to the indication function, $\|X\|_*$ is the nuclear norm of matrix X, $\|X\|_\infty$ refers to the infinity norm of matrix X, d_1 and d_2 refer to the size of matrix X and r refers to the rank of matrix X.
Inspired by Chen *et al.* (2020), we follow the setting of encoding the E into K 'affiliation' matrices and recovering them by NNRLLFM. The values in each 'affiliation' matrix are 1 for observed 'on' elements, -1 for both observed 'off' elements and unobserved elements. The link functions we choose are

$$\begin{cases} f_1 = \dfrac{1}{1 - e^{-x}} \\ f_{-1} = \dfrac{e^{-x}}{1 - e^{-x}} \end{cases}. \tag{7}$$

2.2.4 Greedy merge

GM is a structure learning algorithm first proposed by Naik *et al.* (2013). The algorithm iteratively merges the conditions or targets with no conflict on observed elements to construct distributions, a partition of the E. Imputation is made by predicting the phenotype of unobserved elements to be the phenotype of observed elements in the distribution under the same condition or on the same target. Since some elements may not be imputed if there is no observed element under the same condition or on the same target, the method will assign these the majority phenotype and when used for active learning, query such elements as a priority.

2.3 Active learning

For choosing experiments, we used three different variants of uncertainty sampling: least confident, margin and entropy querying (Settles, 2009). Each predicted phenotype was accompanied by a confidence score. We defined two criteria for ranking the confidence of the prediction; one directly using the predictive confidence score of the unobserved e, and the second using the information entropy of each experiment's prediction. Given a score for phenotype p_i for an unobserved e, this is calculated using

$$\mathrm{prob}_{e,p_i} = \frac{\mathrm{score}_e(p_i) + (1 - \mathrm{score}_e(p_e*))}{\sum_{p \in Ps_e}(\mathrm{score}_e(p) + (1 - \mathrm{score}_e(p_e*)))}, \tag{8}$$

where $p_e* = \mathrm{argmin}_{p \in Ps_e}\, \mathrm{score}_e(p)$

$$\mathrm{Entropy}(e) = \sum_{p \in Ps_e} -\mathrm{prob}_p \times \log_2 \mathrm{prob}_p. \tag{9}$$

For the IBC method, we adopted a hybrid query strategy. That is, half of the batch was selected using the least predictive confidence score and the rest is selected by the highest entropy.

For the GM method, we used the least-confident query strategy described in Naik *et al.* (2013). Note that, when used for active learning, the algorithm will first query unpredicted elements, and then query elements for which the imputation was made by the distribution with the fewest observed elements. For the SIB method, a margin sampling query strategy was adopted, as shown in equation (5) in Chen *et al.* (2020); this selects the unobserved elements with the smallest difference between the first- and second-largest affiliation values. For the NNRLLFM-based method, a least-confident query strategy was adopted that selects the unobserved element in the E with the lowest major affiliation value (major refers to the largest affiliation value for one element among all of its affiliation values).

2.4 Datasets

2.4.1 Synthetic datasets

As described previously (Kangas *et al.*, 2014), we constructed synthetic datasets in which experimental spaces consisting of 100 conditions C and 100 targets T are parameterized by a uniqueness u and responsiveness r to describe the distribution of C and T in E. Each target has an unperturbed phenotype; for convenience, we define c_1 as the unperturbed state. The uniqueness ($0 < u <= 1$) of the targets is defined to be the inverse of the number of targets that have the same phenotype for all conditions, and similarly for the conditions. We set the uniqueness of targets and conditions to be the same. The responsiveness ($0 < r <= 1$) of targets is defined as the fraction of targets that change their phenotype from the unperturbed state to another phenotype across all conditions C. To make the simulations more realistic (i.e. comparable to experimental measurements), we optionally added varying amounts of noise. That is, after creating E controlled by a given u and r, 10% or 20% of the total entries in E were selected and their phenotypes changed from their original phenotype to another random one. For tests with random sampling of the experimental space, the same synthetic dataset was used for all methods.

