Root Cause Analysis of Failure in Molecular Biology Workflows

Shuowei Li

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2022

Reading Committee:
Eric Klavins, Chair
Rania Hussein
Gerog Seelig

Program Authorized to Offer Degree:
Electrical and Computer Engineering

University of Washington

Abstract

Root Cause Analysis of Failure in Molecular Biology Workflows

Shuowei Li

Chair of the Supervisory Committee:
Eric Klavins
Department of Electrical and Computer Engineering

Root Cause Analysis of Failure in Molecular Biology Workflows

This dissertation describes a framework for accurate and efficient root cause analysis for repeatable biological workflows. This framework has three significant contributions. Firstly, we introduced Open operational protocol semantics (OOPS), a framework that enables the production of physics-based synthetic datasets for biological workflows. OOPS allows users to model the possible failure modes of a flow. We demonstrate how the OOPS framework generates diverse and physics-based synthetic datasets, including cell-free protein synthesis and polymerase chain reaction. Secondly, given the experimental outcomes and records of the items used for each workflow, our framework generates a probabilistic model to represent the variabilities of the workflows. The details of the experimental records, such as technicians and reagents involved in each operation of trials, are encoded to a low-dimensional embedding using a neural network. With the embeddings as the input, a logistic regression-based model is trained to predict whether a particular trial would succeed, considering the involvement of reagents and technicians. With this formulation, we can perform root cause analysis to identify the reason for the failure of trials in a quantitative manner. Furthermore, due to the small-scale biological data set, this hybrid approach allows users to train a compact neural network for feature extraction that facilities prediction based on logistic regression.

We anticipate that our result will identify the source of variability and accelerate research progress. We found the source of variability with $86.75\%, 98.9\%, 97.71\%, 99.7\%$, and 88.31% accuracy in synthetic cell-free protein synthesis, synthetic polymerase chain reaction, real polymerase chain reaction, real Gibson assembly, and real yeast strain construction, respectively. Lastly, we use statistical methods to rank reagents and technicians by their accuracy. The relative quality or accuracy is conditioned on the specific step of a workflow. We use synthetic and real datasets to demonstrate that the overall success rate has improved by replacing degraded reagents or retraining technicians with low accuracy.

Effective Using Remote Lab in Promoting Simulation and Verification Tools

New modalities for conducting hands-on labs are needed with the transition to remote instruction during the pandemic. In particular, courses that entail major hardware components face challenges in making the hardware available for students reliably and sustainably. Furthermore, the industry partners expect students to be well-trained in simulation and verification tools. This dissertation presents our experience in using a virtual breadboard feature interfaced with real Field Programmable Gate Arrays (FPGA) boards located on the University of Washington's campus, where students can access the FPGA hardware remotely to complete their lab assignments. We evaluated this approach through anonymous surveys of students and industry partners. The findings demonstrate that we effectively transformed a lab assignment, typically carried out in person, into an online modality. Using remotely accessible hardware, a practice that showed promise in promoting students' skills in using verification tools, is a desirable skill in the industry.

TABLE OF CONTENTS

LIST OF FIGURES

GLOSSARY

ACCURACY: has two definitions in this framework. When used as model performance metrics, accuracy is used to evaluate the model's prediction. Based on the given test data and labels, accuracy measures what percent of the predictions are correct. It has a form of $Accuracy = \frac{True}{True + False} = \frac{TP + TN}{TP + TN + FP + FN}$. Furthermore, the accuracy of technicians reflects the probability of successfully finishing each operation of the workflow.

ADC: an analog-to-digital converter (ADC) in electronics is a device that transforms an analog signal into a digital signal, such as sound captured by a microphone or light entering a digital camera

AUC: Area Under the ROC Curve is a metric used to assess the model's effectiveness. It can be found by examining the ROC curve's area under the curve. AUC is a performance metric that considers all potential categorization criteria. AUC should ideally be near 1.0, which denotes a good indicator of separability

AQUARIUM: is an interactive web-based software application developed at Klavins Lab. In Aquarium, users can manage the inventory, design/submit the workflow, execute the protocol, collect/retrieve the resulting data logs, estimate the cost of running the whole workflow

BERNOULLI DISTRIBUTION: is a discrete probability distribution with two possible binary outcomes. It accepts the values of 1 (success) and 0 (failure) with probabilities of p and $q = 1 - p$, respectively

BOOSTING: is a class of machine learning algorithms that turns weak learners into strong ones and is an ensemble meta-algorithm in machine learning for minimizing bias and variance in supervised learning

CFPS: Cell-free protein synthesis (CFPS), also known as in vitro protein synthesis (IVPS), is the production of protein in a cell-free system, that is, without the use of living cells. The environment for in vitro protein synthesis is not constrained by a cell wall or the homeostasis conditions required to maintain cell viability

CONTROLLED EXPERIMENT: , a scientific test that often runs concurrently with the initial trial's original schedule. The controlled experiment limits the surroundings of the sample being investigated while allowing the researchers to test one independent variable at a time

CROSS ENTROPY: , a measurement of the differences between two probability distributions for a particular set of occurrences. It is also employed as a statistic to assess the effectiveness of the model

DAC: A digital-to-analog converter (DAC) in electronics is a device that transforms digital signals into analog signals. The reverse function is performed by an analog-to-digital converter (ADC)

DATAASSOCIATION: a key/value pair associated with inventory, plans, or operations

DE1-SOC: a powerful hardware design platform centered on the Altera System-on-Chip (SoC) FPGA

EMBEDDING: , also called representation learning, is commonly used in recommendation systems and natural language processing. Embeddings are algorithms that convert high-dimensional raw data into low-dimensional vectors. An embedding ideally captures some of the semantics of the input by clustering inputs with comparable semantic properties together in the embedding space

FEATURE(S): individual measurable properties or characteristics of a phenomenon being observed

FP: False Positive (FP), an outcome in which the model predicts the positive (success) class incorrectly

FN: False Negative (FN) results when the model predicts the negative (fail) class inadvertently

F1 SCORE: is a statistic used to assess the model's prediction. It is a tallied average of Precision and Recall. It is helpful when dealing with unequal class distribution. Its formula is $F1 = 2 \times \frac{Recall \times Precision}{Recall + Precision}$

FPGA: Field Programmable Gate Arrays (FPGAs) are semiconductor devices that are built around a matrix of configurable logic blocks (CLBs) connected by programmable

interconnects. After manufacturing, FPGAs can be reprogrammed to meet specific application or functionality requirements

GIBSON ASSEMBLY: a biological method to assemble DNA seamlessly and in the correct order

GPIO: A general-purpose input/output (GPIO) is a software-programmable uncommitted digital signal pin on an electronic circuit board or integrated circuit that can be used as an input, output, or both

GRADIENT-BOOSTED TREES: A machine learning method called gradient boosting is used, among other things, for classification and regression tasks. It provides a prediction model in the form of an ensemble of decision trees-like weak prediction models. The gradient-boosted tree is the name of the resulting algorithm when a decision tree acts as the weak learner

HEX DISPLAY: a device that can receive binary inputs and display the hexadecimal number corresponding to those inputs

HUMAN-IN-THE-LOOP AUTOMATION SYSTEM: allows users to define experimental workflows algorithmically, attach upstream design tools, and send data to downstream analysis software without external monitoring intervention

IDENTITY VECTOR: a unique vector assigns to each technician or reagent. It represents who or what they are in the system

JOB: the model representing actions taken during the execution of an Operation

LABEL: 1 for a successful trial, 0 for a failed trial

LABSLAND: an online remote laboratories

LED: a semiconductor light source produces light when current passes through it

LOGISTIC REGRESSION: a statistic model that uses to model the probability of a binary outcome

MIN/MAX ENCODING: the data in each column is scaled to a range of $[0, 1]$ based on

$$x_{scaled} = \frac{x - min(x)}{max(x) - min(x)} \tag{1}$$

where x is the original variable, and x_{scaled} is the normalized value.

MLA: The Maximum Likelihood Estimation (MLA) method is used to estimate the parameters of a probability distribution. It maximizes a likelihood function so that the observed data is most likely under the assumed statistical model

NOMINAL VARIABLE(S): a categorical variable classification. It is used to identify a set of values. Nominal variables do not provide orders or quantitative values

NORMALIZATION: a scaling technique. A [0,1] range is created by shifting and rescaling the values

ODDS: the probability of an event occurring divided by the probability of it not occurring

ODDS RATIO: a ratio used to calculate the change in the odds of a particular outcome

OPERATION: is a discrete, self-contained procedure on a set of reagents performed by a technician

ONE-HOT ENCODING: a modeling technique for converting nominal variables to numbers. Each category is mapped to a vector with 0 and 1 values indicating whether or not the feature is missing

PRECISION: is a metric for assessing model performance. It displays the percentage of positive predictions that were correct. It has a form of $Precision = \frac{TP}{TP + FP}$

PCR: Polymerase Chain Reaction (PCR) is a common laboratory technique used to make many copies of a particular region of DNA in a test tube

PLAN(S): a set of Operations connected in a graph that roughly represents a workflow

PROBABILITY: the quantitative measurement of the likelihood that a specific event will occur

PYHTON: Guido van Rossum's widely used high-level programming language for general-purpose programming, first released in 1991

PYTORCH: a free and open-source machine learning library created by Facebook's AI Research lab

QUALITY: of a reagent to quantify its ability to react as expected when used as described in a workflow

RAW DATA: The dataset has not yet been prepared for use

RASPBERRY PI: a microcontroller board

ROOT CAUSE: is a factor that contributed to a nonconformance and ought to be permanently removed via process improvement. The core issue, or root cause, is what started the complete chain of events that led to the problem in the first place. It is also the most fundamental cause (s)

RCA: Root Cause Analysis (RCA) refers to various methods, instruments, and procedures used to identify the root causes of issues. Some root cause analysis (RCA) methodologies are more focused than others on locating the actual reasons for an issue. Other RCA methodologies are generic problem-solving approaches

REAGENT: chemicals used in a workflow, such as media, enzymes, and antibodies

RECALL: is a model performance metric known as sensitivity. It displays the percentage of genuine positives that the classifier successfully identified. It has a form of $Recall = \frac{TP}{TP + FN}$

REGULARIZATION: a method for lowering error by properly fitting a function to the provided training data while minimizing the chance of overfitting. The coefficient estimations are reduced until they are zero

REPEATABILITY: in biological research, it measures the variation between results of the successive workflows performed under identical conditions

ROC: Receiver Operating Characteristics (ROC) curve is a metric used to assess the performance of a classifier's model. It summarizes the classifier's ability to distinguish between positive and negative examples. The ROC curve is constructed by plotting recall versus one minus specificity at various threshold settings.

SCIKIT-LEARN: an open-source machine learning library. It features modeling data

SIGMOID FUNCTION: , also known as the sigmoidal curve or the logistic function. It has a form of $y = \frac{1}{1 + e^{-x}}$

SIMULATION TOOL: the foundation is the mathematically simulating genuine phenomenon. In essence, it is a program that allows the user to view and imitate an activity without actually carrying it out. Equipment is frequently designed using simulation software to ensure that the finished product adheres to the design specifications as closely as feasible without requiring expensive process adjustments

STANDARDIZATION: a scaling technique in which the values are centered on the mean and have a unit standard deviation. The attribute's mean becomes zero after standardization, and the resulting distribution has a unit standard deviation. It has a form of $X' = \frac{X - \mu}{\sigma}$

SYNTHETIC DATA: generated by a computer simulation. It contains all the characteristics of the real dataset

SYSTEM VERILOG: a hardware description and hardware verification language is used to model, create, simulate, test, and build electronic systems

TECHNICIAN: executes operations in the workflow. One operation only involves one technician.

TEST SET/DATA: a subset of data to test the trained model

TRIAL: each repetition of the same biological workflow

TRAINING SET/DATA: a subset of data to train a model

TRIDENT: the Python API in Aquarium. Users can manage inventory, plan experiments, and query data for Aquarium

TP: True Positive, an outcome where the model correctly predicts the positive(success) class

TN: True Negative, an outcome where the model correctly predicts the negative(fail) class

VISIR: Virtual Instrument Systems in Reality (VISIR) is a remote laboratory developed by the Blekinge Institute of technology. It allows users to design, wire, and measure electronic circuits

VERIFICATION TOOL: a collection of technologies that use static analysis to prove or disprove the correctness of hardware or software behavior with a specific formal specification or property

WORKFLOW: a set of standard laboratory operations with specific ordering constrained, meaning operation A may be Deon before operation B, but operations B and C may be done independently

XGBOOST: an open-source library targeted gradient boosting

YEAST STRAIN CONSTRUCTION: taking genetic materials to transform it into a given strain of competent yeast cells

ACKNOWLEDGMENTS

I take this opportunity to thank all the people who helped me with this document. Firstly, I would like to express my gratitude and appreciation to my PI, Prof. Eric Klavins, for the dedicated support and guidance during the research. His excellent support and thoughtful advice bolstered the quality of this research and my spirit and faith in the research process. Secondly, I would like to thank my thesis committee: Prof. Rania Hussein, for her valuable time and for sharing her insightful comments and suggestions throughout this work. Finally, Prof. Georg Seelig, for his thoughtful comments and recommendations on this work.

My fellow labmates at Klavins Lab have supported me and had to put up with my stresses and moans for the past few years of study! I would like to thank Labsland and Intel Corporation for their collaboration and support of the remote FGPA labs.

Finally, I cannot forget to thank my family and friends, who always support and encourage me.

This research was funded by the Defense Advanced Research Projects Agency (DARPA) Synergistic Discovery and Design (SD2) program.

DEDICATION

To my dear family

Chapter 1

ROOT CAUSE ANALYSIS OF FAILURE IN MOLECULAR BIOLOGY WORKFLOWS

1.1 Introduction

Repeatability measures the variation between the results of successive experiments performed under identical conditions. In recent years, the biological research community has faced a methodological crisis in which scientists found the results of various publications challenging to replicate in subsequent studies, either by other researchers or by the authors themselves [4]. Unrepeatable experiments can result in time-consuming and costly repetition of experiments, and in some extreme cases, invalid results and conclusions. The low repeatability of experiments has long been recognized as an acute issue in biological research. According to a survey published in *Nature* [4], more than 70% of researchers cannot replicate another scientist's experiments, and more than half of them have difficulty reproducing their own experiments.

Biological experiments can be highly complex and sophisticated, involving dozens of operations and taking weeks to complete. Consequently, the cost of performing such complex and multi-operation experiments can be high [5]. For example, a single protein stability assay can cost around ten thousand dollars and takes three days to complete. If an experiment fails, it is highly desirable to investigate the cause of failure to improve the probability of future successful experiments. The procedure to identify the underlying cause of a failed experiment is referred to as *root cause analysis* [6, 7, 8]. Practitioners have been relying on a combination of experience and exhaustive search to perform root cause analysis, making the process highly time-consuming and cost-prohibitive [9].

Biofoundries introduce an empirical approach to perform root cause analysis to address the repeatability issue. When a failed trial is identified, the system will introduce a set of

new control experiments by replacing chemicals/technicians [10, 11, 12]. The results can be used to quantify the variabilities associated with failure trials. Furthermore, researchers with extensive domain knowledge of the experiments may be able to quantify relative contributions to variability associated with the failed experiments by analyzing the personnel and instruments involved in the experiments [13, 1]. However, as the quantity and complexity of the data produced by automated experiments increases, manually analyzing such data becomes impractical.

This dissertation is based on extensive previous work by the UW BIOFAB [14], a biofoundry located at the University of Washington. The UW BIOFAB developed Aquarium [15], a software system for data management in biological laboratories. In Aquarium, users can manage their laboratory's inventory, design, submit and execute workflows, and collect and retrieve the resulting data logs. Some experiments in this dissertation are based on *Aquarium*, and it is justified first to define several terminologies repeatedly used throughout this dissertation. Operation is a discrete, self-contained procedure on a set of reagents performed by a technician. For example, run gel is an operation in the PCR workflow. Workflow is a set of standard laboratory operations with specific ordering constrained, meaning operation A may be done before operation B, but operation B and C may be done independently, for example. Each execution of a workflow is called a trial. Technician executes operations in the workflow. One operation only involves one technician. However, a workflow may distribute to multiple technicians. We use reagent to refer to chemicals used in a workflow, such as media, enzymes, and antibodies. All inventory and generated data are captured and cataloged during the workflow execution. Experimental logs include technician identity, timestamps for each operation in a trial, trial error records, and inventory handled.

While *Aquarium* is mainly focused on data management of real-world biological experiments, we are also interested in synthetic data for the training and validation of our proposed methods. Open Operational Protocol Semantics (OOPS), also developed by the UW BIOFAB, is a framework for synthesizing physics-based synthetic datasets for biological workflows.

In this work, we focus on two factors that may lead to failed biological experiments: the quality of reagents used in the trials and the accuracy of technicians. We use the term quality of a reagent to quantify its ability to react as expected when used as described in a workflow. Accuracy of technicians reflects the probability of successfully finishing each operation of the workflow. To better harness the insights captured by the experimental records, we take a data-driven approach to address the repeatability issue in workflows. This work is inspired by the rich literature in root cause analysis [6, 7, 8].

We implemented a logistic regression-based method to perform root cause analysis. Given the experimental outcomes and records (technicians and reagents involved in each trial), the objective is to *infer* quality of reagents and accuracy of technicians. A probabilistic model captures the relationship between the observed experimental outcome and the underlying variabilities due to the quality of reagents and technician performance. We use a neural network to encode informative features of reagents and technicians as a low-dimensional *feature embedding*. Examples of features that we used include the age of the reagent and the number of trials a technician has performed. In addition, following well-studied techniques in recommender systems and data mining [16, 17, 18, 19], we also assign a unique low-dimension *identity embedding* to each technician and reagent to account for the information that was not reflected by the feature embedding. With embeddings as input, a logistic regression-based model is trained to predict whether a particular trial would be successful, considering reagents and technicians involved in the trial. With such a probabilistic model, we could identify the most likely root causes that led to the failure. To further improve the overall success rate of all trials, we retrain technicians or replace reagents based on the ranking of technicians and reagents by their accuracy and quality.

Complex biological workflows involve multiple operations. For example, in a polymerase chain reaction (PCR) workflow, operations are make PCR fragments, run gel, extract gel slice, and purify gel slice. Through extensive discussion with practitioners, we observe that in a multi-operation workflow, if one operation failed, the entire trial would inevitably fail. Traditional logistic regression does not directly reflect such important observation [20]. In

our probabilistic model, we explicitly model the outcome of each operation and mandate that all operations have to be successful for the trial to be considered successful. The encoding process and logistic regression variables are jointly optimized through conjugate gradient descent.

This dissertation is organized as follows. We describe related works in Section 1.2. Section 1.3 presents the framework's methodology. In Section 1.4, we introduce the syntax and semantics of a synthetic data generator called OOPS. Case studies and the corresponding results are presented in Section 1.5, demonstrating that our method can infer the root cause of failed trials in both synthetic and real-world datasets. Finally, we talk about the future works in Section 1.6, and conclude the research findings in Chapter 3.

1.2 Literature Review

1.2.1 Data and Experiment Management

In biological research, the online protocol editors [21] allowed the user to detail, share, and discuss the experimental protocols. Autoprotocol [22, 23] allowed users to build and test protocol execution. The protocol management system is also in high demand. Moreover, researchers nowadays use sophisticated equipment and outsource services. Thus it is essential to find an optimal way to incorporate the experimental inputs and outputs into the workflows.

Combing protocol editors [21] with protocol management systems can provide a platform for different laboratories/research teams to organize the research activities. Aquarium [14], is an example of such a system. In Aquarium, users can manage the inventory, design and submit experiments, execute the protocol, and collect and retrieve the resulting data logs. To be more specific, *designer* can specify a *protocol* either using web-based graphical user interface (GUI) or Python API Trident. *Protocol* will then be sent to the laboratory as a *job* to be performed by technicians. Aquarium can also estimate the cost of running the whole trial. During *protocol* execution, all inventory and generated data will be captured and cataloged. When a *job* completes, *designer* can download the generated results to the downstream data analysis tools. The use of materials can also be tracked at the same time.

As biological workflows get more complicated today, researchers tend to use equipments manufactured by different companies. *Synthace Antha* is a software designed to facilitate researchers' management of the equipments involved in the experiments.

Other biofoundaries, such as Ginkgo Bioworks and Zymergen [22], aim to automate the experimental design process by integrating biology, software, and hardware. This approach allows reserachers to optimize workflow design, speed up research and discovery, and allow trials to progress rapidly.

Great efforts have been made to improve workflows repeatability by standardizing reaction conditions and experiment protocols [24, 25, 1]. Pardee [24] evaluated the batch-to-batch variations for the cell-free transcription-translation system.

Batch-to-batch variation was also studied in Silverman *et al.*'s work [25]. The main goal of this research is to standardize the methodology of cell-free extract preparation. Silverman *et al.*[25] characterized the variations of the preparation process. The authors conducted an in-depth investigation by preparing three identical extracts on three different days. Researchers believed the RNA degradation rates in batches do not contribute much to the experimental performance. Instead, the result demonstrated that the variation is due to both extract-extract variation and trial-trial variations.

Cole *et al.* [1] ran identical standardized workflow called cell-free protein synthesis across three laboratories to identify and quantify variability. A linear mixed model was fit into the data to estimate how the inputs contribute to the workflows' variability. A mixed linear model notably aimed to disentangle the within-cluster effects from between-cluster effects. This model assessed a standard protocol's variability by introducing the inputs, such as sites, technicians, reagents, extract, and shared materials.