2.4.2 Enzyme activity screening dataset

This dataset contains the experimental results of compound target interaction assays from Drug Target Commons (DTC) platform (Tang *et al.*, 2018). Specifically, considering the assays with the 'biochemical' assay format, 'functional' assay type, 'enzyme activity' assay sub-type, the 'inhibition' endpoint mode of action and the endpoint standard units of percentage, '%', we first selected 389 compounds and 259 targets that had the most assay records (the assay records are available in Supplementary File S1). We further filtered out targets that had recorded assays for fewer than 40% of candidate compounds, yielding 259 targets. We finally filtered out the compounds that had recorded assays with fewer than 60% of the remaining targets. This yielded 210 targets and 195 compounds, giving 40 950 possible entries in E. Of these 39 734 (97.03%) had assay records. We used the endpoint standard value (EPSV) recorded in the DTC dataset to define categorical phenotypes (using the average EPSV for compound target combinations that had multiple assay records). We assigned combinations with the enzyme activity EPSV $<25\%$ as the phenotype 1, $25\% \le \mathrm{EPSV} < 75\%$ as phenotype 2, $75\% \le \mathrm{EPSV} < 125\%$ as phenotype 3 and $125\% \le \mathrm{EPSV}$ as phenotype 4. Among all 39 734 measured assays in E, there are 1941 assays with phenotype 1, 5960 assays with phenotype 2, 31 594 assays with phenotype 3 and 239 assays with phenotype 4.

2.4.3 Protein subcellular patterns screening dataset

To test the candidate algorithms on data from another real experimental setting, we also used the dataset created by Naik *et al.* (2016). Each experiment in that study consisted of imaging an NIH-3T3 cell line expressing a particular fluorescently-tagged protein after treatment with a particular drug. The full experimental setting consisted of 48 cell lines and 48 compounds but we reduced it to 47 cell lines and 46 compounds by removing highly variable experiments. Both cell lines and compounds were silently duplicated (by assigning two indices to each cell line or drug such that the machine learner did not know that they were duplicated) to create a 94×92 matrix (the duplications guarantee that there are similarities in E to learn). In the original study, image features that describe the fluorescence pattern of the tagged proteins were calculated, and these were converted into statistically distinguishable phenotypes. To simulate active learning on this dataset, we returned to the learner the average feature values reported for a given cell line and drug provided in the Supplementary information of the original study (https://doi.org/10.7554/eLife.10047.014).

The phenotypes of the experiments in E were determined by a form of agglomerative hierarchical clustering. Every round n, we trained a hierarchical clustering classifier using the features of all observed images in O_n and a distance threshold of 10 to decide how many clusters to allow in a given round. The phenotypes were learned using only images of observed experiments because we do not have access to the image features of unmeasured experiments in a real scenario. Once the clustering metric was determined, we trained a 5-nearest neighbor classifier, as $P_n(e)$, to assign all unobserved e in E into those clusters. This was used as the ground truth for evaluating models; thus the ground truth phenotype changed from round to round to correspond to the set of phenotypes that the model had seen so far (it obviously could not predict a phenotype it had not yet observed) and the model was re-trained using the phenotypes learned in each round. The workflows of the three types of experiments are shown in Figure 2.

3 Results

3.1 Synthetic data experiment

We first assessed the performances of the four methods on matrix completion for synthetic datasets (as described in Section 2). These were created across 16 combinations of u and r with 32 phenotypes. For each combination of u and r, we generated three independent E with different random seeds and did this with and without different amounts of simulated experimental noise. For each generated E, we performed the whole loop of matrix completion (with batch size of

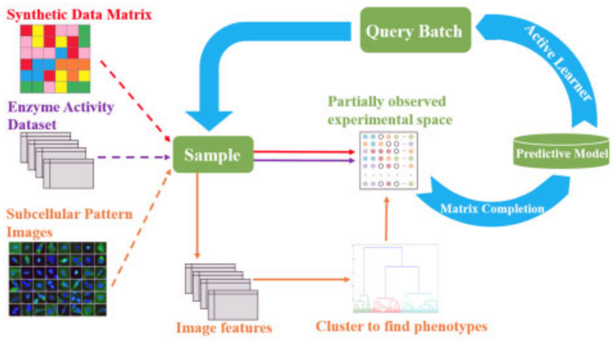

Fig. 2. Workflows of the active learning processes

1%) twice by starting with two different randomly chosen sets of initial observations. Thus, for each combination of u and r, there are six replicate experiments, and the results reported in the following are average results of the six repeats. The results are assessed by the accuracy, class-weighted F1 score and unweighted F1 score for the entire space. In Figure 3, the average accuracies of low and high u, with 20%, 40%, 60% and 80% of the observations, are shown (the numerical values of the accuracy, class-weighted F1 score and unweighted F1 score, along with averages for each method, can be found in Supplementary Files S2–S4). NNRLLFM performance was at or near the worst for all tests when given only 20% of the data and when given 40% for low u. There are no cases for which it performs the best. The SIB method performs poorly for most cases with high u. It is competitive for other cases but the only cases for which is performs slightly better than the other methods are those with large amounts of training data. GM performs the best of all methods for high u and low amounts of training data. It is competitive for other cases. IBC performs the best for almost all cases with low u, regardless of the amount of training data. It is second to GM in most of the high u cases.

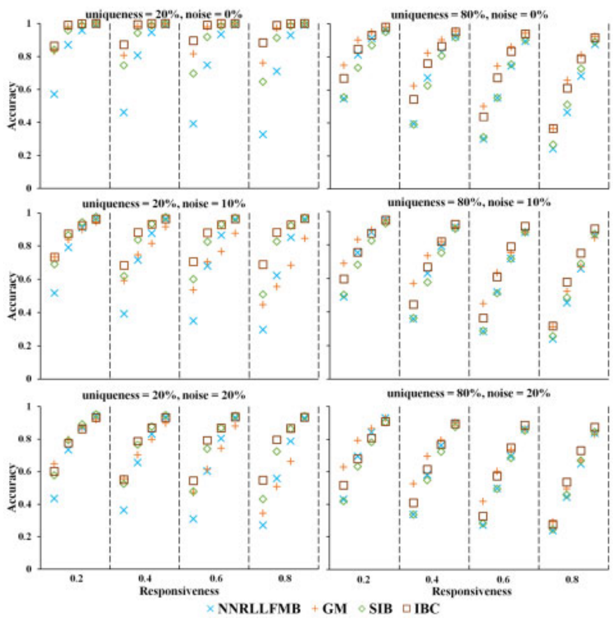

Fig. 3. Performance of categorical matrix completion methods for different synthetic datasets. Each symbol shows the full space accuracy averaged over six replicates with different random seeds and initial samples (the average standard deviations for each method across all rounds were <2%; see Supplementary Table S1). Each panel shows a given combination of noise and u, with subpanels for different values of responsiveness. Within each subpanel, results are shown for completion given a randomly chosen 20%, 40%, 60% and 80% of all observations (from left to right). Results for the NNRLLFM-based method are shown in blue, GM in orange, SIB method in green and IBC in brown

Overall, no single method performs the best for all cases, but the clustering-based methods, GM and IBC, are the best overall performers. When we consider averages for all combinations of u and r, IBC performs with an average accuracy of 79.3% followed by GM with an average accuracy of 76.3%. SIB is next with the average accuracy of 72.9%, and NNRLLFM is last with the average accuracy of 68.5%. Thus IBC outperforms all of the previously described methods.