As shown in Figure 1.1, Cole *et al.* [1] presented a table listing all possible factors contributing to the overall variability of CFPS. After running various controlled experiments, Cole *et al.* found four factors dominate the variability in CFPS workflow: (1) technician pipetting technique, (2) the time to set up the experiment, (3) pipetting differences in loading standard curves among different sites, and (4) reagent. We will model our CFPS workflow using the same factors that dominate the variability in CFPS workflow.

category	description	contributing factors
Operator	the person preparing the reaction	• pipetting technique (number/rate of mixes, pipetting accuracy, *etc.*)
		• time to set up an experiment
Site	the laboratory and instrument where the reaction is run	• ~~differences in instruments that are not controlled for by standard curves (e.g., bias across~~ plates)
		• ~~plasticware and pipettes~~
		• pipetting differences in loading standard curves
		• ~~ambient conditions (e.g., humidity)~~
Reagent	all components of the reaction except the extract and shared materials	• lot number, catalog number, and manufacturer of individual components
		• technique of individual preparing each batch of reagents (pipetting accuracy, pH measurement accuracy, *etc.*)
Extract	material derived from lysed *E. coli* cells	• variability between culture batches
		• differences between lysis instruments
		• batch-to-batch variability within lysis instruments
		• differences between operator of lysis instruments
		• execution of postprocessing steps
Shared Materials	materials shared from a common stock	• differences between aliquots (shared materials are DNA template, RNase inhibitor, malachite green, and T7 RNA polymerase)
Day	unknown sources of day-to-day variability	• factors from categories above that change from day-to-day, such as ambient conditions

Figure 1.1: All possible factors contribute to the overall variability of CFPS. Cole *et al.* [1] found four factors dominate the variability in CFPS workflow: (1) technician pipetting technique, (2) the time to set up the experiment, (3) pipetting differences in loading standard curves among different sites, and (4) reagent.

1.2.2 Synthetic Data Generation

Synthetic data is widely used in machine learning and data analysis. Sun *et al.* [26] used neural network to identify an optimal embedding for synthesizing high-dimensional data. Alberti *et al.* [27] presented a novel method of producing synthetic question answering corpora, by integrating question creation and answer extraction models and filtering the outcomes to ensure round-trip consistency. Bowen *et al.* [28] presented an extensive evaluation of several differentially private synthetic data algorithms. Tsoukalas *et al.* [29], implemented an R-package called anySim, for synthesis of non-Gaussian correlated random variables, stochastic processes, and random fields.

1.2.3 Root Cause Analysis

Root Cause Analysis (RCA) is a retrospective method of identifying the root cause of faults that generated a given set of symptoms [30]. A statistical model is built to represent the relationship between observed symptoms and faults to perform this inference process in complex systems.

Using machine learning or statistical learning methods, we can construct a surrogate model to link observed symptoms to a set of possible faults. Kumari *et al.* [31] used a Hierarchical Bayesian Model to study the rare events in the chemical process industry. The hierarchical Bayesian model is a probabilistic graphical model that captures the corresponding relationship between a set of random variables with a directed acyclic graph. Kumari *et al.* introduced an informative prior to compensating for limited data and expert judgments. This informative prior allows the system to cope with source-to-source variability. Ferreira and Vasilyev [32] proposed a decision tree-based method. Event logs are transformed into logical representation and train a decision tree-based classifier. The sub-sets in the generated model can compare the average duration with other sub-sets, and provide a root cause for this difference. Dean *et al.* [33] employed an unsupervised learning method called Self Organizing Map to predict anomalies in virtual cloud systems.

1.2.4 Logistic Regression

The objective of logistic regression is to characterize the relationship between a binary outcome variable y and various predictor variables X on a log-odds scale [34]. It is commonly written in the form:

$$P(y|x,\theta) = Bernoulli(y|sigmoid(\theta^T x + c)) \tag{1.1}$$

where x contains the feature values. y denotes the outcome variable, which is a binary variable having either a success (1) or failure (0) as the outcome. θ and c are variables to estimate. θ contains the coefficients for x, and c are intercepts.

Bernoulli stands for a Bernoulli distribution, which is commonly used when the response is in a binary state. $\theta^T x$ computes the linear combination of the inputs. Then we pass it through a sigmoid function to ensure the function $sigmod(\theta^T x) \in [0, 1]$. The sigmoid function is also called the logistic function or logit function. The definition of the sigmoid function is:

$$sigmoid(\theta^T x + c) = \frac{1}{1 + e^{-\theta^T x + c}} \tag{1.2}$$

The log-odds for the binary outcome variable y is a linear combination of a group of predictor variables. The predictor variables x can be either continuous or categorical. We can use logistic regression to determine which predictor variables x are statistically significant, along with a measure of the magnitude of the potential influence with respect to the outcome of interest.

The differences between *probability* and *odds* is worth emphasizing. Probability is the quantitative measurement of the chance that a particular event will occur. Odds are defined by the probability that an event will occur, divided by the probability that it will not occur. Change in odds of an outcome, for example, an increase in odds of a failed trial associated with a certain technician - is measured as a ratio called *odds ratio*. For example, if a trial associated with technician A has an odds of failure of 2.0, and a trial not associated with technician A has an odds of failure of 0.5, then the odds ratio associated with technician A would be $2 : 0.5$ or 4. It is the same as an increase in the probability of failure from $\frac{1}{3}$ to $\frac{2}{3}$.

The odds ratio for the outcome associated with an event (for example, the reagent used in the experiment) is determined by the coefficient calculated for each predictor variable. These odds ratios measure the magnitude of each predictor variable associated with a particular outcome. In other words, it defines the uncertainty for each predictor variable in terms of the magnitude of the influence.

Logistic regression has been applied to various fields, including medical fields [20, 34], social sciences [35] and marketing [36]. Medical researchers used two staged logistic regression to assess the illness severity of a patient in an Intensive Care Unit (ICU) [34]. Kavade [37] developed a predictive logistic regression model to determine the severity of a future incident utilizing the Human Factors Analysis and Classification System. The proposed Human Factors Analysis and Classification System is a well-established framework for performing the root cause analysis for an unsafe event.

Thiru *et al.* [38] detects coronary heart disease using a logistic regression model. The authors reduce the numbers of predictor variables using the Receiver Operating Characteristic (ROC) curve. The prediction result of the logistic regression model is a continuous number. However, the ideal result should be in a binary manner (whether a given patient has coronary heart disease). To map the classification result to a regression result, the authors need to find an optimal cut-off line. ROC curve would be the optimal tool to fulfill this requirement. This ROC curve can be generated by plotting the cumulative distribution function of the false-positive rate in the x-axis versus the cumulative distribution function of the probability of detecting coronary heart disease in the y-axis.

Logistic regression sometimes yields unstable estimation in high dimensional settings, where the number of predictor variables $p \gg$ is the number of samples n. To resolve this issue, Park *et al.* [20] introduced an adaptive penalized logistic regression based on L_1-regularization to rule out these insignificant predictor variables. This method encouraged sparse solutions and used a limited number of variables for predicting.

1.2.5 Gradient-boosted Trees

Classification and regression methods based on ensembles of decision trees are widely adopted in practice [39]. Popular gradient-boosted trees libraries include XGBoost [40], LightGBM [41], and CatBoost [42]. Such methods can naturally handle combinations of continuous inputs and categorical inputs.

Boosting is an ensemble technique that sequentially groups weak learners to improve learning performance. Once a weak learner is learned, the misclassified points will be assigned a higher weight, and the correctly predicted points will be weighted lower. In the next iteration, the boosting algorithm will update the model according to the weights of the points assigned in the subsequent iteration. The boosting algorithm will build a more robust model by capitalizing on the misclassification error. The resulting algorithm is called gradient-boosted trees when using a decision tree as the weak learner.

On the downside, gradient-boosted trees is known to be prone to overfitting if too many decision trees are added [39]. Thus gradient boosted trees require meticulous tuning of the hyperparameters. The gradient-boosted trees model is also hard to interpret.

1.2.6 Neural Network based Automatic Feature Engineering

Embedding, also called representation learning, is a commonly used technique in recommendation systems or natural language processing [16]. For example, in a recommender system for an online movie streaming website, we recommend movies to users based on the features of the user (such as zip code and browser version), and the past clicking history of the user. In our context, we would like to recommend which reagent is due for replacement or which technician is due for replacement.

One-hot encoding is an option to represent nominal or categorical data. For example, as shown in Figure 1.2, the failed trials are presented in the table. Each row of the table is a failed trial with involved reagents. In this table, the last row is represented as a sparse tensor [0, 2, 999], meaning reagents 0, 2, and 999 are used in this trial. Using one-hot encoding,

the size of the representation is equal to the number of possible categories (total number of technicians or reagents) to represent. As the number of categories grows, one-hot encoding faces scalability issues.

Embeddings represent large sparse vectors in a lower-dimensional space that preserves semantic relationships. Recommender systems that utilize embedding are [18, 19, 17].

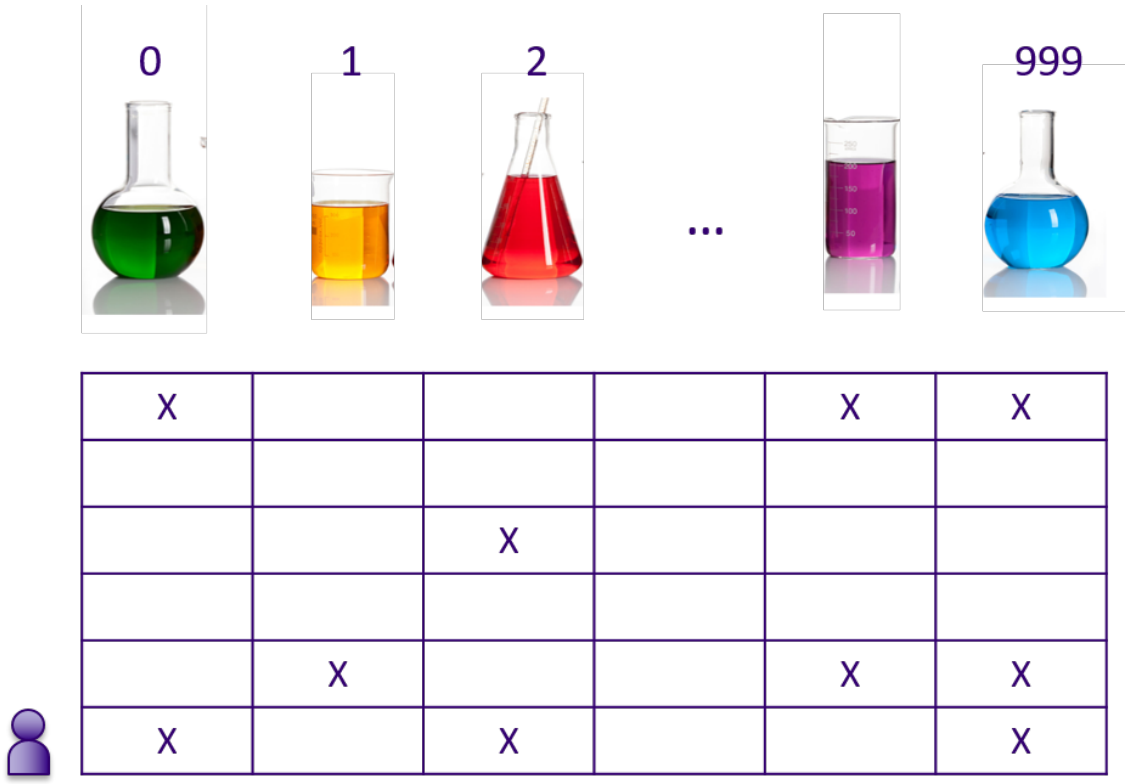


Figure 1.2: A table shows all failed trials for a biological experiment. Each row of the table is a failed trial with the involved reagents. The last row in the table corresponds to a sparse tensor [0, 2, 999], meaning reagents 0, 2, and 999 are used in this trial.

A large amount of data is required to train a deep neural network effectively. Such high data requirement is not an issue in specific applications where data is readily available (for example, E-Commerce), but it could become a significant hurdle in our task where the data scale is much smaller. In this work, we take a hybrid approach where we use a compact

neural network to encode the available information of the technicians and reagents as an embedding [43], which serves as input to a standard logistic regression. With this setup, we could afford to train a compact neural network for feature extraction that facilities prediction based on logistic regression.

1.3 Methodology

Our objective is to identify the most likely root causes (either due to a technician with low accuracy or a degraded reagent) of a failed trial. The flowchart of the proposed system and contributions are summarized in Figure 1.3.

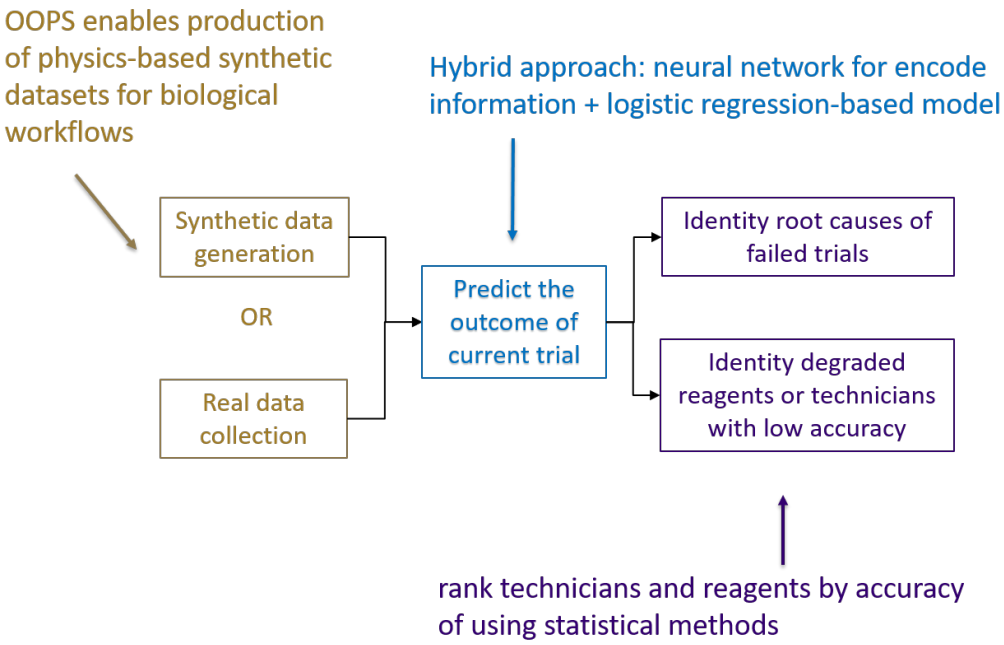


Figure 1.3: Problem statements and contributions of the proposed system.

The input to our method are:

- The known features of technicians and reagents. For technicians, features are statistics of technicians performing certain workflow operations, including the number of technicians who have attempted a specific operation, the number of trials that technicians completed, the number of trials that technicians aborted, and the number of successful and failed trials that technician has performed. For reagents, the features are the type of reagent and its age.

- The involvements of technicians and reagents in each operation of the trial. Using an experiment management system allows for the collection of such data. Each technician and reagent in the experiment management system is associated with a unique *ID*. In each operation, the ID of the technician who performed and the ID of the reagents used are recorded.

Formally, let a training dataset $\boldsymbol{S}$ with N trials that involved N_T technicians and N_R reagents. Each trial $\boldsymbol{s} = (\boldsymbol{u}_1^t, \boldsymbol{u}_1^{c1}, \cdots, \boldsymbol{u}_1^{cQ_1}, \cdots, \boldsymbol{u}_Z^t, \boldsymbol{u}_z^{c1}, \cdots, \boldsymbol{u}_Z^{cQ_Z}, y) \in \boldsymbol{S}$ is a sequence of $(Q_1 + \cdots + Q_Z + Z)$ technician and reagent IDs. $\boldsymbol{u}_j^t$ and $\boldsymbol{u}_j^{ca}$ are technician ID and reagent IDs in operation j. Let Z be the number of operations in the trial, and $j \in \{1, 2, \cdots, Z\}$ be the index of Z operations. Define $a \in \{1, \cdots, Q_j\}$ be the index of Q_j reagents involved in each operation j.

1.3.1 Synthetic Data Generation

The effectiveness of data-driven machine learning algorithms depends on a large amount of training data with precisely labeled ground truth. However, the availability of labeled training data could be of significant concern. Biological experiments are notoriously costly and time-consuming, thus limiting the number of experimental logs. Furthermore, the underlying root cause of the failed experiment is generally unknown. In such cases, synthetic data closely mimics real-world data dynamics, and distribution is highly desirable to facilitate algorithm design and validation.

To this end, we designed a physics-based synthetic data generator, named Open Operational Protocol Semantics (*OOPS*), which aims to generate synthetic datasets with the same mathematical and statistical properties as the real-world biological workflows datasets.

OOPS allows us to define sources of variabilities in reactions, such as the accuracy of technicians and the qualities of reagents. For example, reagents modeled in OOPS contain various attributes, such as concentration, volume, pH value, and temperature. Changing these attributes of reagents affects the final products of reactions. The statistical distribution

of the attributes (both for technicians and reagents) can be defined by users.

OOPS also allows flexibility to define how the attributes of technicians and reagents affect the outcome of biological workflows. For example, the attributes could impact the parameter of an ordinary differential equation [44, 45], which captures the dynamics of biological workflow and, in turn, decides the output of workflow.

With this formulation, OOPS could serve as a practical synthetic data framework for root cause analysis in biological workflows. The ground-truth root causes of the failed trials are known to users. We will show in a later section that the availability of ground-truth labels greatly helps the design and validation of the algorithms. Additionally, we will show that the insights we gained with synthetic data extrapolate to real-world data.

Admittedly, gaps between real-world and synthetic data exist. A biological workflow is a very complex procedure that an ordinary differential equation cannot fully describe with a few dozen parameters. The seriousness of the gap is conditioned on how accurately we could model the relationship between attributes of technicians and reagents, and the output of workflows. In this work, we employed models in state-of-the-art works [44] to minimize the gap between real-world and synthetic data.

We will describe the detailed procedure of synthetic data generation in Section 1.4.

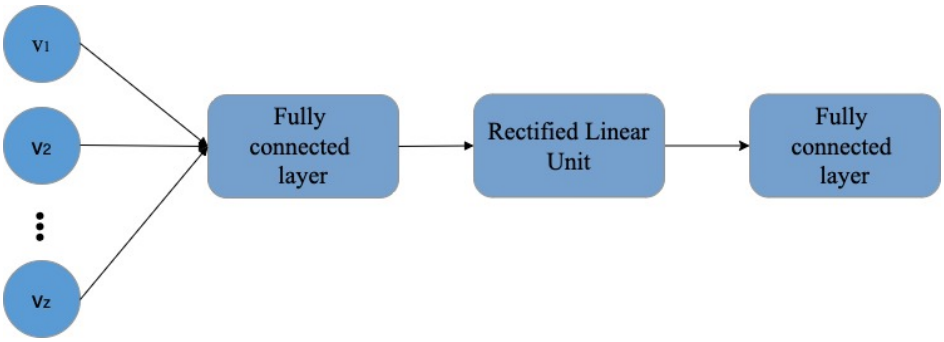
1.3.2 Feature Engineering

The features of technicians and reagents contain precious prior information for our prediction task. However, such prior information often requires some feature engineering to be effectively used in a probabilistic model. Therefore, instead of using manual feature engineering to encode such prior information, we employ a neural network to encode the features. The neural network automatically learns a non-linear transformation of the prior information based on the training data. Such automatic feature engineering using the neural network has been proven to be highly successful in various applications. We refer to the output of the neural network as the *feature embedding* of a technician or a reagent. The neural network projects the features into a high-dimensional manifold to facilitate the downstream

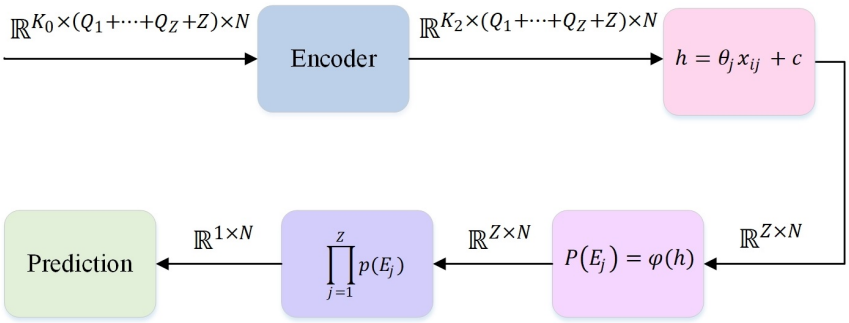
prediction tasks. Leveraging automatic feature engineering enabled by neural networks is a critical technical contribution of this work. As we shall show in Section 1.5.2, our neural network-based feature engineering outperforms manual feature engineering.

Figure 1.4a is an overview of the neural network architecture we used to generate feature embeddings. This neural network consists of the following layers: [Input - Fully Connected Layer - Rectified Linear Unit - Fully Connected Layer]. The input layer has a size of $\mathbb{R}^{K_0 \times (Q_1 + \cdots + Q_Z + Z) \times N}$. The output of the neural network is of shape $\mathbb{R}^{K_1 \times (Q_1 + \cdots + Q_Z + Z) \times N}$, where K_0 and K_1 are hyperparameters to control the capacity of the network, N is the dimension of the feature embedding.

Feature embedding is not sufficient to capture all relevant information. For example, a technician may have a long tenure in another lab, so their statistics of completing a certain operation cannot be collected and properly reflected in their feature embedding. Another example would be two bottles of reagents of the same type may have an exact age, but one bottle has been contaminated and would lead to failed trials. To address such ambiguity issues, we introduce a randomly generated vector of unit length to each technician and reagent to augment their feature embedding. We call such a vector *identity embedding*. Using a random unit vector to introduce individuality is a commonly used technique in recommender systems, and data mining [18]. Feature embedding and identity embedding are concatenated to generate technician and reagent embedding. Figure 1.4 summarizes how embeddings are generated.



(a) The encoder encodes the informative features and identity features of technicians and reagents as low-dimensional feature embeddings. K_0 and K_1 are hyperparameters to control the capacity of the network. Z is the total number of operations. The total number of trials is defined as N. Q is the number of reagents involved in each operation.



(b) The logistic regression-based model with embedding. The **Encoder** is the neural network architecture shown in Figure 1.4a. K_2 is a hyperparameter to control the capacity of the network.

Figure 1.4: Given the known features of technicians and reagents, and the involvement of technicians and reagents in each operation of the trial, the goal is to predict the probability that the trial would succeed. Figure 1.4a is the neural network architecture that performs the automatic feature engineering. The output of the encoder is an embedding that summarizes all the information that we have regarding a technician or a reagent. In Figure 1.4b, a logistic regression-based model uses the embedding to capture the relationship between the observed outcome and the underlying variabilities due to technicians and reagents.