We next considered the ability of these methods to guide active learning. For each round of active learning, we used the trained models as the basis for selecting experiments using uncertainty sampling in batches of 1% of E. Synthetic datasets were constructed as for Figure 3. For our tests of the SIB method, we configured SOFT IMPUTE to use a maximum number of iterations of 100 and a soft singular value threshold of $\frac{s}{50}$, where s is the largest singular value in each iteration. We also dynamically adjusted the number of affiliation matrices. Chen *et al.* (2020) used k affiliation matrices, where k was set to the number of phenotypes in the entire dataset. In an active learning setting in which not all data are available, the number of possible phenotypes is typically not known. We therefore used only k' affiliation matrices in our implementation of their method, where k' is the number of phenotypes observed up to that round (this approach was also used for our implementation of the NNRLLFM-based method). These implementation differences may account for the slight differences in results we obtained compared to those reported in Chen *et al.* (2020). Results of accuracy for six repeats are shown for low and high u in Figure 4 (the numerical values of accuracy, class-weighted F1 score and unweighted F1 score can be found in Supplementary Files S5–S7). The IBC method and the SIB method both performed well. For all non-noise cases and noisy cases except for some low-responsiveness cases, the IBC model

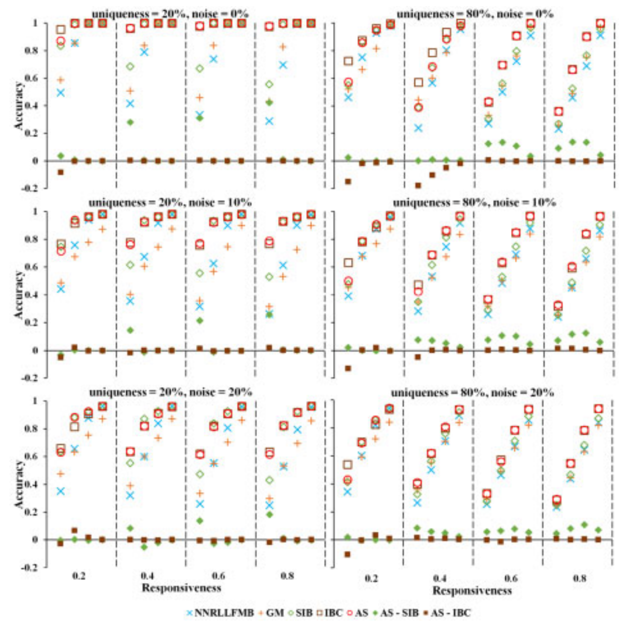

Fig. 4. Performance of matrix completion models for active learning with synthetic datasets. Each symbol shows the full space accuracy averaged over six replicates with different random seeds and initial samples (the average standard deviations for each method across all rounds were all <2%; Supplementary Table S1). Each panel shows a given combination of noise and u, with subpanels for different values of r. Within each subpanel, results are shown for completion given 20%, 40%, 60% and 80% of all observations selected by active learning (from left to right). Results for the NNRLLFM-based method are shown in blue, GM in orange, SIB method in green and IBC in brown. Shown in red are results for the adaptive switching strategy (see text). The improvement in accuracy of the switching strategy over IBC is shown with brown diamonds and its improvement over SIB is shown with green diamonds. Some improvement is seen for the switching method compared to SIB, but little is seen compared to IBC alone. Note that the green diamond being above the brown diamond indicates that IBC alone outperforms SIB alone. The numerical values can be found in Supplementary File S5

performs better than all other methods. Also, the SIB method performed better at the beginning stage of some noisy, low-responsiveness cases. As seen in Supplementary File S5, the average accuracy for IBC is again the best at 83.2%, SIB is next at 76.6%, followed by NNRLLFM and GM at 66.6% and 66.2%, respectively.

The fact that the models' performances were related to *r* and the fraction of observations available suggested that an adaptive strategy could be used to decide which completion method to use. We therefore performed studies similar to those in Figure 3 for *u* from 10% to 90% (in steps of 10%), *r* from 20% to 100% (in steps of 20%), added noise from 0% to 25% (in steps of 5%), number of phenotypes of 4, 8, 12, 24, 36, 48 and fraction observed from 1% to 100% (in steps of 1%) to collect the predictive accuracies of the IBC methods and the SIB methods upon varied matrices. Then, we generated polynomial features (max degree of 4) for the independent values (*u*, *r*, number of phenotypes, noise and observation fraction) using the PolynomialFeatures function from the scikit-learn preprocessing package. We divided the data into a training set and test set at a ratio of 4:1. Using the training set, we performed a linear regression between the extended polynomial features and the matrix completion accuracy for the IBC method and the SIB method, respectively. The linear regression accuracies on the test set for the IBC method and the SIB method were both 94.1%. With these pre-trained linear regression models, we carried out a denser grid search of *u* from 10% to 90% (step size of 10%), *r* from 10% to 90% (step size of 10%), noise from 0% to 25% (step size of 5%), number of phenotypes from 2 to 48 (step size of 2) and observation fraction from 1% to 100% (step size of 1%) to predict the matrix completion accuracies under the given matrix features and the observation fraction. After that, we learned a decision tree to use the *r* and observation fraction (independent values) to predict the optimal method (the method with the higher matrix completion accuracies between the IBC method and SIB method). The tree is shown in Figure 5. Using this decision tree, we implemented an adaptive switching strategy for categorical matrix completion that estimates the *r* of the partially observed matrix and uses it and the observation fraction to choose the method for matrix completion. The performances of this approach on synthetic data are also shown in Figure 4. The adaptive switching strategy has an average accuracy of 82.0% (Supplementary File S5), better than SIB alone and almost equal to IBC alone.

3.2 Enzyme activity prediction experiment

Having characterized the methods using synthetic datasets, we next compared them on a real experimental dataset for prediction of enzyme activity. For this experiment, we performed five repeats of the entire learning loop with different randomly selected starting samples and report the average results here. The performances assessed by the accuracy of the four matrix completions methods in active learning are shown in Figure 6 (the performances assessed by class-weighted F1 score and unweighted F1 score can be found in Supplementary Figs S1 and S2). The SIB model performs the best, with the IBC model performing second best. The better performance of the SIB method is consistent with the synthetic dataset results, since for this dataset the *r* of *E* is low (with only four phenotypes and with the frequency of the majority phenotype being more than 80%) which corresponds to cases in which the SIB model performed relatively better. However, the adaptive switching strategy, which starts with the IBC method and switches to the SIB method after observing 13.5% of *E*, catches up to the performance of the optimal method quickly.

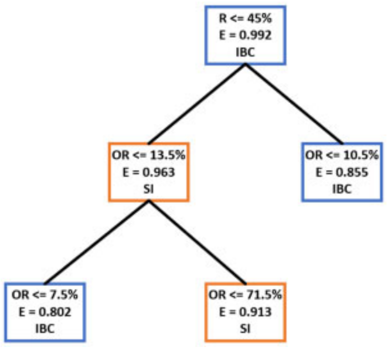

Fig. 5. The decision tree for the adaptive switching setting. Here, OR refers to the observation fraction, R refers responsiveness and E refers to entropy of each node (node purity)

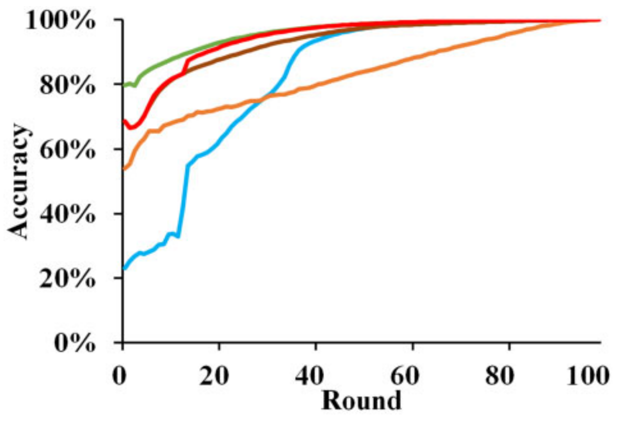

Fig. 6. Learning curves for active learning using different categorical matrix completion models on the enzyme activity screening dataset. Active learning was done in batches of 400 experiments (~1% of the full *E*). The average standard deviations for each method for all rounds were <1.7%. The blue curve refers to the NNRLLFM-based method, the orange curve refers to the GM method, the green curve refers to the SIB method, the brown curve refers to the IBC method and the red curve refers to the adaptive switching strategy