1.3.3 Probabilistic Model

The embedding of each technician and reagent effectively summarizes all prior information that we have regarding their accuracy. We implement a probabilistic model built upon logistic regression with embedding as input. Our probabilistic model aims to predict the probability that a particular trial would succeed, given which technicians and reagents were involved in each operation.

In a multi-operation workflow, it has been observed by the practitioner that if one operation fails, then the entire trial will fail. Therefore, to facilitate the accurate localization of technicians with low accuracy or degraded reagents, we explicitly model the probabilistic distribution of each operation's outcome (success or failure). This is a critical difference between our method and traditional logistic regression.

For the sake of discussion, let us assume that a trial only involves one operation and describe how we predict the probability of success of a single operation in a trial. Obviously, in this case, if the single operation in the trial is successful, then the trial is successful. The prediction is based on logistic regression. With technician embedding $\boldsymbol{x}^t$ and reagent embedding $\boldsymbol{x}^r$ as the input, the goal is to predict the outcome of the trial, which is a binary random variable $\boldsymbol{y} \in \{0, 1\}$. Again, if a trial only consists of a single operation, the outcome of the single operation is essentially the outcome of the trial. We model $\boldsymbol{y}$ as a Bernoulli random variable

$$\boldsymbol{P}(\boldsymbol{y} \ = 1) = \phi((\boldsymbol{\theta}^t \cdot \boldsymbol{x}^t) \times (\boldsymbol{\theta}^r \cdot \boldsymbol{x}^r) + \boldsymbol{c}) \tag{1.3}$$

where $\boldsymbol{\theta}^t$, $\boldsymbol{\theta}^r$ and $\boldsymbol{c}$ are variables of logistic regression to be estimated, $\phi(x) = \dfrac{1}{1 + e^{-\boldsymbol{x}}}$ is the logistic function. If more than one reagent is involved in an operation, each reagent's distinct $\boldsymbol{\theta}_r$ is introduced. For example, with two reagents involved in an operation, Equation 1.3 becomes $(\boldsymbol{\theta}^t \cdot \boldsymbol{x}^t) \times \prod_{a=1}^{Q_1} (\boldsymbol{\theta}^{r,a} \cdot \boldsymbol{x}^{r,a}) + \boldsymbol{c} = (\boldsymbol{\theta}^t \cdot \boldsymbol{x}^t) \times (\boldsymbol{\theta}^{r,1} \cdot \boldsymbol{x}^{r,1}) \times (\boldsymbol{\theta}^{r,2} \cdot \boldsymbol{x}^{r,2}) + \boldsymbol{c}$, where $Q_1 = 2$. With the probability of success of a single operation defined in Equation 1.3, we now describe modeling the probability of success of a trial with multiple operations. We assume that a trial is successful if and only if all the operations in the trial are successful. Such an

assumption help us better localize the operation in which an error occurred and consequently facilitates identifications of technicians with low accuracy or degraded reagents. To facilitate further discussion, let $\boldsymbol{E}_j$ be the event that the jth operation of a trial is successful, thus $\boldsymbol{P}(\boldsymbol{E}_j) = \phi\big((\boldsymbol{\theta}^t \cdot \boldsymbol{x}^t) \times \prod_{a=1}^{Q_j}(\boldsymbol{\theta}^{r,a} \cdot \boldsymbol{x}^{r,a}) + \boldsymbol{c}_j\big)$.

The probability that a trial is successful is simply the product of the probability of success for all operations in the trial

$$\begin{aligned}\boldsymbol{P}(\boldsymbol{y}=1) &= \boldsymbol{P}(\bigcap_{j=1}^{Z}\boldsymbol{E}_j) \\ &= \prod_{j=1}^{Z}\boldsymbol{P}(\boldsymbol{E}_j) \\ &= \prod_{j=1}^{Z}\phi\Big((\boldsymbol{\theta}^t \cdot \boldsymbol{x}^t) \times \prod_{a=1}^{Q_j}(\boldsymbol{\theta}^{c,a} \cdot \boldsymbol{x}^{c,a}) + \boldsymbol{c}_j\Big)\end{aligned} \tag{1.4}$$

In the training dataset, a multi-operation trial is repeated N times. Let $\boldsymbol{P}(\boldsymbol{y}_i = 1)$ denote ith trial is successful, where subscription $i \in \{1, \cdots, N\}$ is the index of trial in training dataset. We also define $\boldsymbol{x}_{i,j}^t$ and $\boldsymbol{x}_{i,j}^{r,a}$ as technician embeddings and reagent embeddings in jth operation of ith trial, respectively. Thus the probability of success of ith trial is simply

$$\boldsymbol{P}(\boldsymbol{y}_i = 1) = \prod_{j=1}^{Z}\phi\Big((\boldsymbol{\theta}_j^t \cdot \boldsymbol{x}_{i,j}^t) \times \prod_{a=1}^{Q_j}(\boldsymbol{\theta}_j^{r,a} \cdot \boldsymbol{x}_{i,j}^{r,a}) + \boldsymbol{c}_j\Big) \tag{1.5}$$

In the training dataset, we also have the ground-truth outcome of each trial. We perform Maximum Likelihood Estimation to estimate the variables of the prediction model in Equation 1.5. Note that the variables to be estimated include θ_j and c_j, as well as weights of the neural network, denoted W, which generates technician embeddings and reagent embeddings. In other words, the neural network that performs the automatic feature engineering is jointly optimized with logistic regression variables.

Formally, the objective function of the optimization is

$$
\begin{aligned}
&\boldsymbol{L}(\boldsymbol{\Theta}, \boldsymbol{c}, \boldsymbol{W}) \\
&= \log \left[\prod_{i=1}^{N} \left(\prod_{j=1}^{Z} \phi\Big((\boldsymbol{\theta}_j^t \cdot \boldsymbol{x}_{i,j}^t) \times \prod_{a=1}^{Q_j} (\boldsymbol{\theta}_j^{r,a} \cdot \boldsymbol{x}_{i,j}^{r,a}) + \boldsymbol{c}_j \Big) \right)^{\boldsymbol{y}_i'} \left(1 - \prod_{j=1}^{Z} \phi\Big((\boldsymbol{\theta}_j^t \cdot \boldsymbol{x}_{i,j}^t) \times \prod_{a=1}^{Q_j} (\boldsymbol{\theta}_j^{r,a} \cdot \boldsymbol{x}_{i,j}^{r,a}) + \boldsymbol{c}_j \Big) \right)^{1-\boldsymbol{y}_i'} \right] \\
&= \sum_{i=1}^{N} \left(\sum_{j=1}^{Z} \boldsymbol{y}_i' \log(\phi\Big((\boldsymbol{\theta}_j^t \cdot \boldsymbol{x}_{i,j}^t) \times \prod_{a=1}^{Q_j} (\boldsymbol{\theta}_j^{r,a} \cdot \boldsymbol{x}_{i,j}^{r,a}) + \boldsymbol{c}_j \Big) + (1 - \boldsymbol{y}_i') \log(1 - \prod_{j=1}^{Z} \phi\Big((\boldsymbol{\theta}_j^t \cdot \boldsymbol{x}_{i,j}^t) \times \prod_{a=1}^{Q_j} (\boldsymbol{\theta}_j^{r,a} \cdot \boldsymbol{x}_{i,j}^{r,a}) + \boldsymbol{c}_j \Big) \right)
\end{aligned}
\tag{1.6}
$$

where $\boldsymbol{W}$ denotes the weights of the neural network for feature engineering, $\boldsymbol{y}_i'$ is the ground-truth outcome of ith trial.

1.3.4 Inference

What we described in the previous section is a probabilistic model that captures the relationship between observed outcomes and underlying variabilities due to technicians or reagents. With such a probabilistic model, we can predict the probability of a trial succeeding, given which technicians and reagents were involved in each operation of the trial.

Such a probabilistic model could provide actionable insights into achieving the following two goals:

Goal 1: Given an unsuccessful trial, we can identify the technician or reagent that most likely leads to an unsuccessful trial. Identifying the technician or reagent that most likely leads to the unsuccessful trial is straightforward using our probabilistic model. As shown in Equation 1.4, our probabilistic model predicts the probability of success for each technician and reagent in the trial. Given the technicians and reagents involved in each operation of an unsuccessful trial, we plug in their embeddings in Equation 1.5, and we immediately have the probability of success of each technician and reagent. We could pay more attention to analyzing the technicians or reagents with a lower probability of success.

Goal 2: We can identify inaccurate technicians or degraded reagents that are due for re-training (technician) or replacement (reagent). Arguably a more exciting and constructive insight that our method could offer is the ability to rank the relative accuracy of technicians and reagents. The technician or the reagent with a lower accuracy compared to its peers can

be tagged for retraining technicians or replacing reagents.

Ranking technicians and reagents based on their relative accuracy or qualities compared to other technicians and reagents can be achieved by considering the following intuition: if the probability of success of trials can be improved by substituting a technician with a second technician with all the other conditions being identical, then the second technician is considered to have higher accuracy. Without the loss of generality, let us consider the following example. Assume that technician Alice and technician Bob are tasked to perform the same operation of a trial. Each of them would individually perform ten repetitions of the trial respectively. Intuitively, we could say Alice has higher accuracy compared to Bob, if Alice has more successful trials than Bob with *all the other conditions being the same*. Here the other conditions include the technicians and reagents that are involved in the trials other than Alice and Bob. One strategy to keep all the other conditions the same so we can perform sensitivity analysis solely on Alice and Bob is to place Alice and Bob in one operation of a trial, and randomly sample technicians and reagents to fill the slots in the other operations. Using our probabilistic model, we can predict if the trial is successful, given the technicians and reagents involved in each operation. We repeat the random sampling procedure several times and record the number of failed trials. Based on the number of failed trials, we could rank the accuracy of the technicians and the qualities of the reagents.

Note that with such a strategy, the relative accuracy or quality is conditioned on the specific operation. This is highly desirable in practice, as a technician may perform one operation (task) with high accuracy, but may not have the same accuracy level at other operations (tasks). The technician with low relative accuracy can focus on the operation or task the technician is not good at during retraining, and not spend time on the operations that the technician can already accurately perform. We will demonstrate the efficacy of our method with both synthetic and real-world datasets.

1.3.5 Discussion

Aside from the logistic regression-based model described in this section, a few other relevant methods could also be employed for similar objectives. However, their severe drawback in our specific application makes them less favorable than our method. Bayesian networks could also be used in the context of root cause analysis [46]. However, it is well known that the learning of Bayesian Networks does not scale well [46]. It is challenging to use Bayesian Networks with nearly one hundred predictor variables. Gradient boosting-based methods [39] have been widely adopted in statistical analysis. We will show that our proposed method outperforms XGBoost [47], as our method better exploits the structure of the problem.

1.4 Synthetic Data Generation

This section introduces the syntax of *OOPS* for modeling biological workflows. Formalization of the syntax of a protocol language in Abate *et al.* [48] has been adopted in this work.

State of the lab can be considered as the collective state of all the test tubes, beakers, flasks, and other lab items at any particular time. To change the states of containers, we introduce *operations* as discrete, self-contained procedures on a set of reagents performed by a technician, such as retrieving reagents and pipetting. A process P manipulates a list of containers through a set of operations and finally yields a new container as a result. A set of containers is defined as an *inventory*. Consequently, we model a process in the lab as a series of inventory states followed by a series of operations on those states:

$$i_0 \xrightarrow{O_0} i_1 \xrightarrow{O_1} i_2 \xrightarrow{O_2} \dots \tag{1.7}$$

Algorithm 1 shows how to create a new inventory by calling the constructor.

Algorithm 1 OOPS - Create a New Inventory

1: i_0 = oops.Inventory()

In OOPS, each *container* can have various attributes, including volume, template, pH value, and the properties of the reagent stored in the container. Different containers may have different initial conditions. We could *create* containers or operations. Other operations include retrieve, in which modeling a technician takes something from a freezer and brings it to the lab bench; *store*, in which modeling a technician returns a container to the freezer.

Protocols are simply sets of operations to be performed on inventory states. Operations are needed to be used inside Protocols in OOPS.

We now use a cell-free protein synthesis (CFPS) workflow as an example to explain how OOPS models a wet-lab protocol. In biological machinery, CFPS aims to produce protein without living cells.

Example 1. Figure 1.5 shows a sample illustrating how to use states to describe a wet lab protocol. This sample protocol involves two states, three actions, and four reagents. For state i_0, the inventory contains four containers containing plasmid, master mixes, E. coli cell extract, and an empty one for cell-free protein synthesis (CFPS) reaction. Each of them has its unique volumes.includes retrieving plasmid, reaction mixture, and E. coli, and pipetting them to the reaction container. The volume of each containers in i_1 is marked accordingly in figure 1.5. Action a_1 transfer plasmid, master mixes, and E. coli into CFPS reaction tube. The last action a_2, stores all four test tubes. State i_1 records the inventory of four test tubes after the operation retrieves, transfers, and stores. Algorithm 2 demonstrates how to use protocol and operations in OOPS.

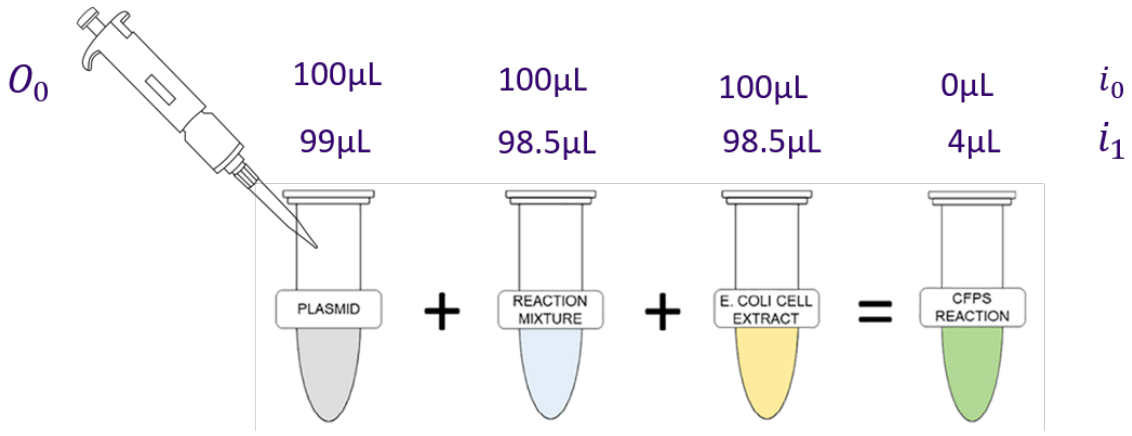


Figure 1.5: A sample illustrates how to use states to describe a wet lab protocol. This sample protocol involves two states, three actions, and four reagents. For state i_0, the inventory contains four containers containing plasmid, master mixes, E. coli cell extract, and an empty one for cell-free protein synthesis (CFPS) reaction. Each of them has its unique volumes. Action a_0 includes retrieving plasmid, reaction mixture, and E. coli, and pipetting them to the reaction container. The volume of each container in i_1 is marked accordingly. State i_1 records the inventory of four test tubes after the operation retrieves, transfers, and stores.

Algorithm 2 OOPS - States

```
plasmid = oops.Inventory([{"volume":100. * u.uL,
                           f'plasmid': (1.00 + random.random()) *3*u.nanomole/u.L}])
master_mixes = oops.Inventory([{"volume":100. * u.uL,
                           f'master_mixes': (1.00 + random.random()) *4 * u.nanomole/u.L}])
E. coli = oops.Inventory([{"volume":100. * u.uL,
                           f'e_coli': (1.00 + random.random()) *3 * u.nanomole/u.L}])
protocol = oops.Protocol(
    create({"volume": 0 * u.uL}, name = "cfps_reaction_tube")
    inv = plasmid(1) + master_mixes(1) + e_coli(1) + cfps_reaction(1)
    retrieve(plasmid, master_mixes, e_coli, cfps_reaction)
    transfer ("plasmid", "cfps_reaction_tube", 1 * u.uL, sigma = 0.05)
    transfer ("reaction_mixture", "cfps_reaction_tube", 1.5 * u.uL, sigma = 0.05)
    transfer ("E. coli_Extract", "cfps_reaction_tube", 1.5 * u.uL, sigma = 0.05)
    store(plasmid, master_mixes, e_coli, cfps_reaction)
)
```

Control actions can be used to wrap operations in OOPS.

1. *maybe*(operation, p = 0.5): operation will be executed with a probability of 0.5.

2. *choose* (operations, probs) takes a list of operations and a list of associated probabilities as inputs. These two inputs are of the same length. A particular operation will be chosen based on its associated probability.

3. *either* takes two operations and one probability of operation one as input. It is equivalent to *choose* shown in Algorithm 3.

Algorithm 3 OOPS - Control Actions (Either)

1: **choose**([operation1, operation2], $(p, 1-p)$

4. *uniformRandomPick* allows users to randomly select a container using a uniform distribution, where a user-specified operation list defines its range. Figure 1.6 shows an example usage of control action uniformRandomPick. In this setup, we have four containers that contain plasmids. We can randomly pick one of them using the control action uniformRandomPick. A sample algorithm implemented in OOPS is shown in Algorithm 4.

Algorithm 4 OOPS - Control Actions(uniformRandomPick)

1: **uniformRandomPick**(
[**steps**(**transfer**"template","m", $1 * u.uL$, sigma = 0.05, **log**("tech_id", 1)),
steps(**transfer**"template","m", $1 * u.uL$, sigma = 0.1, **log**("tech_id", 3))])

5. *steps*(operations) wrap all operations. The system can only execute one operation at a time. Operations are executed one after another.

6. *skip* indicates no actions will be taken.

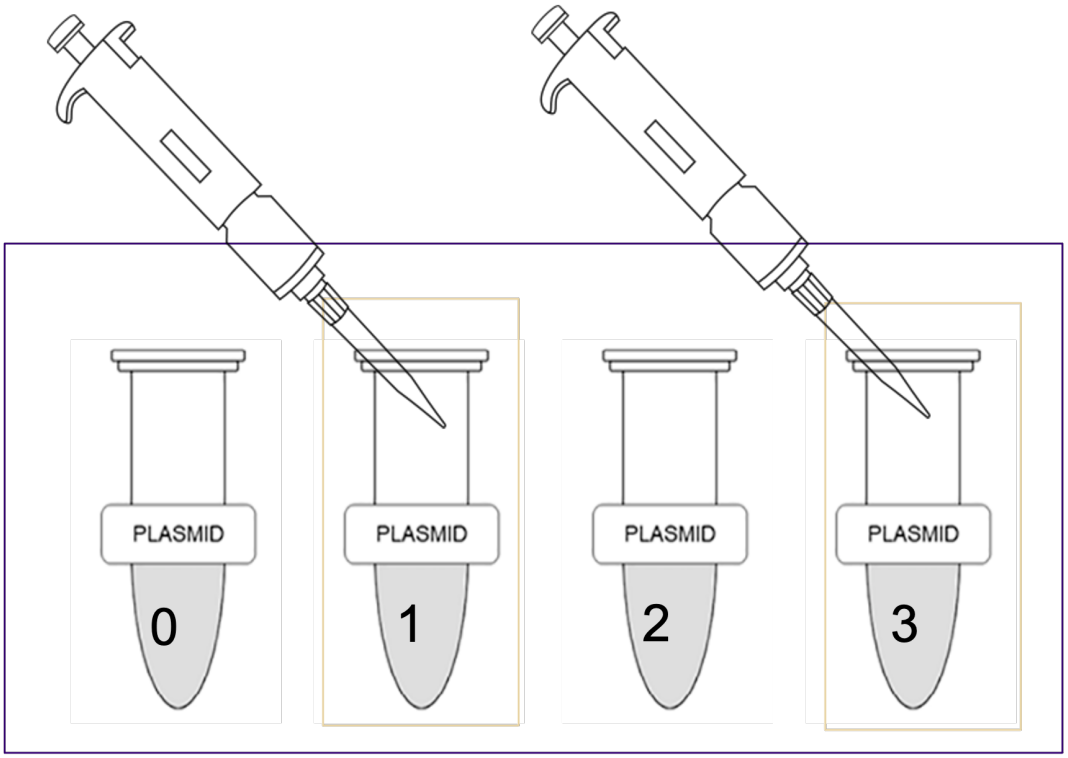


Figure 1.6: Sample usage of control action uniformRandomPick. In this setup, we have four containers that contain plasmids. We can randomly pick one of them using control action *uniformRandomPick*. The chance of retrieving plasmid 1 and plasmid 3 are equal.

7. *say*(message) prints out message as system outputs. It will not change inventory.

In biological workflows, the accuracy of technicians is one of the significant root causes of experimental failure. OOPS is able to model the accuracy of technicians as a failure mode. Three types of accuracy issues can be modeled in the current system. We can model a technician occasionally retrieves the wrong test tube from the freezer, mislabels the test tube, and transfers the volume varies with variations.

Example 2. Figure 1.7 shows that a technician may mislabel the test tube occasionally. A technician is instructed to create a container with the DNA "ATCG". However, there is a chance he/she creates a container with the DNA "CATG" and mislabelled it as the original DNA in desire. Algorithm 5 shows how to model this absent-minded technician using OOPS.

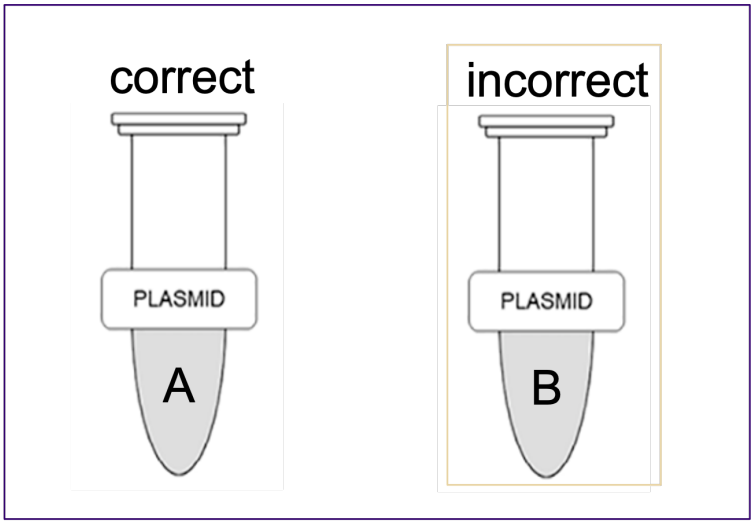


Figure 1.7: A technician may occasionally mislabel the test tube.