We also performed tests using each of the models to make predictions but using randomly selected experiments [Supplementary Fig. S3 (accuracy), Supplementary Fig. S4 (class-weighted F1 score) and Supplementary Fig. S5 (unweighted F1 score)]. IBC and SIB perform better when used with active learning than with random selection. However, the other two models show little difference between active and random learning. This may be expected for the NNRLLFM-based method but perhaps not for the GM algorithm given its good performance on synthetic datasets. However, it can be explained by the fact that the active GM algorithm is configured to always query instances that do not perfectly match any of the other rows or columns (as described in the Section 2) but in this dataset, most of the rows and columns cannot be matched perfectly. This disadvantage can also be seen in noisy synthetic datasets in Figure 4. The best area under the learning curve (an estimate of the overall performance, equal in this case to the average accuracy across all rounds) was 96.2% for SIB. The switch strategy performs next best at 94.7%, followed by IBC at 93.4%. GM and NNRLLFM are last at 83.4% and 82.8%, respectively.

3.3 Subcellular pattern prediction experiment

As a further test of performance of the different models on real experimental datasets, we compared them on a high throughput screening image dataset consisting of the average features of fluorescent microscopy images for GFP-tagged cells treated with various drugs. After each round of data collection, we first compared the predictions of the model made for unobserved experiments with the ground truth of the *E* we generated for that round (this is necessary because we cannot penalize a learner for not predicting a phenotype that has not yet been observed in any experiment). As with the enzyme activity dataset, the entire experiment was repeated five times. Learning curves of the accuracy are shown in Figure 7 (learning curves assessed by class-weighted and unweighted F1 score can be found in Supplementary Figs S6 and S7). It is important to note, however, that our tests were done with only the average feature values for each combination of target and condition (see Section 2), and therefore the learning rates are not directly comparable to those reported previously. The results are also not directly comparable to those of Chen *et al.* (2020) because those results used the phenotypes learned in each round by the original study rather than learning the

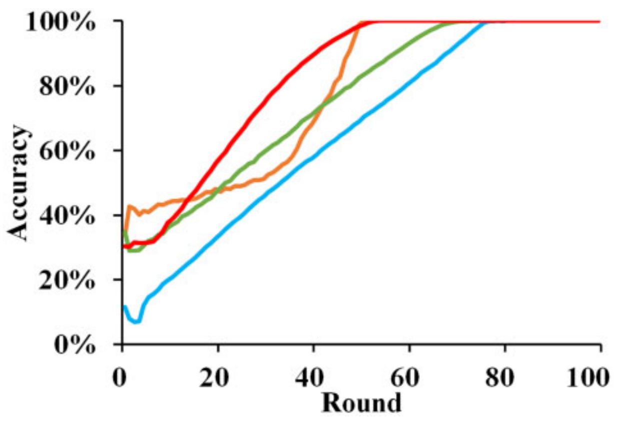

Fig. 7. Learning curves of categorical matrix completion models on the subcellular patterns dataset. The average standard deviations for each method for all rounds were <2.4%. The blue curve refers to the NNRLLFM-based method, the orange curve refers to the GM method, the green curve refers to the SIB method, the brown curve refers to the IBC method and the red curve refers to the adaptive switching strategy

phenotypes anew only from the data observed up to that round (thus taking advantage of information from experiments that were not observed). To make our performance comparisons fair, we evaluated all algorithms while modeling only observed phenotypes.

In this dataset, targets and drugs are each duplicated. Thus u is ∼25%. As seem with synthetic datasets with low u, IBC performs better than the other three model designs on this real dataset. After 42 and 54 rounds, the IBC-based model can learn a predictive model with 90% and 100% accuracy, respectively. The GM model also shows good performance on training to 100% accuracy, using only 51 rounds. However, its performance lags the IBC model for the first 49 rounds. The other single methods performed significantly worse. As expected, the adaptive switching strategy performs identically to the IBC method because it starts with IBC and decides at each round to continue with it (there are slight differences early on due to differences in random initialization). As measured by the area under the learning curve, IBC and the switching strategy are the best, both of them achieving 82.3%. GM performs next best at 77.8%, followed by SIB at 75.6% and NNRLLFM at 65.6%.

Supplementary Figure S8 shows the differences in accuracy by active and random learning for each model (the results assessed by class-weighted and unweighted F1 score can be found in Supplementary Figs S9 and S10, respectively). Clearly, the IBC active model show better performance throughout sampling. The active GM model performs worse than the random GM model but the accuracy increases rapidly after 40% of E has been measured.

As discussed above, every $e \in E$ has four replicates; we refer to a unique e and its replicates as a quad. Thus, there are a total of 47 × 46 quads in E. The learning rate in this setting should clearly depend on how well the learning avoids sampling multiple times in each quad (Naik *et al.*, 2016). That is, a good learner should be able to find out which experiments are replicated with a minimum number of trials; ideally, it would sample every unique unobserved e once. Supplementary Figure S11 shows the discovery rate of quads. The IBC model discovers all unique experiments using only 46 rounds which is the least among all model designs. Moreover, IBC measures most of quads only once at the early stage of the whole learning process unlike other model designs. This reveals the basis for the superior performance of the IBC model: it wastes less effort on measuring replicate experiments.

4 Conclusion

In this article, we have introduced and characterized an improved imputation method suitable for learning latent similarities for categorical matrix completion. It shows superior performance over previous methods both for matrix completion given a certain amount of data and when used for active learning. This performance improvement was observed on many cases of simulated datasets and on one real experimental data. However, for one experimental dataset, an earlier method, based on SOFT IMPUTE, was observed to perform somewhat better. Therefore, we also have developed and characterized an adaptive switching active learning approach. The adaptive switcher is trained to predict which matrix completion method will perform better given current data, and then switches method as necessary as the active learning proceeds. The adaptive switching strategy trades off slightly poorer results compared to using our new IBC method for most completion problems in order to avoid weaker results on other problems. Thus, it would be expected to give the best overall performance for a novel experimental setting.

As discussed in Section 1, active learning for drug discovery has been successfully driven either by models that use features to describe drugs and targets (Kangas *et al.*, 2014; Lang *et al.*, 2016; Reker *et al.*, 2017) or by models of the categorical type described here (Naik *et al.*, 2016). We suggest that the feature-driven approach be used when well-characterized compounds and targets are being considered, but that the categorical approach may be preferable (at least initially) when exploring novel chemical or target spaces. The categorical approaches are also expected to be necessary when considering experimental variables for which appropriate features are not available, such as for patients in pharmaco-genomic studies when a specific set of relevant genes to measure patient similarity is not known in advance. In any case, we expect that improved active learning algorithms for both scenarios will play an increasingly important role in data-driven drug discovery in the future.

Conflict of Interest: none declared.

Acknowledgements

This work was supported in part by grant P41 GM103712 from the National Institutes of Health. We would like to thank Mark A. Davenport and his colleagues for providing the open source One-Bit Matrix Completion (1BMC) Toolbox.

References

Aly,M. (2005) Survey on multiclass classification methods. *Neural. Netw.*, **19**, 1–9.

Balcan,M.-F. *et al.* (2009) Agnostic active learning. *J. Comput. Syst. Sci.*, **75**, 78–89.

Cai,J.-F. *et al.* (2010) A singular value thresholding algorithm for matrix completion. *SIAM J. Optim.*, **20**, 1956–1982.

Candes,E.J. and Plan,Y. (2010) Matrix completion with noise. *Proc. IEEE*, **98**, 925–936.