The other source of variability introduced by technician performance is modeled using liquid handling. Moving liquids from one container to another is known as liquid handling. However, even robots are unable to transfer liquids with perfect accuracy.

Algorithm 5 OOPS - Modeling an Absent-minded Technician

```
1: error_prone_create = oops.Protocol(
     either(
       steps (create({"sequence":"ATCG"}, name="my_first_primer")),
       steps (create({"sequence":"CATG"}, name="my_first_primer"), say("oops!")
     ), p = 0.5 # half of the time, the technician gets it right ) )
```

Example 3. As shown in Figure 1.8, a liquid handling agent (human or robot) is instructed to draw up one microliter from container A and then transfer one microliter into container B. Algorithm 6 demonstrates how to model liquid handling using *transfer* operation in OOPS. This transfer operation decreases the volume in container A and increases the volume in container B. This transfer operation asks for a *variance* as an input. The absolute volume that has been transferred is modeled by a normal distribution with a zero mean and technician's variance. Note that different technicians have different variances that represent their accuracy.

Algorithm 6 OOPS - Liquid Handling

```
1: transfer("plasmid","cfps_reaction", 1 * u.uL, sigma=variance)
```

The physical process underlying a biological workflow is modeled as a reaction. We keep track of the results of reactions, since many reactions are finished before a test tube is put back in storage.

Example 4. consider a simple chemical reaction

$$A + B \rightarrow C \tag{1.8}$$

where two aliquots from tubes holding A and B are joined into a third tube. Either all of A is consumed, all of B is consumed, or both are consumed in the reaction to produce C. Assume $a_0 > b_0$, and starting concentrations of A and B in the reaction tube are $[A] = a_0$

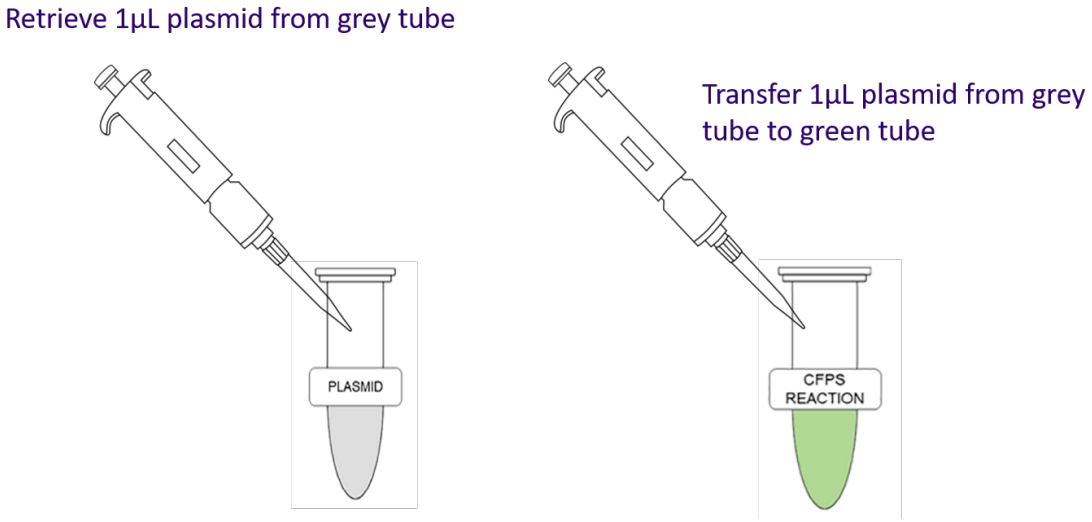


Figure 1.8: The technician is instructed to transfer $1uL$ plasmid from the grey tube to the green tube. However, the absolute volume of liquids transferred is modeled as a normal distribution with a zero mean and technician variance.

and $[B] = b_0$, respectively. B becomes the limiting component at that point. Therefore, the reaction will stop after finishing up B.

$$[A] = a_0 - b_0$$
$$[B] = 0$$
$$[C] = b_0$$

The OOPS implementation is presented in Algorithm 7. Once we define the protocol, we can simulate it in OOPS. Algorithm 8 shows how to simulate the protocol one thousand times.

In Section 1.5.1, we illustrate the usefulness of OOPS in two case studies, highlighting the potential impact of our work on scientific discovery.

Algorithm 7 OOPS - Modeling Reactions

```
1: inv = oops.Inventory()(
            .create({"A": 10 * u.umole / u.L, "volume": 100 * u.uL}, name="A")
            .create({"B": 15 * u.umole / u.L, "volume": 100 * u.uL}, name="B")
            .create({"volume": 0 * u.uL}, name="C")
2: sample_reaction = oops.Protocol(
            retrieve_by_name("A", bench_name = "a")
            retrieve_by_name("B", bench_name = "b")
            retrieve_by_name("C", bench_name = "c")
            transfer("a", "c", 10 * u.uL)
            transfer("b", "c", 10 * u.uL)
            reaction(
                "c"["A","B", "C"], )
                lambda a, b, c: [a - min(a, b), b - min(a, b), c + min(a, b)]
            ),
            store("a"),
            store("b"),
            store("c")
   )
```

Algorithm 8 OOPS - Simulations

```
1: sample_reaction.reset()
2: num_sims = 1000
3: for each i in range(num_sims) do
4:     sample_reaction.run(inv)
```

1.5 Experimental Results

We demonstrate the efficacy of our method on synthetic datasets and real-world datasets collected from a biological laboratory. We illustrate how our framework can be employed to predict outcomes given the involvement of technicians and reagents, identify the root causes of the failed trials, and identify degraded reagents and technicians with low accuracy. The flexibility of our proposed framework is demonstrated using combinations of operations. Various standard outputs are interrupted for the proposed model. All evaluation metrics are adopted from [49].

1.5.1 Details of Synthetic Datasets Generation

The motivation and advantage of using a synthetic dataset have been discussed in Section 1.3.1. In the same section, we also described the general procedure of synthesizing training data with ground-truth labels using the OOPS framework, utilizing the syntax introduced in Section 1.4. We now present details of two case studies, namely cell-free protein synthesis (CFPS) and Polymerase Chain Reaction (PCR), illustrating how the proposed framework can be employed to model biological workflows. Furthermore, we demonstrate that the proposed framework can be employed to perform automated analyses of the protocol parameters while accounting for the uncertainty in the performance of technicians and the qualities of reagents.

Synthesis of Technician and Reagent Feature

The features of technicians and reagents carry important information that is highly valuable for predicting the accuracy of technicians and reliability of reagents, and are the inputs to our root causes analysis framework.

We generate feature and identity encoding (ID) in OOPS for each technician and reagent. As shown in Algorithm 9, each technician and reagent is assigned a randomly generated vector of unit length as their identity encoding. Then the proposed system logs the involvements of the technicians and reagents by using their identity information.

Algorithm 9 Generate Identity Information

1: **function** GenIdentity($\boldsymbol{R}$, $\boldsymbol{T}$) **return** a dictionary
2: **inputs:** technician list $\boldsymbol{T}$, reagent list $\boldsymbol{R}$
3: **local variables:** *iDict*, a dictionary that stores identity for each reagent r and technician t
4: **for each** $r \in \boldsymbol{R}, t \in \boldsymbol{T}$ **do**
5: $iDict \leftarrow$ Random-Vector(unit length)

Algorithm 10 and Algorithm 11 are the pseudo codes to generate technician encodings and reagent encodings, respectively. We generate the statistics of prior success for each technician to represent their accuracy. These statistics involved four pieces of information.

1. We count the number that the technician has attempted each operation.

2. The number of successful and failed trials the technician has performed is counted.

3. Let X^{ber} be samples from a Bernoulli distribution, where the probability of success p is the technician's accuracy.

4. We randomly generated fifty data points to synthesize the number of successes and failures trials a technician has performed. Here we cannot generate too much data due to the Law of large numbers. Since using too much data could directly feed the accuracy of each technician into the model. Out of these generated fifty samples, the number of successful and failed trials are logged.

For each reagent, we model the age of the reagent by adding Gaussian noise to the quality of the reagent.

Algorithm 10 Generate Technician Encoding

function GenTechEncoding(*trDict, iDict*) **return** technician encoding

inputs:*trDict*, a dictionaries that store accuracy of each technician, *iDict*, a dictionary that stores identity for each technician t, trails $\boldsymbol{S}$, operations $\boldsymbol{O}$, technician list $\boldsymbol{T}$,

local variables: u^t is the technician ID for technician t

$f^{t,o,1}$ is the number that the technician t has attempted the operation o

$f^{t,o,2}$ and $f^{t,o,3}$ are the number of trials that technician t completed and aborted

$f^{t,o,4}$ and $f^{t,o,5}$ are the number of successful and failed trials that the technician t has performed

x^t is the technician encoding for technician t

for each $t \in \boldsymbol{T}$ **do**

  for each $s \in \boldsymbol{S}$ **do**

    for each $o \in \boldsymbol{O}$ **do**

      count the number that t has attempted for each operation $f^{t,o,1}$

    count the number of successful$f^{t,o,4}$ and failed $f^{t,o,5}$ trials that t is involved in

  for $i = 1$ to 50 **do**

    $X^{ber} \sim Bernoulli(r^t)$

  count the number of successful$f^{t,o,2}$ and failed $f^{t,o,3}$ trials

  $technician\_identity \leftarrow iDict(u^t)$

  $x^t \leftarrow f^{t,o,1} + f^{t,o,2} + f^{t,o,3} + f^{t,o,4} + f^{t,o,5} + technician\_identity$

Algorithm 11 Generate Reagent Encoding

function GenReagentEncoding(*rqDict, iDict*) **return** reagent encoding
    inputs: *rqDict*, a dictionary that stores the quality of each reagent, *iDict*, a dictionary that stores identity for each reagent r, reagent list $\boldsymbol{R}$
    local variables: u^r is the reagent ID for reagent r, x^r is the reagent embedding for reagent r
    for each $r \in R$ **do**
        $quality\_clean\_signal = iDict(u^r)$
        $\mu_{noise}, \sigma_{noise} = 0, 0.1$
        $Gaussian\_noise = random.normal(\mu_{noise}, \sigma_{noise})$
        $reagent\_identity = iDict(u^{t,o})$
        $x^r \leftarrow (quality\_clean\_signal + Gaussian\_noise) + reagent\_identity$

Modeling PCR Workflow

PCR is a fundamental workflow to make various copies of a specific region of DNA in a test tube [50]. We model polymerase chain reaction (PCR) workflow to generate synthetic datasets. Using OOPS, a large dataset of PR experimental records can be synthesized by parameterizing the accuracy of technicians and quality of reagents,

PCR workflow is modeled by tracking the results of reactions. Initially, a technician retrieves and transfers the reagents: primers, template, and master mix, to a tube. This liquid transfer process could potentially introduce errors, since the technician cannot transfer liquids with absolute precision. This approximation is modeled by a normal distribution, with a zero mean, and the standard deviation σ is the technician's accuracy. All reagents have the attribute of volume, concentration, and age. The liquid transfer updates concentrations and volumes accordingly. Therefore, the concentration of the reagent is updated accordingly based on the volume change. Furthermore, the reagent's age affects the reactants' concentration, which in turn affects the reaction rate. The concentration and volume of the products

(in most cases, a reagent) are the keys to determining the outcome of each trial.

PCR workflow creates amplicons as the product. Ideally, all the primers are converted into amplicons. However, the failed trial occurs when primers do not bind to each other or bind nonspecifically to undesired regions of the template. If the concentration of the resulted amplicons is high, we consider the trial is successful. Conversely, a failed trial would have a low concentration of the amplicons. The other common source of the error is technician performance, such as technicians do not follow the workflow procedure, or retrieve the wrong reagents. Algorithm 12 shows the pseudo-code of modeling PCR workflow with both root causes.

Algorithm 12 Modeling PCR Workflow using OOPS

function SYNTHETICPCR(*trDict, rqDict*) **return** a dataset

inputs:*trDict, rqDict*, dictionaries that store the accuracy of each technician and the quality of each reagent, respectively, trails $\boldsymbol{S}$, technician list $\boldsymbol{T}$, reagent list $\boldsymbol{R}$

local variables: forward primer FP, reverse primer RP, and master mix M are reagents

amplicon A

u^t and r^t are the ID and accuracy of a technician t

$u^{r,X}$ and $r^{r,X}$ are the ID and quality of a reagent X

y^s is the outcome of the trial s

for each $s \in \boldsymbol{S}$ **do**

  $u^t \leftarrow$ Random-Select($\boldsymbol{T}$)

  $u^{r,FP}, u^{r,RP}, u^M \leftarrow$ Random-Select($\boldsymbol{R}$)

  $r^t \leftarrow trDict(u^t)$

  $r^{r,FP}, r^{r,RP}, r^{r,M} \leftarrow rqDict(u^r)$

  rate = $Normal(0, r^{r,M}) \times 0.9 \times Normal(0, min(r^{r,FP}, r^{r,RP}))$

  Volume(A) = Volume(M) $\times$ rate $\times$ min(Volume(FP), Volume(RP))

  if Conc(A) is low **then** $y^s = 0$

  else $y^s = 1$

Modeling CFPS Workflow

Cell-Free Protein Synthesis (CFPS) produces protein by employing biological machinery instead of living cells [51]. As a result, researchers can quickly generate and produce modest quantities of functional proteins using CFPS workflow. A cell extract, an energy source, a supply of amino acids, cofactors like magnesium, and DNA containing the needed genes are common reactants of a cell-free reaction workflow. A cell extract is produced by lysing the target cell and centrifuging cell walls, DNA, and other detritus. The remaining components of the cell, such as ribosomes, aminoacyl-tRNA synthetases, translation initiation, and elongation factors, nucleases, are necessary cell machinery. We assume these nutrients for gene expression are in infinite supply while modeling the CFPS workflow.

In CFPS, linear expression templates and plasmids are the two forms of DNA that can be utilized. Circular plasmids are exclusively produced inside cells. PCR, which duplicates DNA far more quickly than growing cells in an incubator, can produce linear expression templates much more efficiently. Plasmid yields are typically substantially greater in CFPS, even though linear expression templates are more straightforward and quicker to manufacture. In this study, we employ the reference plasmid P70a-deGFP, in which the sigma 70-specific promoter P70a has been used to clone the gene deGFP, which produces the reporter protein deGFP.

An essential component of a cell-free process is an energy source. Typically, the extract is combined with a supply of amino acids and a separate mixture containing the required energy source for the process. Phosphoenol pyruvate, acetyl phosphate, and creatine phosphate are typical sources. These energy sources are added to extract in this work.

Cell extracts derived from E. coli, rabbit reticulocytes, wheat germ, insect cells, and yeast Kluyveromyces are commonly used nowadays [51, 52]. These extracts are all commercially available. E. coli is the most commonly used lysate. It is the cheapest extract to make and takes the least amount of time. Furthermore, vast numbers of E. coli can be easily cultured and subsequently lysed using a homogenizer or a sonicator [51]. E. coli has the

highest protein yields as well. High yield production, however, can limit the complexity of the synthesized protein, particularly in post-translational modification. In this aspect, lower efficient eukaryotic systems may be helpful if modifying enzyme systems are preserved in the extracts. We use P70a in this sample CFPS protocol. According to reports, the phage lambda-derived P70a is one of the most potent E. coli promoters ever discovered.

Figure 1.9 shows how to model CPFS workflow as an OOPS protocol. We follow Cole *et al.*'s work to model the source of variabilities of CFPS [1]. We model two sources of variabilities: technicians transfer different volumes of reagents, or degraded reagents are used. This OOPS protocol starts with randomly selecting a technician and a plasmid. The plasmid may be fresh or degraded. The selected reagent introduces the reagent issue into the CFPS protocol. Then the selected technician retrieves the plasmid, master mix, and E.col cell extract and creates a new test tube for a reaction. Next, all reagents are transferred to the reaction tube, in operation three. Since tech cannot transfer the absolute volume of liquids, this transfer process introduces a performance error. Operation four is the CFPS reaction. Finally, the technician stores all reagents in operation five after the reaction completes.

For CFPS workflow, we adopt an ODE-based model proposed by Ryan Marshal to predict the behavior of biological components in cell-free extracts [44]. This ODE-based model uses relevant kinetic constants and concentrations of molecular components to describe the kinetics of deGFP synthesis. This mechanistic-based model involves many factors that affect the reaction. Among these factors, the most critical factors are plasmid concentration and the ratio of the plasmid to the cell-free lysate. Algorithm 13 presents the implementation of CFPS reaction using OOPS. Algorithm 14 shows how to simulate the CFPS workflow to obtain a data set.

Figure 1.10 is a scatter plot of the mass of matured deGFP mRNA to technician id and template id. X axis is (technician ID, template id), and the y-axis is the mass of matured protein. Note that each template has a unique concentration, and each technician has a unique variance that describes their accuracy. The trials performed by different technicians are plotted in different colors. For example, trials performed by technician zero are colored

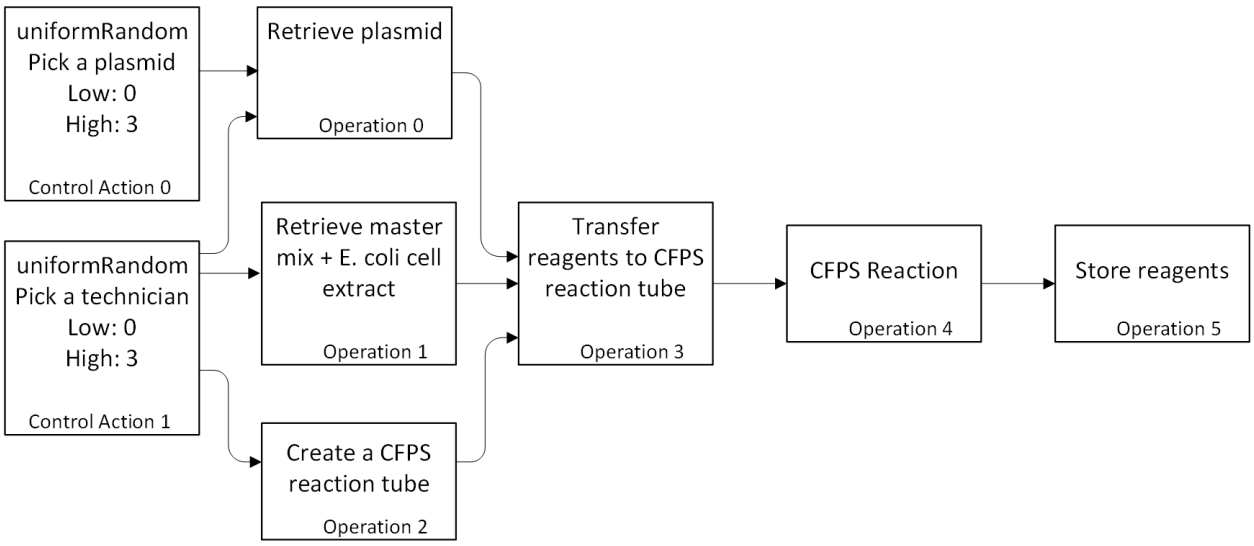


Figure 1.9: Modeling CPFS workflow as an OOPS protocol. Each box is either a control action or an operation. Once selecting a technician and a plasmid to perform the trial, all reagents are retrieved and transferred to a new test tube for reaction. Operation four "CFPS reaction," models a CFPS workflow based on a physics-based model. All reagents are stored after the reaction finishes.

Algorithm 13 Modeling a CFPS Reaction using OOPS

function SYNTHETICCFPS(E_t, C_m, R_t) **return** a dataset

inputs: trails $\boldsymbol{S}$, total concentration of active core E. coli RNAP [nM] E_t, rate of transcription [nt/s] C_m, concentration of ribosomes [nM] R_t

local variables: total concentration of sigma 70 [nM] ($S_{70t} = 30$), P70a transcription rate [s^{-1}] ($k_cm = 0.065$), MM constant for mRNA synthesis [nM] ($Km = 1$), total concentration of sigma 70 ($S_{70t} = 30[nM]$), rate of mRNA degradation [nM/s] ($k_{dm} = 6.6$), MMs constant for RNA degradation ($K_{mm} = 8000$), translation rate constant [s^{-1}], ($k_{cp} = 0.006$), translation MM constant[nM] ($K_{mr} = 10$), dissociation constant sigma 70 and E. coli core RNAP [nM] ($k_{70} = 0.26$), length of messenger RNA [nt] ($L_m = 800$), rate of translation [nt/s] ($C_p = 2.5$), rate of protein maturation [s^{-1}], ($k_{mat} = 0.000725$)

for each $s \in \boldsymbol{S}$ **do**

calculate the concentration of free RNA polymerase

e_0 = lambda $E_0 : E_0 + E_0 * S_{70t}/(k_{70} + E_0) + E_0 * S_{70t} * P_{70\_conc}((i-1)/2)/(E_0 * S_{70t} + K_m * (k_{70} + E_0)) * (1 + k_{cm} * L_m/C_m) - E_t$

E_0 = fsolve$(e_0, E_t)[0]$ # concentration of free core RNA polymerase $[nM]$

degfp mRNA (estimate using forward Euler)

$y\_pre = m[i-2]$ if $(i > 1)$ else 0

$y\_i = y\_pre + h * (k_{cm} * P_{70\_conc}((i-1)/2) * E_0 * S_{70t}/(E_0 * S_{70t} + K_m * (E_0 + k_{70})) - k_{dm} * y\_pre/(K_{mm} + y\_pre))$

calculate the concentration of free ribosomes R_0

r_0 = lambda $R_0 : R_0 + R_0 * y_i/(R_0 + K_{mr}) * (1 + k_{cp} * L_m/C_p) - R_t$

R_0 = fsolve$(r_0, R_t)[0]$ # concentration of free ribosomes

non-mature deGPF (estimate using forward Euler)

y_dark_pre = m_dark[i - 2] if (i ¿ 1) else $(k_{cp} * y_i * R_0/(K_{mr} + R_0))$

$y\_dark\_i = h * (y\_dark\_pre + k_{cp} * y_i * R_0/(K_{mr} + R_0) - k_{mat} * y\_dark\_pre)$

mature deGPF (estimate using forward Euler)

$y\_mat\_pre = m\_mat[i-2]$ if $(i > 1)$ else $(k_{mat} * y\_dark\_i)$

$y\_mat = y\_mat\_pre + h * (k_{mat} * y\_dark\_i)$

Algorithm 14 OOPS - Simulation

```
1: for i = 0 to 1000 do
2:     cfps_protocol.run(inv)
```

in blue. The circled blue points and orange points involve the same template zero. Blue points are trials involving technician zero, with a variance of 0.05. Orange points are trials performed by technician one, with a variance of 0.5. Since the volume of retrieved reagents is a normal distribution, with a variance of involved technicians and a zero variance. The final product of technician one, which has a larger variance, is more spread out compared to technician zero, which has a smaller variance, since the volume ratio of reagent and lysate affects the concentration of the final product and vol of the final solution. The circled red points are used to show how the quality of templates affects the mass of matured deGFP mRNA. For any given technician, we group trials by template IDs. For example, the fourth group of red points (counting from left to right) is the trails involving technician three and template three. Template three has a concentration that is higher than five nanoMolar. We consider template three a fresh template, since the template concentration affects the final product's reaction rate. Template two is degraded due to its low concentration. If a line is drawn to indicate trials with a high enough mass of matured protein, we found that a fresh template leads to higher probabilities of a high mass of matured protein. Generally speaking, a combination of a technician with a smaller variance and a template with a concentration higher than five nanoMolar will more likely lead to a high mass of matured protein. Based on this information, we could associate the failed trial with the involvement of technicians and reagents. Also, we could identify technicians who have high-performance error rates or degraded reagents that are due for retraining or replacement.