Candes,E.J. and Tao,T. (2010) The power of convex relaxation: near-optimal matrix completion. *IEEE Trans. Inf. Theory*, **56**, 2053–2080.

Cao,Y. and Xie,Y. (2015) Categorical matrix completion. In: *2015 IEEE 6th International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, IEEE, pp. 369–372.

Chen,J. *et al.* (2020) Categorical matrix completion with active learning for high-throughput screening. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics. doi: 10.1109/TCBB.2020.2982142*

Chiang,K.-Y. *et al.* (2015) Matrix completion with noisy side information. In: *Advances in Neural Information Processing Systems*, pp. 3447–3455.

Cohn,D.A. *et al.* (1995). Active learning with statistical models. In: *Advances in Neural Information Processing Systems*, pp. 705–712.

Coley,C.W. *et al.* (2020) Autonomous discovery in the chemical sciences part I: progress. *Angew. Chem. Int. Ed.*, **59**, 22858–22893.

Davenport,M.A. *et al.* (2014) 1-bit matrix completion. *Inf. Inference J. IMA*, **3**, 189–223.

Hinkson,I.V. *et al.* (2020) Accelerating therapeutics for opportunities in medicine: a paradigm shift in drug discovery. *Front. Pharmacol.*, **11**, 770.

Huang,L. *et al.* (2017) Matrix completion with side information and its applications in predicting the antigenicity of influenza viruses. *Bioinformatics*, **33**, 3195–3201.

Kangas,J.D. *et al.* (2014) Efficient discovery of responses of proteins to compounds using active learning. *BMC Bioinformatics*, **15**, 143.

Klopp,O. *et al.* (2015) Adaptive multinomial matrix completion. *Electron. J. Stat.*, **9**, 2950–2975.

Lafond,J. *et al.* (2014) Probabilistic low-rank matrix completion on finite alphabets. In: *Advances in Neural Information Processing Systems*, pp. 1727–1735.

Lang,T. *et al.* (2016) Feasibility of active machine learning for multiclass compound classification. *J. Chem. Inf. Model.*, **56**, 12–20.

Lounkine,E. *et al.* (2012) Large-scale prediction and testing of drug activity on side-effect targets. *Nature*, **486**, 361–367.

Mazumder,R. *et al.* (2010) Spectral regularization algorithms for learning large incomplete matrices. *J. Mach. Learn. Res.*, **11**, 2287–2322.

Murphy,R.F. (2011) An active role for machine learning in drug development. *Nat. Chem. Biol.*, **7**, 327.

Naik,A.W. *et al.* (2013) Efficient modeling and active learning discovery of biological responses. *PLoS One*, **8**, e83996.

Naik,A.W. *et al.* (2016) Active machine learning-driven experimentation to determine compound effects on protein patterns. *Elife*, **5**, e10047.

Reker,D. (2020) Practical considerations for active machine learning in drug discovery. *Drug Discov. Today Technol.*, **32–33**, 73–79.

Reker,D. *et al.* (2017) Active learning for computational chemogenomics. *Future Med. Chem.*, **9**, 381–402.

Settles,B. (2009) Active learning literature survey. *Technical report*. University of Wisconsin-Madison Department of Computer Sciences.University of Wisconsin.

Sun,H. and Murphy,R.F. (2020) An improved matrix completion algorithm for categorical variables: application to active learning of drug responses. In: *ICML 2020 Workshop on Real World Experiment Design and Active Learning*.

Tang,J. *et al.* (2018) Drug target commons: a community effort to build a consensus knowledge base for drug-target interactions. *Cell Chem. Biol.*, **25**, 224–229.

Wang,Y. and Elhamifar,E. (2018) High rank matrix completion with side information. In: *Thirty-Second AAAI Conference on Artificial Intelligence*.

Warmuth,M.K. *et al.* (2003) Active learning with support vector machines in the drug discovery process. *J. Chem. Inf. Comput. Sci.*, **43**, 667–673.