OOPS allows users to construct a model of the experimental protocol that includes various possible failure modes. As shown in Figure 1.11, we demonstrate a variation of the CFPS workflow described above. The previously described model contains two sources of variabilities, (1) technicians transfer different volumes of reagents, and (2) degraded reagents are

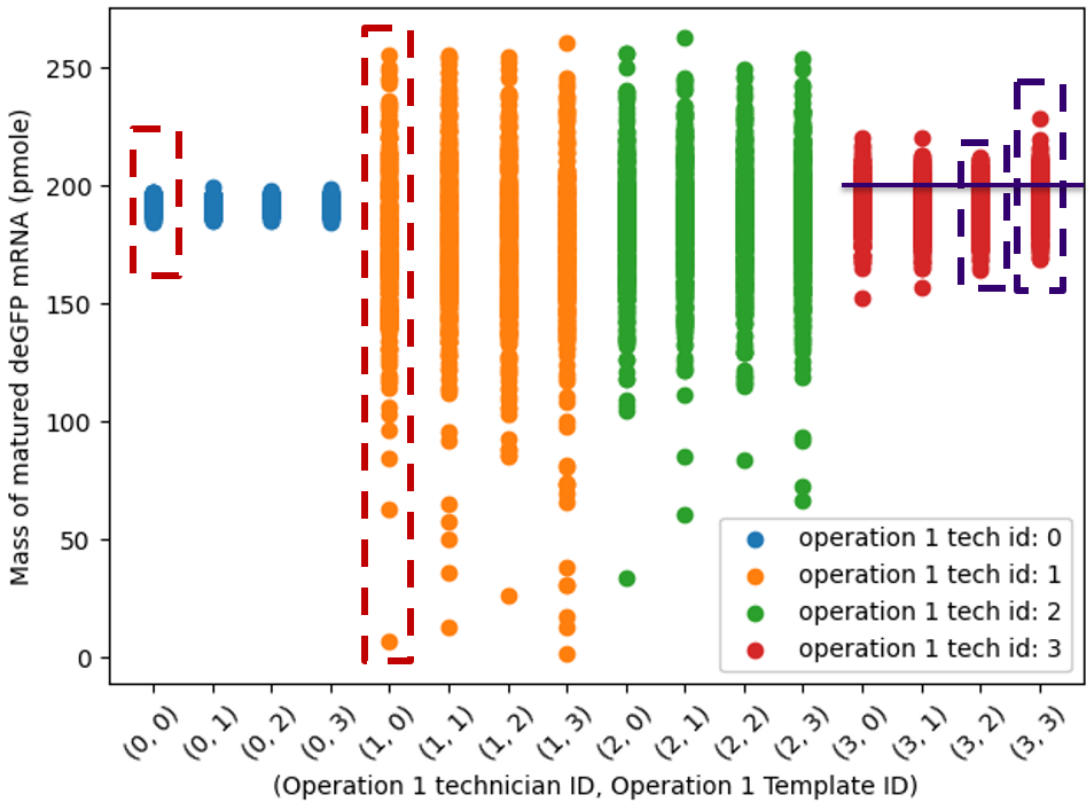


Figure 1.10: A scatter plot of the mass of matured protein to technician id and template id. Generally speaking, if we have a combination of a technician with a smaller variance and a template with a concentration higher than five nanoMolar, we tend to have a high mass of matured protein.

used. This variated CFPS workflow contains an additional type of error that the technician may retrieve the wrong test tubes. The operation that introduces this additional error is highlighted in red. In operation zero, the selected technician may have a chance to retrieve the wrong test tube. This performance error causes the trial to fail. Figure 1.12 shows this additional error that the technician retrieves a wrong test tube leading to zero mass of matured deGFP mRNA.

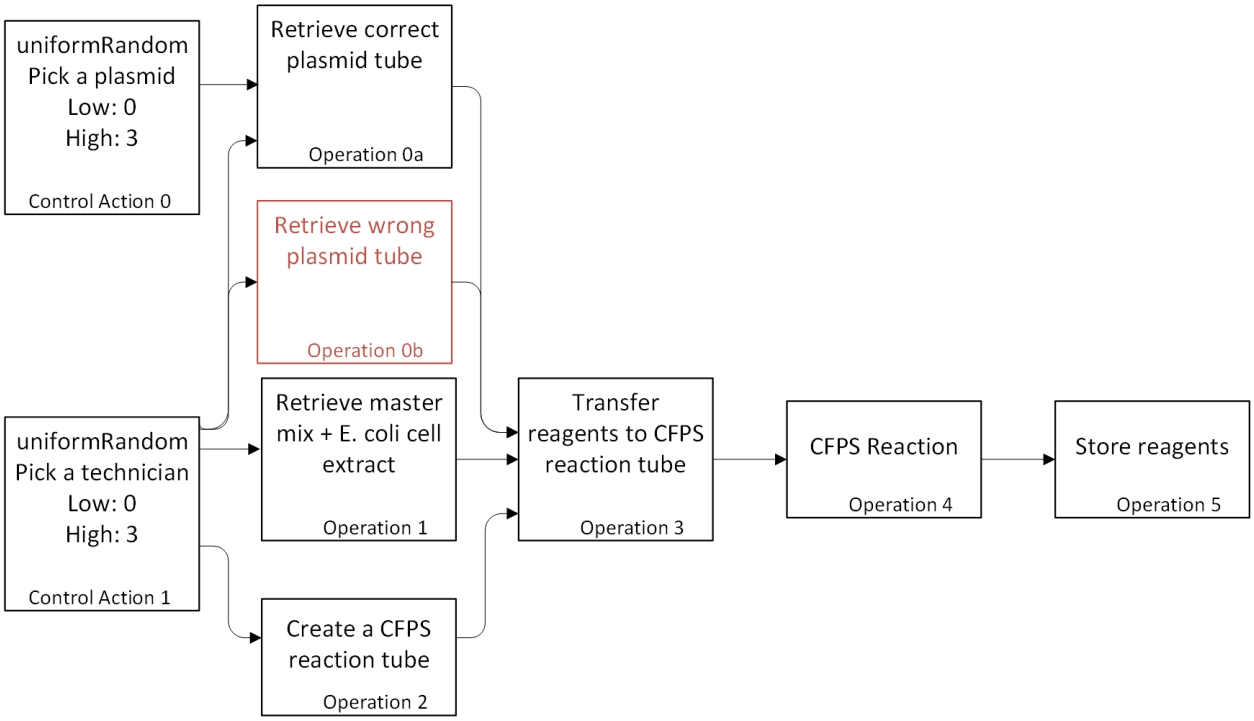


Figure 1.11: A variation of CFPS workflow. This sample workflow includes three sources of variabilities: (1) technician transfers different volumes of reagents, (2) degraded reagents, and (3) technician may retrieve the wrong test tubes.

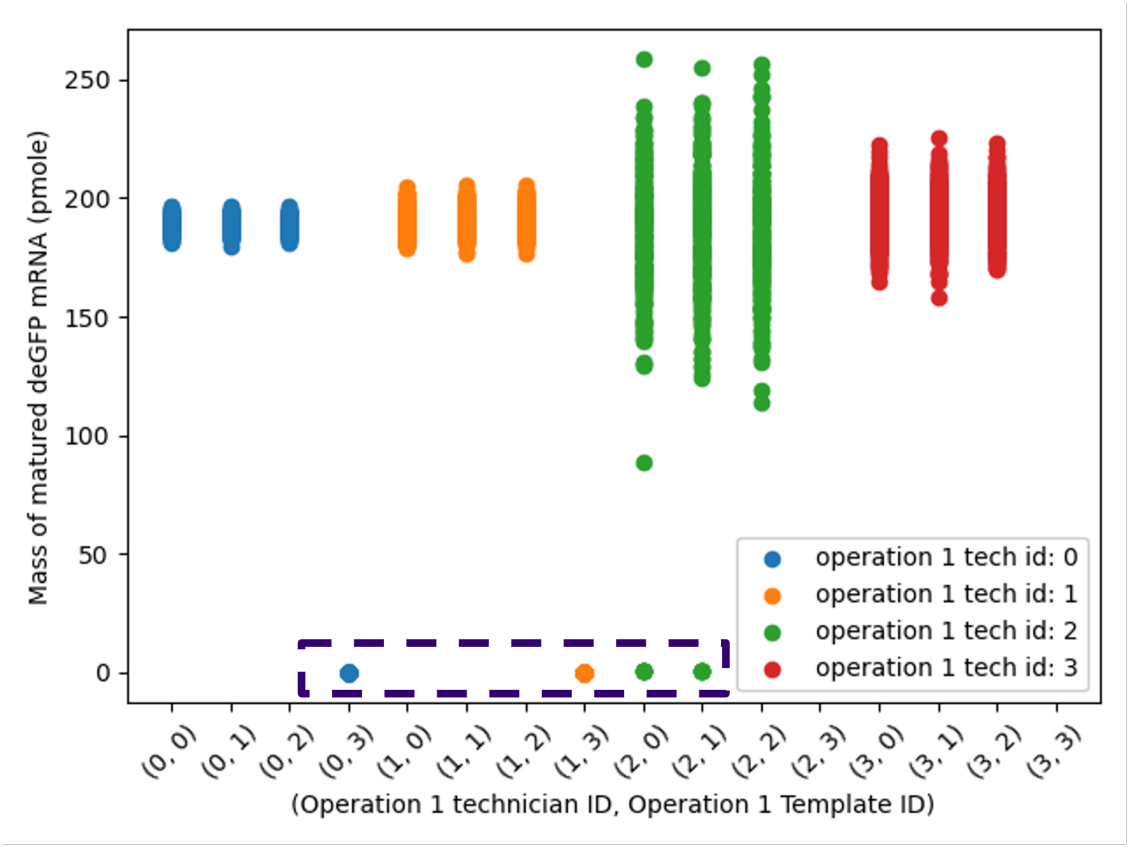


Figure 1.12: The technician retrieves the wrong test tube leading to zero mass of matured protein.

Embeddings

In our proposed method, some input data (reagent IDs and technician IDs) are nominal or categorical data. Nominal data or categorical data are discrete, while numerical data are continuous. Commonly used modeling methods prefer data in numerical form. Thus it is essential to convert nominal variables into numerical variables. We introduce a customized nominal data encoding method called embedding.

We learn an embedding, which is detailed in Section 1.2.6, as a neural network for our prediction task. This approach leads to a well-customized for our particular system. Again we use the CFPS workflow as an example. This particular workflow involved one operation, four technicians, and four reagents. Table 1.1 shows a sample raw data of CFPS workflow. This table contains two trials. We know which technician and which reagent is involved in each trial. The outcome of each trial depends on the mass of matured protein.

Table 1.1: A sample raw data of CFPS workflow. Two trials have been presented in this table. We know which technician and which reagent is involved in each trial. The outcome of each trial depends on the mass of matured protein.

	Operation 1 technician ID	**Operation 1 template ID**	**Mass of matured deGFP mRNA (pmole)**
1	2	1	175.895
2	0	0	170.232
...			

We include IDs as part of the model's predictor variables. We know which technician and reagent are involved in each step by including IDs. A randomly generated 1×3 vector to each technician and reagent is used to augment their feature information.

Each technician also contains a 1×2 row vector representing the statistics of the technicians performing certain operations, including the number of successful and failed trials the technician has performed. For example, for all the trials that technician two has performed, he/she has succeeded 450 times and failed 580 times. For reagents, the feature is the age of

the reagent. The row vectors for technician two and template one are shown in Equation 1.9 and 1.10, respectively.

$$\underset{(technician\ ID)}{2} \rightarrow \begin{bmatrix} 0.69 \\ 0.31 \\ 0.96 \\ 450 \\ 580 \end{bmatrix} \tag{1.9}$$

$$\underset{(template\ ID)}{1} \rightarrow \begin{bmatrix} 0.51 \\ 0.32 \\ 0.16 \\ 3.45 \end{bmatrix} \tag{1.10}$$

Next, we replace the technician and template ID with encodings containing identity and feature information for model training. Table 1.2 shows a trial of the generated CFPS dataset. The first nine rows are input data. We also log the real root causes of each trial. These real root causes are used in the synthetic data generation phase. Furthermore, we use ground-truth root causes to evaluate the proposed probabilistic model. These real root causes are not used for model training. Finally, the last row is a binary output for each trial. Since we would like to perform root cause analysis for any trials that do not have a high enough mass of matured deGFP protein, we pick 165 picomole as the cut-off line and mark all trials that have a mass of matured deGFP protein that is lower than 165 picomole as zero outputs. This cut-off line is selected based on various prior works [44, 53] and numerous simulations.

A neural network is trained to generate low-dimensional embeddings for each technician and reagent. During the training process, the weight of the neural network is optimized. Similar embedded vectors are placed closer to each other. The resulted embeddings serve as input to a logistic regression-based model. Both embedding and the logistic regression-based model are implemented using PyTorch. The neural network that performs the automatic fea-

Table 1.2: A trial of the generated CFPS dataset. The first nine rows are input. The next four rows are real root causes. These real root causes are only used in synthetic data generation and model validations. The last row is a binary output that depends on the mass of the final product.

	1	2
Operation 1 technician identity 0	0.69	
Operation 1 technician identity 1	0.31	
Operation 1 technician identity 2	0.96	
Operation 1 technician succ count	450	
Operation 1 technician fail count	580	
Operation 1 template identity 0	0.51	
Operation 1 template identity 1	0.32	
Operation 1 template identity 2	0.16	...
Operation 1 template age	3.45	
Operation 1 template conc	3.61	
Operation 1 template volume	0.84	
Total volume	3.83	
Operation 1 technician variance	0.2	
Outcome	0	

ture engineering is jointly optimized with logistic regression variables. This hybrid approach allows us to train a compact neural network for feature extraction that facilities prediction based on logistic regression with a small-scale dataset.

Statistical Results and Validation

Model Evaluation Metrics We evaluate the model in Figure 1.4 using the following metrics: accuracy, cross-entropy, F1 score, recall, and area under curve (AUC) [49]. Define

- True Positive (TP): Trials that succeed and were also predicted to succeed.
- True Negative (TN): Trials that failed and were also predicted to fail.
- False Negative (FN): Trials that failed, but the predictor says they succeeded.
- False Positive (FP): Trials that succeed, but the prediction says they failed.

The most common diagnostic of a logistic regression model is accuracy, which is the proportion of correct predictions over the total predictions.

$$Accuracy = \frac{True}{True + False} = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.11}$$

Cross Entropy measures the differences between two probability distributions for a given set of events. It is also used as a metric to evaluate model performance.

Recall is defined as the probability of the model predicting an observation's "success" given that in truth. Conversely, specificity is the probability of the model predicting "fail" given that it fails.

$$\begin{aligned} Recall &= \frac{TP}{TP + FN} \\ Specificity &= \frac{TN}{TN + FP} \end{aligned} \tag{1.12}$$

The Area under the receiver operating characteristic (ROC) curve (AUC) has been used to assess the discrimination of the fitted model. The ROC curve is created by plotting

recall against the one minus specificity at various threshold settings. Ideally, the ROC curve should go close to the top left corner of the plot, meaning we simultaneously have a high discrimination ability with high recall and specificity. Otherwise, the ROC curve ends with a 45-degree diagonal line if a model has no discrimination ability.

We always summarize the discrimination ability by reporting the Area under the ROC curve (AUC). For example, AUC = 1 indicates that the model has perfect discrimination ability. In addition, AUC gives a successful classification rate by the logistic regression model.

Precision presents the proportion of positive identifications that were actually correct.

$$Precision = \frac{TP}{TP + FP} \tag{1.13}$$

F1 score is a weighted average of Recall and Precision. It is useful when we deal with an uneven class distribution.

$$F1 = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{1.14}$$

Comparing Embeddings to Manual Feature Engineering In Section 1.5.1, we present the details of modeling PCR workflow. The accuracy of technicians and quality of reagents are used to compute outcomes for each trial of the synthetic dataset. However, these ground truth values are commonly not available in real life. Instead, we have informative features and involvements of technicians and reagents in each operation.

We simulate the PCR workflow one million times to generate a dataset. This large data size allows a fair comparison between embeddings and manual feature engineering. Depending on the accuracy of technicians and the qualities of reagents used to generate the synthetic dataset, the ratio of successful and failed trials can be highly unbalanced. Therefore balancing is required to perform the machine learning task. We handle imbalanced datasets by oversampling the minority class (failed trials). To be more specific, we re-sampled the minority class to make the ratio roughly 1 : 1. This method allows us to use all majority class items, at the cost of overusing the minority class items.

Table 1.3 shows that using embeddings as the feature engineering technique significantly improved model performance compared to approaches that use other types of feature engineering techniques. The same synthetic PCR dataset has been used for this performance comparison. The data size of the synthetic PCR dataset is one million. The success rate is thirty percent. We allocate twenty percent of the data to the test set and eighty percent to the training set. The first column shows the performance metric of the model that uses embedding, which is described in Section 1.5.1. The first column is the performance metric of the model that uses embedding. The remaining columns show the performance metrics of the model using different feature encoding methods. *normalize feature* indicates the usage of standardization to normalize data in each feature information. *identity (one hot)* allows us to use the encoded technician IDs as predictor variables. It converts IDs to indicator columns. With *non-linear*, we add non-linear terms to the predictor variables. *feature (min/max)* denotes the usage the min/max encoding to normalize each column of the predictor variables.

As shown in Figure 1.13, test set accuracy, cross-entropy, f1 score, and recall against AUC are plotted. The result with embedding is highlighted in red. Our neural network-based feature engineering outperforms manual feature engineering. In Table 1.4, a confusion matrix is presented to analyze classification results with the test set using a synthetic PCR dataset.

Table 1.3: Model performance improvement with embeddings with a synthetic PCR dataset. Besides embeddings, we explored various combinations of encoders when preprocessing the raw dataset. The same raw dataset is used for a fair comparison. The data size is one million.

	embedding	normalize feature + identity (one hot)	normalize feature	normalize feature + identity (one hot) + nonlinear	normalize feature + identity	identity (one hot)	feature (min/max) + identity
data size	$1M$ (30% succeed)						
test set accuracy	93.94%	89.4%	88.78%	89.32%	88.79%	89.28%	88.97%
cross-entropy	2.396	3.30	4.17	3.30	4.17	4.710	3.71
F1 score	0.931	0.843	0.906	0.843	0.912	0.883	0.893
recall	0.931	0.843	0.906	0.843	0.912	0.883	0.893
AUC	0.988	0.815	0.786	0.813	0.805	0.759	0.788

Table 1.4: Confusion matrix for failed trial prediction using the proposed probabilistic model based on embeddings with a synthetic PCR test set.

Actual	Total	Predicted as	
		Success	Failed
Failed	12096	11436 (TN) 94.54%	660 (FP) 5.4%
Success	187904	1489 (FN) 0.79%	186415 (TP) 99.21%

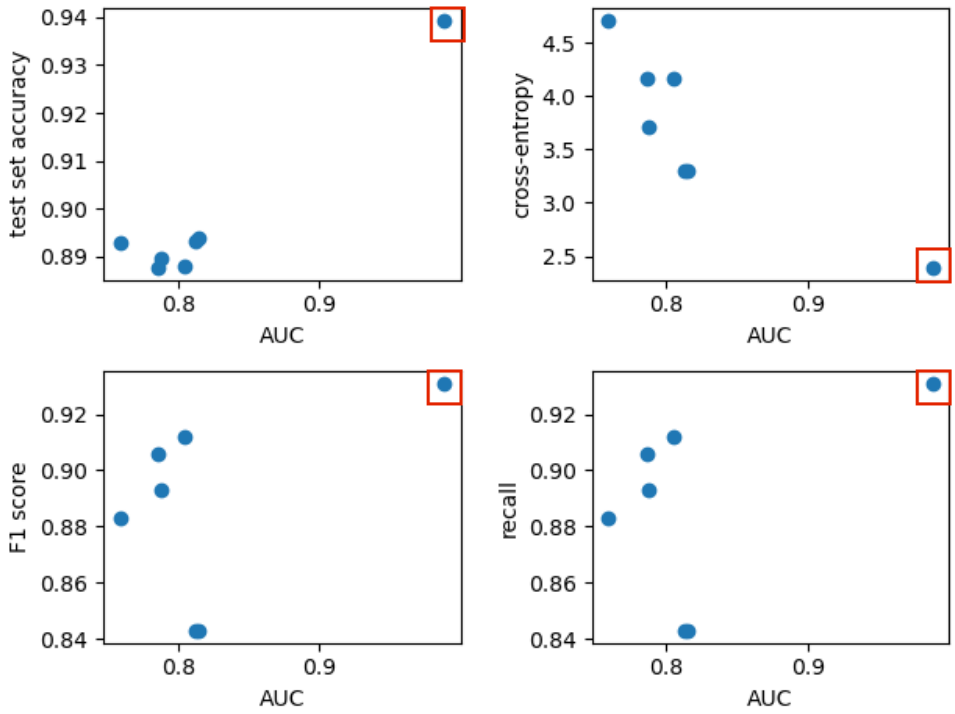


Figure 1.13: Model performance improvement with embeddings. Besides embeddings, we explored various combinations of encoders when processing the raw dataset. The raw dataset has been for a fair comparison. The data size is one million.

PCR Prediction Results We now show our experimental results on a synthetic dataset using PCR workflow. Among all technicians and reagents involved in this PCR workflow, we first set technician one has a lower accuracy of 0.7, and all other technicians and reagents have 0.95 as their accuracy or qualities. The technician's accuracy affects the volume of reagents transferred to the reaction tube. Note here that technicians' accuracy and the quality of reagents range from zero to one. A higher value means a technician or a reagent is more likely to lead a successful trial. Assigning a number for each technician and reagents not equal to the maximum number introduces variations to the system. Using this dataset, We aim to show that the proposed framework can identify the technician with lower accuracy than their peers. Furthermore, we would test if our proposed system can effectively predict outcomes with a small dataset.

The data size of the PCR workflow is one thousand. The first column of Table 1.5 shows the model performance metrics. This data size is much smaller than the prior synthetic data generated using the same PCR workflow. The typical size of the biological dataset is roughly around thousands. Our hybrid approach that uses a compact neural network to encode the available information of technicians and regents as embeddings, and uses embeddings as input to train a logistic regression-based model, is able to solve the prediction tasks with limited data scale.

We detail the process of modeling the PCR workflow and probabilistic model in Section 1.5.1 and Section 1.3.3, respectively. The model performance using a PCR dataset is shown in the first column of Table 1.5. This model is evaluated using accuracy, cross-entropy, F1 score, recall, and AUC. Using the test set, our model can predict each trial's outcome with high accuracy of 98.9%. Furthermore, a high AUC of 0.938 is obtained, meaning the model has a good discrimination ability. This high AUC is desirable, since our test set in this PCR dataset is balanced. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

CFPS Prediction Results We also evaluate the model performance using two variations of synthesized CFPS workflow. The details of modeling the CFPS workflow and probabilistic model can be found in Section 1.5.1 and Section 1.3.3, respectively. The model performance using two datasets is presented in the second and third columns of Table 1.5.

Both CFPS workflows model reagents issues, and technician transfers different volumes of reagents. Synthetic CFPS w/ 3 root causes also model the technician may retrieve the wrong test tube. The data size for both CFPS workflows is six thousand. The success rates are 11.9% and 20.22% for CFPS workflow models with two root causes and with three root causes, respectively. We randomly sample 80% of experimental logs for each dataset as the training set to estimate the latent variables. The remaining 20% of data is the test set.

As shown in table 1.5, our proposed probabilistic model can accurately predict each trial's outcome. Figure 1.14 shows confusion matrixes using the synthetic CFPS datasets that model two root causes and three root causes, respectively.

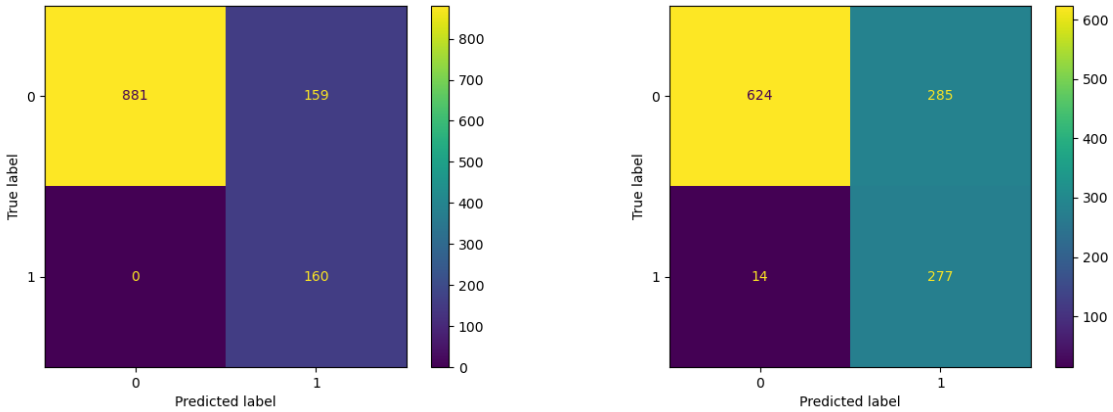
We further verify the model performance using cross-entropy, F1 score, recall, and AUC. Modeling an additional root cause, where technicians may retrieve the wrong tube, causes performance degradation in all terms of metrics. This additional root cause is modeled as a technician performance error. However, it introduces the mixture effect into the dataset. From a reagent point of view, the wrong test tube contains zero desired reagent concentration. However, we define any templates with a concentration that is lower than 5 nanoMolar as a degraded reagent. Thus it is hard to distinguish a correct but degraded template from a wrong template, purely based on data.

However, we still obtain high AUC and F1 scores in both cases. Since both datasets are highly imbalanced, we focus on the F1 score because it can still evaluate performance objectively, even with skewed class balance.

We further evaluate if the proposed probabilistic model correctly uses the input data. The first two columns of Table 1.6 show the model performance using (1) both identity and feature information and (2) only identity information, respectively. The model that uses both identities and feature information as inputs outperformed the model that uses

Table 1.5: Model performance metrics using a synthetic PCR dataset, a synthetic CFPS dataset with two root causes, and a synthetic CFPS dataset with three root causes. Our proposed probabilistic model can still accurately predict each trial's outcome, even with a small dataset.

	Synthetic PCR	**Synthetic CFPS w/ 2 root causes**	**Synthetic CFPS w/ 3 root causes**
data size	1000 (54.8% succeed)	6000 (11.9% succeed)	6000(20.22% succeed)
test set accuracy	98.9%	86.75%	75.08%
cross-entropy	0.207	4.576	8.606
F1 score	0.994	0.868	0.751
recall	0.994	0.868	0.751
AUC	0.938	0.924	0.819



(a) Using a CFPS workflow that models two root causes.

(b) Using a CFPS workflow that models three root causes.

Figure 1.14: Confusion matrix using CFPS datasets.

identity information. Feature embeddings contain valuable prior information for technicians and reagents. Therefore, feature embeddings allow us to place similar items near each other, facilitating downstream prediction tasks.

Baseline Comparison using a CFPS dataset We also compare the model performance of the proposed model with a popular baseline for prediction tasks called gradient-boost trees [40]. Gradient-boosted trees are a popular supervised learning algorithm in classifications and regression tasks. Boosting is an ensemble method that combines a few weak learners in sequential order.

As shown in the first and third columns of Table 1.6, our proposed neural network-based model outperforms this widely-adopted classification tool.

Table 1.6: Baseline comparison using a synthetic CFPS dataset. The first column shows the model performance of the proposed neural network-based model. The second column shows the result of a model that uses identity information as input. Comparing the first and second columns, we demonstrate that our proposed model uses the right amount of inputs. Finally, in the third column, we show the model performance of gradient-boosted trees, a popular baseline for classification and regression. Again, comparing the results in the first and third columns, we show that our proposed model outperformed the baseline.

	Original	**No Feature Embedding**	**Gradient-boosted Trees**
Test set accuracy	86.75%	83.58%	84.50%
Average Log Loss (cross-entropy)	4.576	5.670	4.493
F1 score	0.868	0.836	0.851
Recall	0.868	0.836	0.851
AUC	0.924	0.909	0.912

Identity Root Causes of Failed Trials We would like to identify the technician or reagent that most likely led to the failed trial for each failed trial. The details of this

inference goal are presented in Section 1.3.4. Firstly, we plug in the embeddings of each technician and reagents involved in the unsuccessful trial, and train the model in Equation (1.5) to obtain the latent variables $\boldsymbol{\theta}_j$ and $\boldsymbol{c}_j$. Then, by applying a Sigmoid transform to the multiplication of $\boldsymbol{\theta}_j$ and the neural network output, the probability of success can be calculated for each technician and reagent involved in each trial.

$$\begin{aligned} \boldsymbol{P}(\boldsymbol{y}=1) &= \phi(\boldsymbol{\theta}_j \cdot \boldsymbol{x}_j + \boldsymbol{c}_j) \\ \phi(z) &= \frac{1}{1+e^{-z}} \end{aligned} \tag{1.15}$$

Then we can rank the probability of success of each technician and reagent involved in each failed trial i in ascending order to obtain a ranking list $\boldsymbol{PR}_i$. We define $\boldsymbol{PR}_i(j)$ as the rank (or position) of the j-th probability of success of technician or reagent in trail i. The lower the rank, the higher the chance that this particular technician or reagent is likely to lead to an unsuccessful trial. The predicted root causes for each failed trial are technicians and reagents with a lower probability of success, denoted as $\boldsymbol{PR}_i(0)$. Defined $\boldsymbol{RR}_i$ as the root cause for each failed trial i, such as the involvement of degraded reagents or technicians with low accuracy. Then we can compare the real root cause $\boldsymbol{RR}_i$, which is the ground truth, with the predicted root cause $\boldsymbol{PR}_i(0)$.

A sample failed trial is shown in Figure 1.15. Each column is a potential root cause for this PCR workflow. The first row presents real root causes. Each real root cause is marked with "x". Predicted root causes are listed in the second row. Similar to real root causes, all predicted root causes are marked with "x". In this example, the real root causes are reverse primer, master mix, and technician performance. The predicted root causes are reverse primer, master mix, and template. If the predicted root cause matches the real root cause, we highlight them using a gold box. Otherwise, we mark unmatched root causes with a blue box. This trail is counted as a failed root cause identification since not all predicted root causes match real root causes.

If the real root causes and predicted root causes are matched for a failed trial, we consider this root cause identification is successful. The accuracy of root cause identifications for a

	Forward primer	Reverse primer	Master mix	Template	Technician
Real root causes	-	x	x	-	x
Predicted root causes	-	x	x	x	-

Figure 1.15: A sample failed trial. Each column is a potential root cause for this PCR workflow. The first and second rows present real and predicted root causes, respectively. We use a gold box to indicate if predicted root causes match predicted root causes. The blue box marks the unmatched predictions. This trail is counted as a failed root cause identification since not all predicted root causes match real root causes.

synthetic CFPS that models two root causes, a synthetic CFPS that models three root causes, and a synthetic PCR is 92.81%, 80.26%, and 85.5%, respectively.

Table 1.7: The accuracy of the root cause identifications for the synthetic datasets. If the predicted root causes match the real root causes, we consider this root cause identification is successful.

	Synthetic CFPS w/ 2 root causes	**Synthetic CFPS w/ 3 root causes**	**Synthetic PCR**
Accuracy of successful root causes identifications	92.81%	80.26%	85.5%

Identify Degraded Reagents and Technicians with Low Accuracy The second inference goal, detailed in Section 1.3.4, is to identify technicians with low accuracy or degraded reagents due to retraining or replacement. More specifically, we would like to rank techni-

cians and reagents based on how likely they lead to failed trials. Since our method produces a success rate for each technician and reagent, we can rank predictions from highest to lowest and compare them with the ground truth values.

For the synthetic PCR dataset, technician zero has a lower accuracy of 0.7, and all other technicians and reagents have the same accuracy of 0.95. Therefore, our proposed statistical method can identify technician zero with the lowest accuracy.

Two variations of CFPS workflow have been modeled using OOPS. The first dataset is collected from a CFPS workflow that models degraded reagents, and the technician transfers different volumes of reagents. We use the term "CFPS that models two root causes" for the first dataset. The second dataset is called "CFPS that models three root causes". Besides the exact root causes modeled in the first dataset, we also model that the technician may retrieve the wrong test tube.

For CFPS that models two root causes, we demonstrate the ranking of reagent and technician by success rates in Table 1.8 and Table 1.9, respectively. The first row shows the ranking based on real root causes. In the second row, we obtain the ranking of the reagent or technician by using the proposed model. The third row shows the model's results that use identity information as input. Comparing two different types of encoding methods, we found that using identity and feature information achieves the best performance when identifying the degraded reagents and technicians with low accuracy.

We compare our results with the ranking obtained from raw counting to demonstrate that the proposed statistical method to identify degraded reagents and technicians with low accuracy is effective. For each technician and regent, we count the number of successful trials that the technician or reagent is involved in and calculate their success rates. These success rates obtained from raw counting are used as baselines. In Table 1.10 and Table 1.11 we show the baseline comparison using the CFPS that models two root causes. The failure outcome in raw data is a mixture of a technician performance issue and a reagent issue. Thus our proposed method outperformed the raw counting baseline. As shown in Table 1.12 and table 1.13, the same trend is observed using the CFPS that models three root causes.

Table 1.8: Rank reagents using a CFPS workflow that models degraded reagents, and technicians transfer different volumes of reagents. Compare two cases that one uses feature and identity information as input, and the other case only uses identity information. The ranking results show that using identity and feature information is the most efficient encoding method.

Reagent: ranking (success rate)	**0**	**1**	**2**	**3**
Real root causes ranking	0	1	2	3
Probabilistic model	0 (0.02%)	1 (0.04%)	2 (0.04%)	3 (80.86%)
Without feature embeddings	3 (0.02%)	0 (0.04%)	1 (0.14%)	2 (49.68%)

Table 1.9: Rank technicians using a CFPS workflow that models degraded reagents, and technicians transfer different volumes of reagents. Compare two cases that one uses feature and identity information as input, and the other case only uses identity information. The ranking results show that using identity and feature information is the most efficient encoding method.

Technician: ranking (success rate)	**0**	**1**	**2**	**3**
Real root causes ranking	2	3	1	0
Probabilistic model	2 (17.62%)	3 (19.7%)	0 (20.88%)	1 (21.82%)
Without feature embeddings	0 (3.8%)	3 (6.18%)	1 (19.18%)	2 (20.14%)

Table 1.10: Baseline comparison for technicians using synthetic CPFS data that models degraded reagents and technicians transfer different volumes of reagents. Our proposed statistical methods outperformed the ranking obtained from raw counting.

Reagent: ranking (success rate)	**0**	**1**	**2**	**3**
Real root causes ranking	0	1	2	3
Probabilistic model	0 (0.02%)	1 (0.04%)	2 (0.04%)	3 (80.86%)
Raw counting	1 (5.88%)	3 (23.53%)	2 (29.41%)	0 (41.18%)

Table 1.11: Baseline comparison for reagents using synthetic CPFS data that models degraded reagents and technicians transfer different volumes of reagents. Our proposed statistical methods outperformed the ranking obtained from raw counting.

Technician: ranking (success rate)	**0**	**1**	**2**	**3**
Real root causes ranking	2	3	1	0
Probabilistic model	0 (17.62%)	2 (19.7%)	3 (20.88%)	1 (21.82%)
Raw counting	0 (0.06%)	3 (0.18%)	2 (28.94%)	1 (30.54%)

Table 1.12: Baseline comparison for reagents using synthetic CPFS data that models degraded reagents, technicians transfer different volumes of reagents, and technicians may retrieve wrong test tubes. Our proposed statistical methods outperformed the ranking obtained from raw counting.

Reagent: ranking (success rate)	**0**	**1**	**2**	**3**
Real root causes ranking	3	1	2	0
Proposed model	0 (14.78%)	2 (21.78%)	3 (29%)	1 (47.5%)
Raw counting	3 (0%)	2 (28.72%)	1 (33.11%)	0 (38.18%)

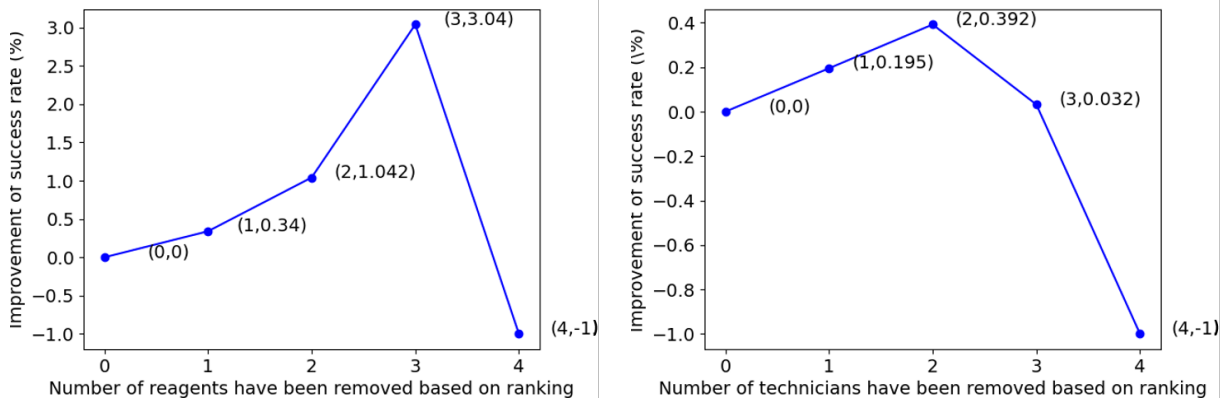
Table 1.13: Baseline comparison for technicians using synthetic CPFS data that models degraded reagents, technicians transfer different volumes of reagents, and technicians may retrieve wrong test tubes. Our proposed statistical methods outperformed the ranking obtained from raw counting.

Technician: ranking (success rate)	**0**	**1**	**2**	**3**
Real root causes ranking	0	1	3	2
Proposed model (item 1)	0 (28.28%)	2 (29.08%)	3 (29.12%)	1 (29.5%)
Proposed model (item 2)	3 (28.24%)	0 (28.6%)	1 (28.86%)	2 (29.02%)
Proposed model (average)	0 (28.44%)	3 (28.68%)	2 (29.05%)	1 (29.18%)
Raw counting	2 (4.73%)	3 (13.85%)	1 (28.04%)	0 (53.38%)

Ranking with Deletion Lastly, we would like to show if the proposed numerical method works. We discovered a mechanism termed ranking with deletion, which suggests that the suggested framework is functional. The purpose is to demonstrate that the total success rate should increase while removing trials related to technicians or reagents designated for retraining or replacement.

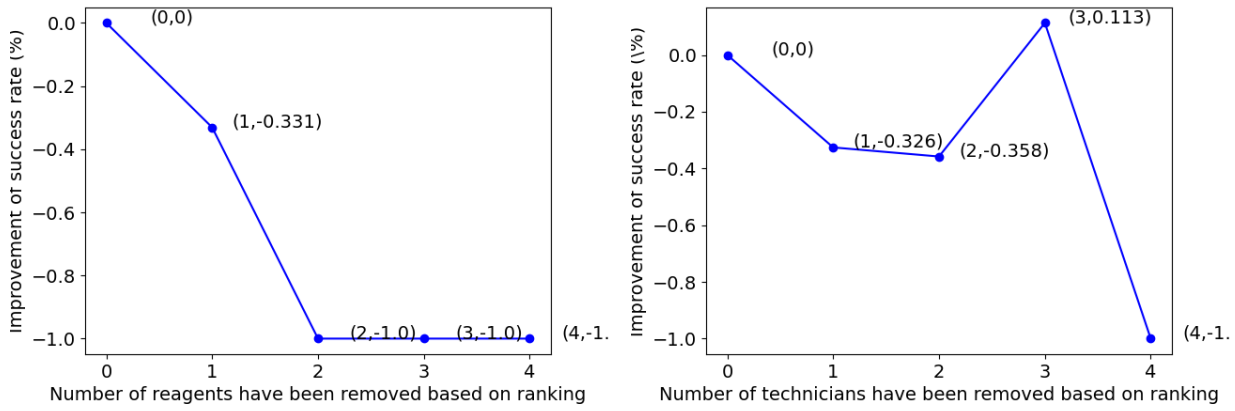
For technician A, define N_A as the total number of trials technician A has performed. M_A is the number of all trials technician A has performed and succeeded in. Given N_A and M_A, a numeric score $\frac{M_A}{N_A}$ associated with technician A can be calculated. We repeat the same procedure to find the numerical scores for each technician and regent involved in the test set. These technicians and regents are ranked based on their associated numeric scores. The lowest K technicians and regents needed to be retrained or replaced for this workflow. Then we removed all trials associated with technicians or technicians that needed to be retrained or replaced, and observed the change in the percentage of successful trials.

In this sample CFPS workflow, four technicians and four reagents are involved. Figure 1.16 shows the result of ranking with deletions with a CFPS model that models degraded reagents and technicians transfer different volumes of reagents. In this case, the ranking of reagents by success rate is $0, 1, 2, 3$. We start with deleting all trials that use reagent zero, and observe the change in success rate. We repeat the same procedure with the order of reagent zero, reagent one, reagent two, and reagent three. The x-axis of Figure 1.16(a) and 1.16(b) shows the number of reagents and technicians that have been deleted, and the y-axis shows the improvement in success rate. We observe that the success rate increases with deleting the reagents and technicians with low success rates. Similar trends have been observed with the result collected from the CPFS workflow that models three potential root causes: (1) degraded reagents, (2) technicians transferring different volumes of reagents, and (3) technicians may retrieve the wrong test tubes.



(a) reagents (b) technicians

Figure 1.16: Baseline Comparison: ranking with deletions using a CFPS workflow that models two root causes: degraded reagents and technicians transferring different volumes of reagents. By deleting the reagents and technicians with low success rates, we observe the overall success rates increase.



(a) reagents

(b) technicians

Figure 1.17: Baseline Comparison: ranking with deletions using a CFPS workflow that models three root causes: degraded reagents, technicians transferring different volumes of reagents, and technicians may retrieve the wrong test tubes. By deleting the reagents and technicians with low success rates, we observe the overall success rates increase.

1.5.2 Real-World Datasets

Prediction Results We also conducted experiments to verify that our proposed method is capable of root causes analysis in real-world datasets supplied by Aquarium [14] software running at the University of Washington BIOFAB. The biological workflows in the real-world dataset are PCR, Gibson assembly, and yeast strain construction. Gibson assembly allows DNA to be assembled seamlessly. Yeast strain construction takes genetic material and transforms it into a given strain of competent yeast cells. These real datasets are supplied by Aquarium [14] software running at the University of Washington BIOFAB. We demonstrate how to collect and process these real datasets in Appendix A.

The original experimental logs in Aquarium [14] software include operations batched within the job, the identity of submitting users, technicians involved in each operation, the precise timing of each workflow operation, the data acquisition from equipment, the inventory management, and the error logs.

The original datasets are described in the first row of Table 1.14. We split the dataset into an 80% training set and a 20% test set.

Let input matrix X be technicians and reagents feature and identity information for each workflow. Identify information is a randomly generated 3×1 vector for each technician and reagent. The feature information for each technician includes: the number of technicians that attempted a specific operation, the number of trials that technicians completed, the number of trials that the technicians aborted, and the number of successful and failed trials that the technician has performed. For reagents, the feature information contains age.

The details of constructing a probabilistic model are presented in Section 1.3.3. Table 1.14 presents the model performance for each fitted model. Using the test dataset, the fitted root cause analysis model has been evaluated by accuracy, cross-entropy, F1 score, recall, and area under the curve (AUC). The results in Table 1.14 shows that the models generalize well. Furthermore, with high AUCs and high F1 scores, we demonstrate that our probabilistic model is able to predict classes correctly, even on imbalanced datasets.

Table 1.14: The model performance for each fitted root cause analysis model of each workflow.

	PCR	**Gibson assembly**	**yeast strain construction**
data size	2400 (91% succeed)	1650 (42% succeed)	750 (25% succeed)
accuracy	97.71%	99.7%	88.31%
cross-entropy	0.792	0.209	5.607
F1 score	0.977	0.994	0.838
recall	0.977	0.994	0.838
AUC	0.892	0.995	0.588

Identity Root Causes of Failed Trials We present the inference process, detailed in Section 1.3.4, using the yeast strain construction dataset. In this example, we have twelve operations and one hundred and twenty-four predictor variables. For each operation j, we calculate $\boldsymbol{P}(\boldsymbol{E}_j)$, which is the Sigmoid transformation of multiplications of $\boldsymbol{\theta}_j$ and the output of the neural network.

Table 1.15 shows an example of root cause identification using a trial of yeast strain construction. The ranking of variabilities is based on the values of $\boldsymbol{P}(\boldsymbol{E}_j)$. Note, the higher $\boldsymbol{P}(\boldsymbol{E}_{i,j})$ is, the higher chance we should retrain the technician or replace the reagent involved in operation j. -1 indicating the current operation is skipped.

Table 1.15: The proposed system would perform the inference process to provide the ranked list of variabilities ordered by the computed probabilities of each of the potential root causes. To calculate these failure-associated probabilities $\boldsymbol{P}(\boldsymbol{E}_j)$, we take the sigmoid transformation of multiplications of $\boldsymbol{\theta}_j$ and the neural network's output for each operation j. The higher $\boldsymbol{P}(\boldsymbol{E}_{i,j})$ is, the higher the chance we should retrain the technician or replace the reagent involved in operation j. -1 indicating the current operation is skipped.

operation	10	5	6	9	3	2	11	1	4	0
$\boldsymbol{P}(\boldsymbol{E}_j)$	0.57	0.52	0.52	0.48	0.47	0.44	0.42	0.36	0.27	-1

Identify Degraded Reagents and Technicians with Low Accuracy Ranking with deletion is applied to real datasets to verify if the proposed numerical method is functional. The same procedure is detailed in Section 1.5.1 using synthetic datasets. Since the real workflow usually involves a large amount technicians and reagents, we remove technicians by percentages. Note that reagents are hardly reused in our real datasets. Therefore, we only demonstrate the ranking of deletions results with technicians.

All technicians are ranked based on accuracy using the numerical method proposed in Section 1.3.4. The lowest $k\%$ of technicians require retraining. We conduct studies to show the effects of different values of K. Figure 1.18 shows that compared with the original test set, dropping all trials associated with $K\%$ of technicians who need to be retrained improves the percentage of successful trials, using Gibson assembly data. We further observe that increasing the value of K increases the percentage of successful trials. Due to the limited data size of the test set, we found the improvement of the percentage of successful trials starts to decrease when $K\% \geq 15\%$.

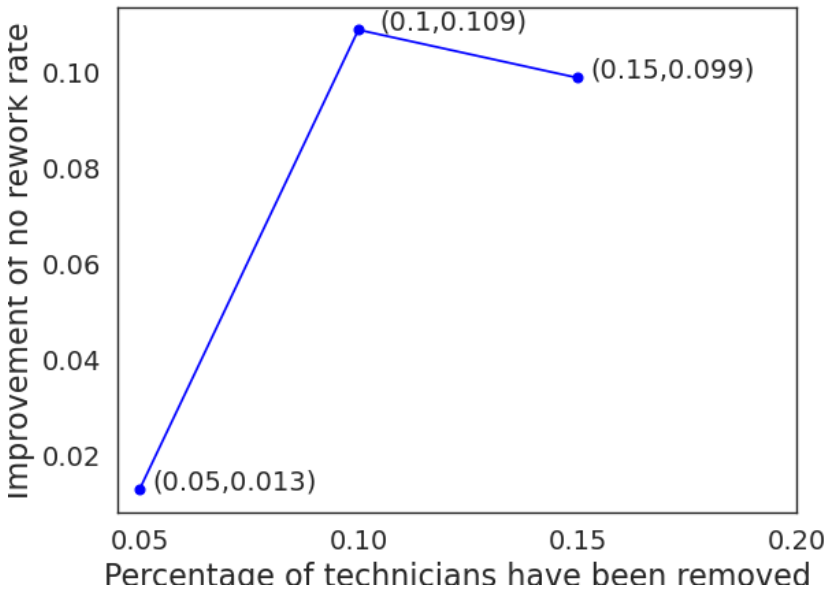


Figure 1.18: We drop all trials associated with $K\%$ technicians who needed to be retrained based on their numerical scores. We observe that the percentage of successful trials increases as the value of K increases. When $K\% \geq 15\%$, the improvement of the percentage of successful trials starts to decrease, this decrement is due to the limited data size of the test data.

1.6 Future Works

We have successfully developed a framework that enables accurate and efficient root cause analysis for biological workflows. Certain future work is required to deploy the proposed method in an industrial setting, such as Ginkgo Bioworks. As shown in Figure 1.19, deploying this framework is a multi-stage procedure and requires collaboration between a biologist who knows the workflows and machine learning engineers. First, the biologist would formalize the biological workflow and provide a mathematical model to model the reactions of the workflow. The machine learning engineer then models the biological model using OOPS. The next step is collecting data from the biologist's wet-lab experiment. OOPS generates the synthetic dataset. The framework's usefulness would be verified using real and synthetic datasets.

A user interface would allow lab managers to access the system quickly needed to be developed by future graduate students in the UW Biofab.

1. A planning system. Lab managers can generate trials of workflows, estimate costs, and duration, and forecast the outcome of each trial.

2. A management system that displays the status of each operation of the workflow. Furthermore, the management system would report the root causes whenever a failed trial is identified.

3. A retraining and replacement recommendation system. The system flags that technicians and reagents are due for retraining and replacement.

4. A system for logging operations details and recording all equipment-generated data. The obtained data is utilized to enhance the suggested framework further.

Modeling and remodeling a new biological workflow requires one biologist and one machine learning engineer to work for about one month. The user interface development would

require two months for two biologists and five machine learning engineers. Even though deploying the proposed framework would take some time, it would save a considerable amount of money and time in the long run. For example, a protein stability assay currently running in the UW BIOFAB costs around ten thousand dollars and takes a week to complete a trial. Suppose we can improve 10% the success rate for this workflow. Twenty-five thousand dollars per year, on average, can be saved. Even worse, rescheduling a failed trial would cause delays in various weeks. Since the following week's schedule is usually fixed, we need to wait a few weeks for reworking.

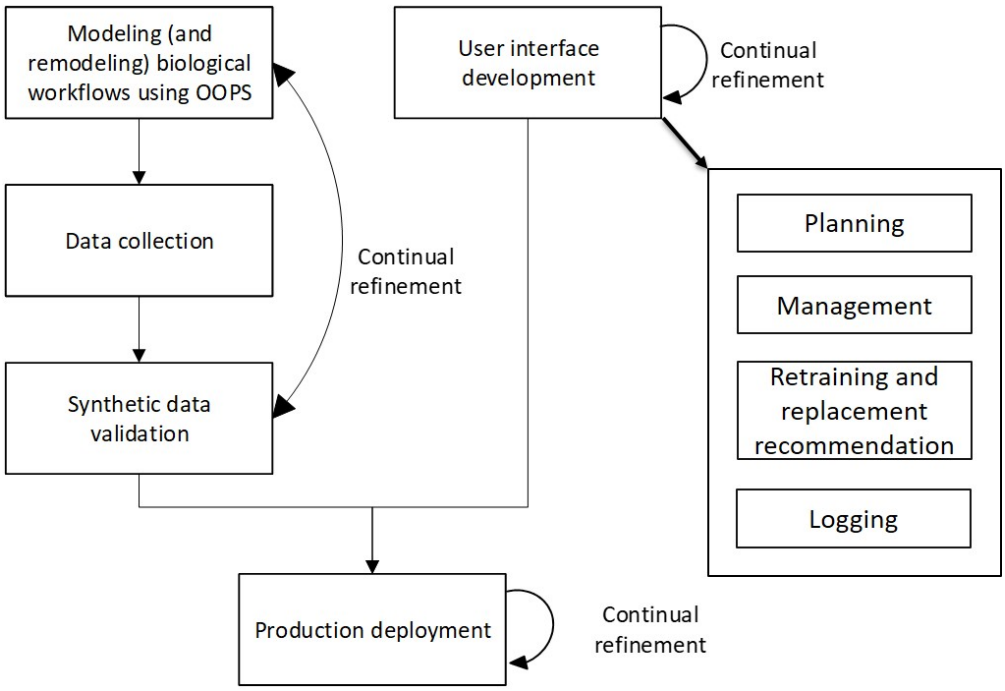


Figure 1.19: Stages of deploying the proposed framework in an industrial setting.

Chapter 2

EFFECTIVE USING REMOTE LAB IN PROMOTING SIMULATION AND VERIFICATION TOOLS

2.1 Introduction

Traditionally, universities teach Field Programmable Gate Arrays (FPGAs) courses using on-campus labs, which have lab benches, PC and workstations, and other supporting equipment. In order to complete assignments, students need to be physically present in the labs to interact with these pieces of equipment. However, this approach was not feasible during the Covid-19 pandemic, and necessitated a new teaching approach based on using global network of remote laboratories. Research shows that remote laboratory facilities prepare students to design engineering systems effectively [54, 55].

While remote laboratories have been used in teaching physics, biology, and other subjects [56, 57, 58], the majority of them are stand-alone systems that cannot be shared with different facilities. In the field of engineering education, VISIR designs a shareable system to provide seamless access for students from various universities [59]. It allows students to explore electronics, sound, and vibration or propagate radio signals through an online platform. Even though previous work demonstrates how applicable the educational remote laboratory facilities are, the state-of-the-art applications lack the ability to teach the student how to use General Purpose Input Output (GPIO) ports to communicate with the FPGA boards.

FPGAs have been widely used in remote laboratories for many years, in different initiatives at different universities [60, 61, 62, 63]. Building on the work of Mayoz *et al.* cite9163773, we added a new layer to provide an easy-to-use solution for students to work with the GPIO ports of the FPGA.

We introduced a feature that uses a virtual breadboard where students can use GPIO ports to interface with real FPGA boards that are accessed remotely through a global network called LabsLand [64, 65]. This remote FPGA laboratory has been used to teach a junior-level course in digital design at the University of Washington during the autumn quarter of 2020. We presented our experience transforming a lab that traditionally used physical breadboards and FPGA boards into an online modality. We demonstrated how the lab used to be conducted pre-Covid-19 and how it has been restructured to fit online instruction.

2.2 Literature Review

The purpose of remote laboratory facilities is to remotely operate and manage specialized experimental facilities in order to teach students knowledge in many engineering domains. Traditionally, online education is not favored in engineering courses requiring students to be physically present in the laboratory, since hands-on physical experiences play a central role in engineering education [66, 67]. In addition, hands-on physical experiments help students achieve the expected learning outcome, promote reasoning skills, and professional preparation for the workforce [68, 69].

With computer technology advancements, people have started to introduce multimedia to facilitate engineering education. Examples of multimedia include interactive on-screen experiments procures, virtual simulations, automated data acquisition, and quantitative analysis of experiments. These computer-based technologies redefined the "hands-on" experiences. The covid-19 pandemic further simulates the widespread proliferation of remote laboratories. Remote laboratories have gained popularity, particularly during the Covid-19 pandemic, due to their accessibility, ease of use, and affordability. As a result, various initiatives for creating remote and virtual laboratory systems have been put forth.

Speech-based virtual assistants have been applied to various application areas. Callaghan *et al.* [70] investigated how to use virtual reality, the Internet of Things (IoT), and voice-controlled virtual assistants to guide students through their experiments, provide supplemental teaching resources as needed, and access, control, and configure instruments and hardware while providing formative and summative feedback. Ak *et al.* [71] presented the design and implementation of the remote laboratory and learning management system. Zine *et al.* [72] provided a brief overview of online laboratories and suggested a simple approach to remote laboratories. Mohammed *et al.* [73] presented a method for improving the remote control of educational laboratory experiments within the electrical engineering sector. Monzo *et al.* [74] presented the Remote Laboratory at the Open University of Catalonia (RLAB-UOC) that allows engineering students studying online to conduct practical experiments

using advanced electronic and communication equipment anywhere and anytime. Furthermore, this work presented the case of a blended learning approach adopted in the remote teaching of electrical power engineering by Arefi *et al.* in 2021. During a new Manufacturing Engineering program at the University of British Columbia, Keulen *et al.* [75] presented and reflect on their experience of remote teaching three hands-on laboratory courses. Todos *et al.* [76] described how to conduct laboratory work under the strict conditions of a pandemic. A real-time remote control engineering teaching lab was demonstrated at Oregon State University's College of Engineering [77]. In a remote laboratory, students are able to access physical experiments at a cost-effective and flexible rate, but at the same time, an added layer of the user interface was introduced. Miele *et al.* [78] described a computer-based remote laboratory component for one of these courses at Rensselaer Polytechnic Institute, including some preliminary evaluations. Marquez *et al.* [79] proposed augmented remote laboratories (ARLs) as a replacement for virtual labs and remote environments. Azad *et al.* [80] discussed the uses of remote laboratories, pedagogy, architecture, and future trends in evaluating remote laboratories. Tawfik *et al.* [81] described cutting-edge remote laboratories for industrial electronics applications. Herrera *et al.* [82] described a remote photovoltaic power lab. Remote laboratories add the value of using real hardware in a remote location, demonstrating to students the additional issues that arise when using real equipment. These pedagogical advantages are compelling if the laboratories are designed to be used in an Interactive Engagement approach [83]. Cvjetkovic *et al.* [84] discussed some remote physics and engineering experiments using Arduino.

However, the majority of them are stand-alone systems that cannot be shared with different facilities. This has led to the recent widespread growth of remote laboratories across all fields of industrial electronics at numerous universities and institutions.

A shared architecture was used in the iLab to develop and deploy remote laboratories, consisting of lab clients, service broker middleware, and lab servers [85]. Several approaches were utilized to give online learners hands-on educational experiences, including remote laboratories, simulation software, and virtual labs, which offered a more structured setting

with planned asynchronous access to physical resources. Lahoud *et al.* [86] explored how these approaches might be applied from the learner's viewpoint to improve the efficiency of online instruction, uphold student satisfaction, and maintain the same standards and outcomes as traditional classroom instruction . Additionally, several educational strategies used by institutions that use Weblab were discussed [87]. A remote lab system that enabled remote groups to access a shared PR2 is described by Pitzer *et al.* [88]. However, this could be quite constructive and rigid.

The board's remote experimentation system is being created so that students can modify and reconfigure experiments using remote node-to-node cable connections [89]. Orduna *et al.* [90] describe and discuss a widely dispersed remote laboratory (VISIR, presented in six universities in Europe plus one in India) shared by three institutions (two universities plus one high school). Kumar *et al.* [91] investigated how real-time resource sharing made possible by remotely operated biotechnology labs can enhance student learning. Sáenz *et al.* [92] introduced an open course in the University Network of Interactive Laboratories, which provides various open-access virtual and remote laboratories on automatic control. Alavi [93] described a remote lab that gives students in online classes an immersive learning endeavor. The difficulties of assuring dependable instrument operation and high-quality data output while facility staff and outside service provider on-site presence was severely constrained were discussed by Gravano *et al.* [94].

2.3 Background

The laboratory experiment highlighted in this work is one out of five other labs students are expected to complete for the course. This lab experiment is a refresher of finite state machines and introduces students to the onboard general-purpose input/output (GPIO). The students achieve this goal while designing a parking lot occupancy counter, which consists of two pairs of photosensors monitoring the activity of cars. For educational purposes, students use switches to mimic the output signal from the sensors. The signals are also displayed on two LEDs to make the signals visible while introducing more circuit components to the laboratory. This gives students opportunities to practice implementing breadboard circuits and interfacing GPIOs, which are essential skills for hardware design. Traditionally, in the on-campus laboratory environment, each student receives a breadboard assembly in conjunction with a DE1-SoC FPGA, which is shown in Fig. 2.2. Students will implement three switches and two LEDs on the breadboard for this laboratory experiment. An example of breadboard circuits is shown in Fig. 2.1.

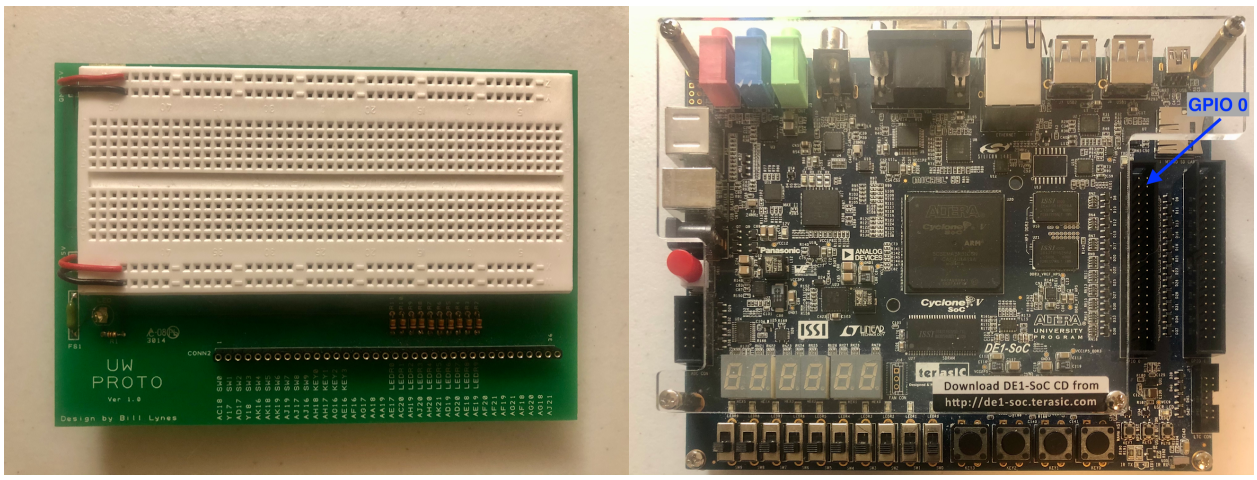


Figure 2.1: The customized breadboard module (left) and the DE1-SoC FPGA (right). The breadboard module is connected to the FPGA board via the GPIO 0 header.

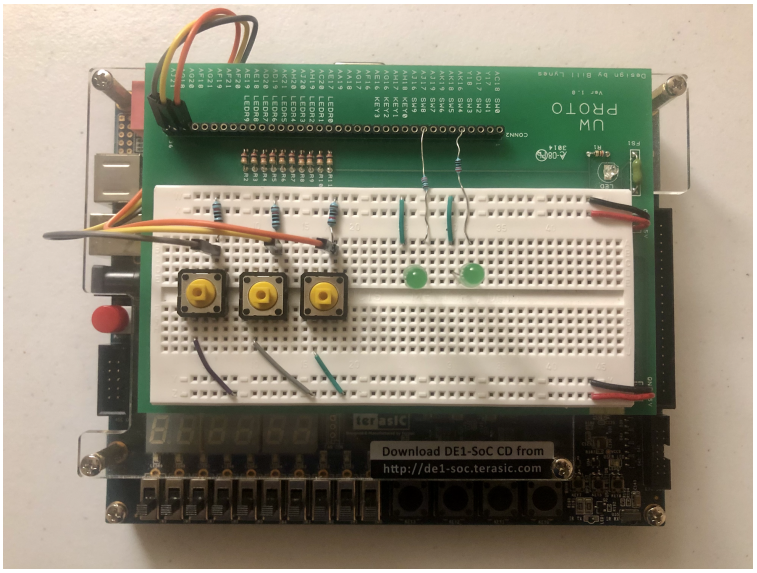


Figure 2.2: The breadboard circuits that are required by the laboratory experiment.

However, manufacturing these breadboard modules for large classes is time-consuming, and most importantly, the physical laboratory kit is not accessible in the remote learning environment. Therefore, a remote laboratory solution that does not rely on hardware distribution is needed for this particular laboratory experiment.

2.4 Technical Solution

2.4.1 User Perspective

To provide an uncompromising learning experience to students, we added a virtual breadboard interface to the remote laboratory facility to simulate hands-on circuit building, providing students the opportunity to interface GPIOs in their SystemVerilog designs.

Design Decisions for the Breadboard User Interface. A breadboard configurator is added to the user interface switch to simulate the hands-on breadboarding experience. Students have access to several circuit components in the configurator, including three switches and two LEDs that are placed onto the virtual breadboard. In addition, we have a GPIO header that is internally connected to the DE1-SoC's JP1 in the backend of the system. Next to the breadboard's implementation area, we provide a pinout diagram for DE1-SoC's GPIO 0 (JP1) to help students understand the relationship between pin names and pin numbers and build the circuit. Fig. 2.3 depicts the default state of the virtual breadboard.

Students will connect each circuit component to an appropriate GPIO pin for this laboratory experiment. To start with, students need to figure out which pin to use by referring to the pinout diagram to the left. As the pin names (for example, $GPIO\_0[20]$) are deliberately not shown on the actual GPIO header, students are forced to use the relationship between pin names and pin numbers, which ensures that all the essential steps for working with the actual boards are taken on the web interface as well. Then, students click on a wire color in the wire selector to choose a wire. Finally, they can use their mouse as a pen for drawing wires and connecting the GPIO header and the breadboard. All the connections on the breadboard will be automatically saved to emulate the real-life experience.

Students can then proceed to the web interface of the remote laboratory in the form of a code editor interface to continue implementing the constructed virtual breadboard circuit. Once they synthesize and upload the project to FPGA, the user interface shown in Fig. 2.4 is presented to the students with all the previous changes made to the breadboard preserved. To verify their design, the students can toggle the switches and observe the LEDs' states

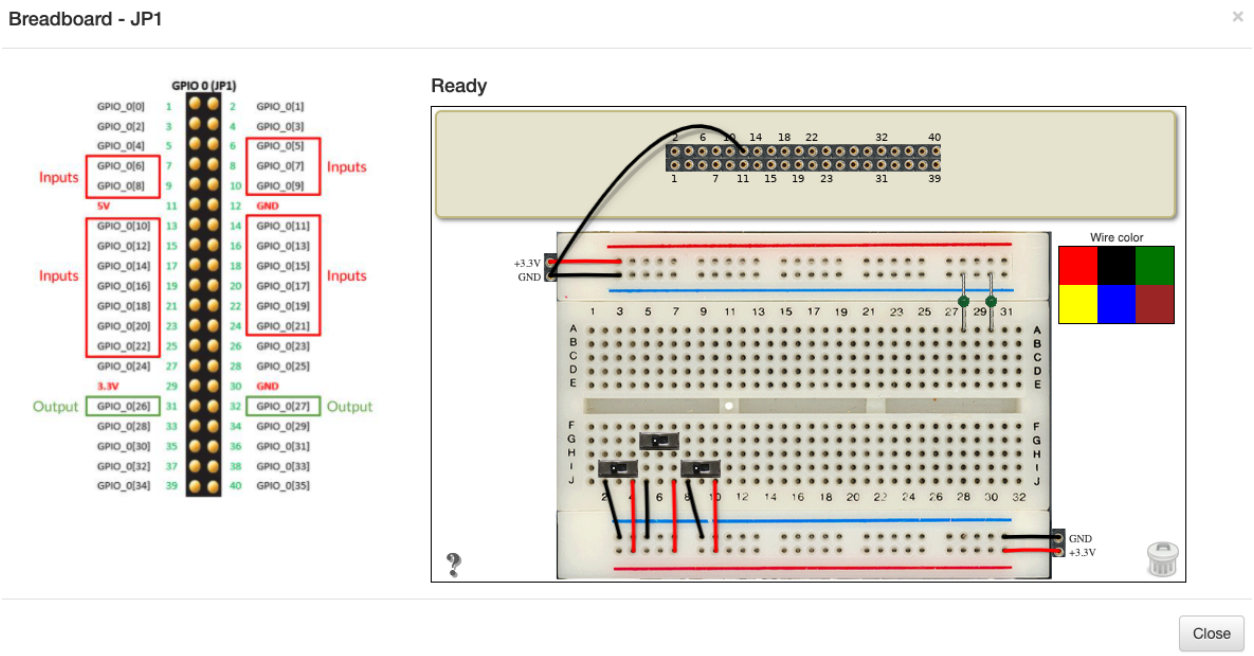


Figure 2.3: The breadboard configurator on the IDE of the remote laboratory.

and other outputs on the DE1-SoC board, such as the HEX display. A tutorial specifically for the breadboard user interface is provided to the students to help them get familiar with the workflow.

This workflow provides several benefits over other possible implementations. Firstly, it closely emulates the process of building a breadboard circuit in a physical laboratory setting, as it allows students to build the circuit before or in parallel with writing code. Secondly, by always saving the state of the breadboard for each student, it simulates the irreversible and nonvolatile nature of working with physical hardware. Thirdly, while providing unlimited time for building breadboard circuits, it also reduces the test time each student spends on the limited number of the FPGA hardware, reducing queueing time and providing a better learning experience for more students.

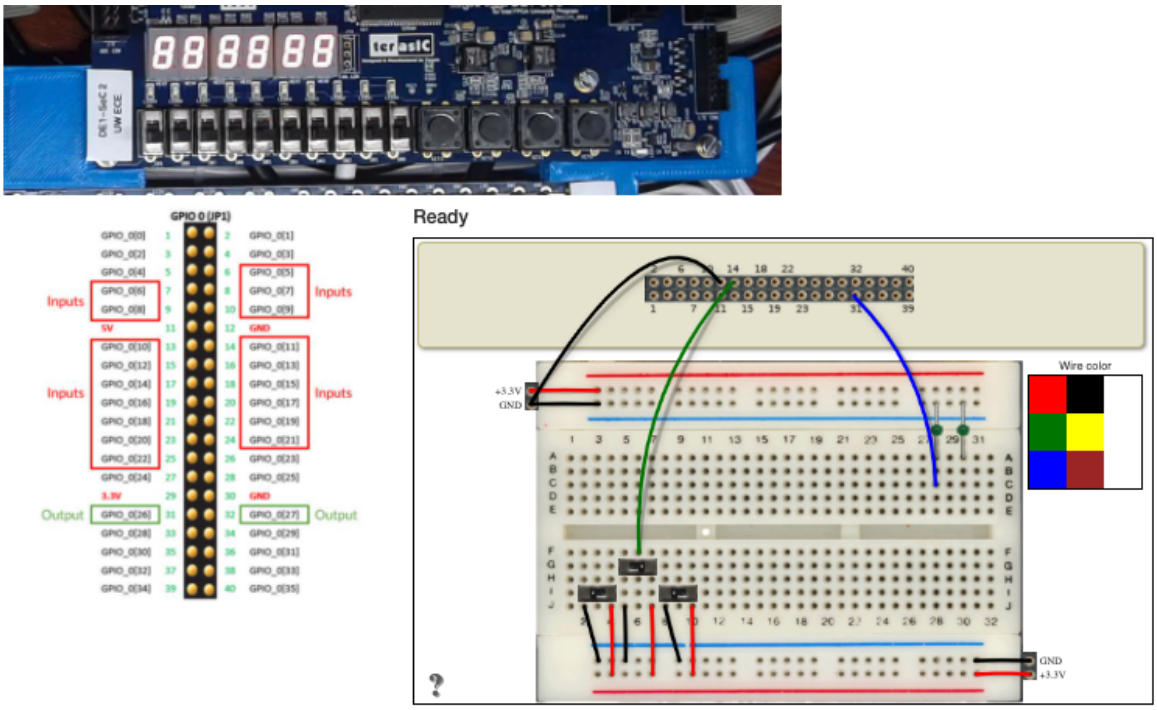


Figure 2.4: The breadboard user interface for testing FPGA programs.

2.4.2 Technical Perspective

The Open Source VISIR client has been used as an external library for the breadboard user interface, particularly the breadboard component. On top of the breadboard component, a JavaScript code is added to check if certain user interface points are connected to the GPIOs, switches, and LEDs on the breadboard. The system automatically detects faulty connections to help students connect what they need to connect to the particular GPIO.

The wiring phase is typically done in the code editor [65]. Once the breadboard is wired, whenever students use the FPGA, they have the same circuit built before, and they can still modify it once they are connected to the FPGA itself. Internally, each change of the wires or switches state is translated into a call to the server to send specific signals to the GPIOs. Once we detect any wire-changing calls, the web browser will turn on and off the LEDs accordingly.

2.5 The FPGA with Breadboard Laboratory Experience at the UW

2.5.1 Contextualization

Due to the logistical difficulty of shipping physical DE1-SoC lab kits to students during the Covid-19 pandemic, all students in the Design Of Digital Circuits class at the UW in Autumn 2020 were asked to use the remote laboratory along with the newly developed breadboard user interface. The remote laboratory does not require shipping any laboratory equipment and can be accessed anywhere and anytime. Before taking this class, all students are required to take a prerequisite junior-level Design of Digital Circuits and System course. Since the remote laboratory facilities just deployed for this offering. Students only have prior knowledge of using the physical DE1-SoC development board and breadboards.

2.5.2 Survey and Results

The students in the Design Of Digital Circuits And Systems class were asked to complete an anonymous online survey to indicate their degree of agreement on certain statements based on a 5-point Likert scale [95], with one being "strongly disagree" and five beings "strongly agree." Out of sixty students enrolled in the course, fifty-five completed the survey. The survey questions were designed to evaluate their experience using the remote platform with the newly developed breadboard user interface. The survey and the corresponding quantitative data for statistical analysis are given in Table 2.1.

The survey results indicate that the virtual breadboard served its purpose as an alternative to the physical breadboard. Furthermore, deliberate design decisions, such as the placement of the pinout diagram, helped students understand the topics being taught in the course. Moreover, while closely emulating the physical breadboard, the virtual breadboard is not hard to use, helping students become proficient in it quickly and thus saving them time on other tasks. However, the survey results also show that the breadboard interface has room for future improvements.

We further extend our evaluation process to industry professionals who work in engineering-

Table 2.1: The online survey, questions, and quantitative data collected from students (N = 55).

#	**Question**	**Min**	**Max**	**Mean**	**Std. Dev.**	**Var.**
1	The pinout diagram shown on the Breadboard UI was helpful	2	5	4.02	0.89	0.8
2	Connecting wires on the breadboard UI was easy	1	5	3.35	1.25	1.56
3	Testing the breadboard circuit with my code was easy	1	5	3.55	1.1	1.22
4	The breadboard interface does not need improvement	1	5	2.95	1.18	1.39
5	The breadboard interface closely emulated the real physical breadboard	1	5	3.67	1.12	1.26
6	The Breadboard UI helped me understand how to use FPGA's GPIOs	1	5	3.64	1.08	1.16
7	I was less afraid of damaging the FPGA board than when I work with circuits in the traditional lab	1	5	3.58	1.46	2.14
8	The Breadboard UI worked out without any problems	1	5	3.38	1.19	1.43

related fields. A new survey for a study on the effectiveness of simulation and experimentation tools in engineering design has been developed. In this survey, we seek the perspective of industry professionals who use modeling and simulation tools to some extent in their job duties. Industry professionals are asked to fill out the survey based on their collective work experience in the industry, whether from their current job or previous job(s). The survey questions consist of Likert scaling questions on a scale from 1 - 5 (1 = strongly disagree, 2 = disagree, 3 = neutral, 4 = agree, and 5 = strongly agree), and brief short answers questions.

The survey has been sent to three types of industry partners. The first group is companies that develop simulation and verification tools, such as Intel and Cadence Design Systems. The second group is companies that use simulation and verification tools, such as Qualcomm and Samsung. Finally, the third group is other engineering-related companies, such as Google, META, Amazon, and Apple. Thirty-five industry professionals completed the survey.

The survey starts with collecting job roles. The demographic results have been reported in figure 2.5. Forty-nine percent of participants are engineers who work in related areas. The remaining 51% of participants are managers.

Figure 2.6 shows that among thirty-five industry professionals who have filled out the survey, only two participants have not used any simulation or verification tools in their job duties. We asked industry professionals to provide examples of online experimentation environments or any setting where they are able to carry out experiments in a remote or virtual/online setting. Examples provided by participants include EDA playground, web inspector in the Chrome browser, Browserstack (a platform that allows users to access real phones), Citrix VDI, MATLAB, AWS, Google Cloud, TWCC, Xcode iPhone simulator, Labview, ORAL_RT, typhoons, Bento Notebook (developed by META), and some internal tools that allow users to access FGPA, new PCB designs, and ASIC boards and servers.

Table 2.2 presents the perspective of industry professionals who use modeling and simulation tools to some extent in their job duties. By answering this questionnaire, industry professionals are asked to rate the statements based on their prior experiences related to

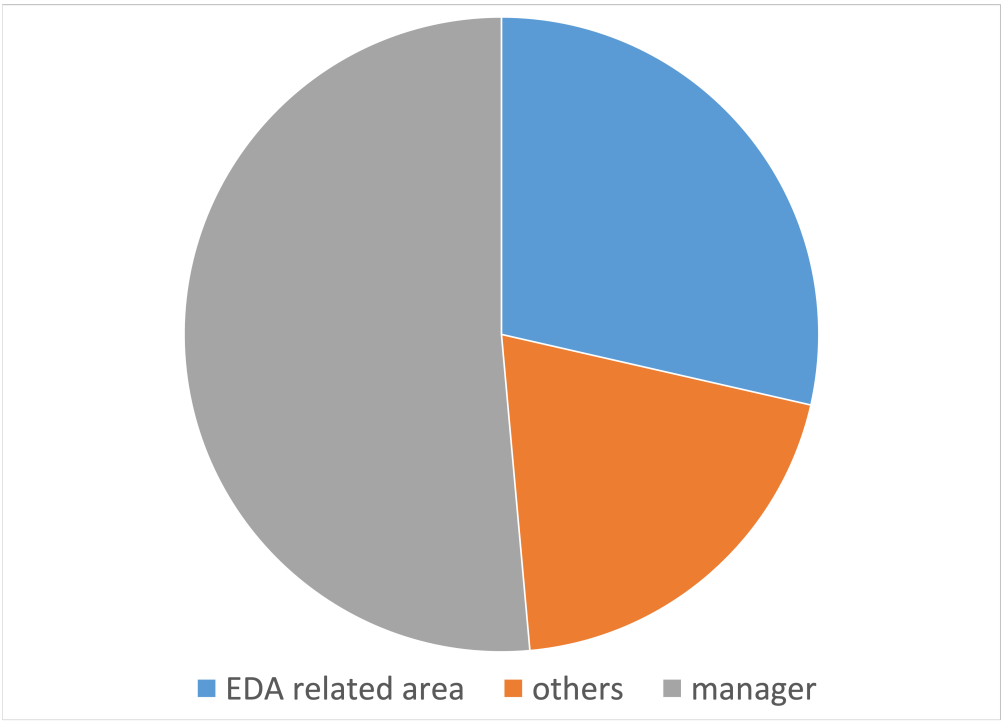


Figure 2.5: Current roles or job titles collected from industry professionals (N = 35).

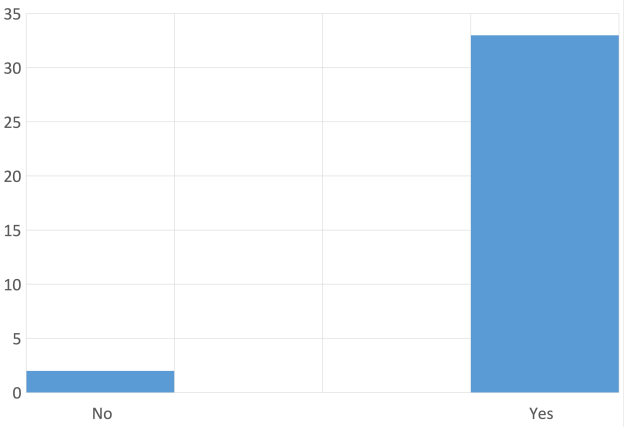


Figure 2.6: Check if industry professionals have used any simulation or verification tools in their job duties (N = 35).

simulation or verification tools. Each of these items tackles at least one objective to develop a remote laboratory for engineering education. Note that the min column is considered an outlier for this survey. Since the variance is slight, and average scores are close to each question's maximum value.

The survey results show the trend that our industry partners believe our proposed framework is well aligned with industry practices. Based on the survey results, we conclude that new employees should be well-trained in using simulation and verification tools at school. Furthermore, a remote laboratory provides users with consistent results, easy access to analyzing and interpreting data, and developing teamwork abilities. This shows that the remote laboratory is a promising tool in engineering education that aligns with industry's expectations of engineering graduates entering the workforce.

In summary, we interviewed students and industry professionals. The students' interviews focused on their experiences with the remote laboratory. The interviews with industry professionals zeroed in on their overall experience with simulation and verification tools. By comparing the results of the two surveys, we found that remote lab forces students to spend more time on simulation tools compared to traditional offline design. Therefore, students have considerably improved their skills in using simulation tools. This founding aligns with the study results of Hussein *et al.*'s work [54].

Table 2.2: The online survey, questions, and quantitative data were collected from industry professionals (N = 35).

#	**Question**	**Min**	**Max**	**Mean**	**Std. Dev.**	**Var.**
1	How much do you work with verification/simulation tools versus real hardware in your job duties?	0	100%	0.54	0.33	0.11

Table 2.2: The online survey, questions, and quantitative data were collected from industry professionals (N = 35).

#	**Question**	**Min**	**Max**	**Mean**	**Std. Dev.**	**Var.**
2	Entry level hires are expected to have solid skills in using simulation and verification tools	2	5	4.06	1.00	1.00
3	Verification of hardware design is typically done using simulation tools prior to deploying a functional solution on real hardware such as FPGAs	1	5	4.06	1.03	1.06
4	Using simulation and verification tools makes the design phase more efficient	3	5	4.49	0.66	0.43
5	Using an online experimentation environment allows users to work effectively in a team	3	5	4.11	0.87	0.75
6	Using an online experimentation environment facilitates data handling. For example, Amazon Web Services (AWS), 1010data, etc	3	5	4.37	0.69	0.48
7	Using an online experimentation environment is an adequate opportunity to connect users in a large variety of teams and let them carry out experiments	3	5	4.21	0.69	0.47

Table 2.2: The online survey, questions, and quantitative data were collected from industry professionals (N = 35).

#	**Question**	**Min**	**Max**	**Mean**	**Std. Dev.**	**Var.**
8	Using an online experimentation environment allows users to obtain consistent experiences	2	5	4.29	0.80	0.64
9	Feedback and assessment can be made readily available by using an online experimentation environment	2	5	4.09	0.83	0.69

2.6 Future Works

We plan to extend the UW remote laboratory to support the LTC connector and the ADC. The LTC connector on the DE1-SoC board contains the Serial Peripheral Interface (SPI) master, an I2C, and a GPIO interface. The communication is enabled by the LTC connector and the hard processor system (HPS) system. The real code will be passed to the ARM using the LTC connector. We are working on providing better support for ARM, for example, running bare-metal code. We plan to support Analog-to-Digital Converter (ADC) and Digital-to-Analog Converter (DAC) on the DE1-SoC in the near future. There are two proposed uses of the ADC/DAC. Firstly, we can connect a single sensor to the FPGA, which captures the analog signal. Then we initialize the ADC to read some digital values based on the reading of the sensor. Secondly, we can connect ADC/DAC to another device, for example, the Raspberry Pi. Combining ADC and LTC connectors, we are opening the doors for students to implement more exciting projects involving sensor readings. Additionally, we are analyzing options to improve the GPIO connectivity in the breadboard by providing virtual components, such as logic gates, to improve the power of the current solution.

Moreover, the current design of the virtual breadboard could be extended to allow that allows students to explore logic gates and more GPIOs. This shall allow the integration of a more sophisticated virtual breadboard with microcontrollers or FPGAs.

Despite the small sample size for the survey collected from our industry partners, the subjects we interviewed were quite diverse. Our future effort will interview more industry professionals to draw more solid conclusions regarding the importance of simulation and verification tools in the industry.

Chapter 3
CONCLUSION

Biological experiments are complex processes, and many factors can influence a biological experiment's success rate. This dissertation established some foundational work in quantitive and statistical analysis of identifying the root cause of failed biological experiments.

The availability of high-quality data is a critical consideration in designing machine learning and statistical models. As a first step, we introduced Open Operational Protocol Semantics (OOPS), a framework that enables the production of physics-based synthetic datasets for biological workflows. OOPS allows the users to control physic-based and realistic factors that impact the result of the experiment. Furthermore, the synthetic dataset with known ground-truth reasons for failed experiments allows us to validate the correctness of our proposed methods.

We then propose a statistical learning method that is specifically designed for root cause analysis in biological experiments. Two distinct features of biological experiments inspired the design of our proposed method: (1) the number of experiment trials available for learning is generally quite limited. (2) The technicians and items are involved in multiple experiment trials. Because of these two features, we design a machine learning method that is data efficient, and aim to exploit the correlation between experimental trials based on the technicians and items involved. The details of the experimental records, such as technicians and reagents involved in each operation of the trials, are encoded to a low-dimensional embedding using a neural network. With the embeddings as the input, a logistic regression-based model is trained to predict whether a particular trial would succeed, considering the involvement of reagents and technicians. We validated the efficacy of our proposed method on synthetic datasets as well as real-world datasets.

During Autumn 2020, we taught Design of Digital Circuits and Systems at the University of Washington using the remote laboratory facility. The remote laboratory facility relies on international collaboration with other universities. We introduced a novel approach during this offering by using a virtual breadboard, which allows students to interact bi-directionally with the FPGA using LEDs and switches. Traditionally, FPGA remote laboratories only support using pre-wired switches and buttons and LEDs seen on the camera. However, it does support wiring the GPIO freely. This contribution represents the first step of providing a hybrid simulated and real solution for teaching labs using GPIO in class, providing students with a certain degree of freedom compared to existing solutions. We evaluated this approach by conducting an anonymous survey of the 60 students enrolled in the course. Results show that we successfully transformed a lab experiment traditionally conducted in person into an online modality. We further evaluate the effectiveness of simulation and experimentation tools in engineering design by collecting data from industry professionals. Surveys used in our study presented students' perspectives and experiences using a remote laboratory in designing digital systems. They also presented the perspectives of a small sample of industry professionals on the importance of simulation and verification tools in engineering design. Surveys showed that the remote laboratory is a promising platform for serving the education needs of students in digital design courses, while promoting students' skills in using simulation and verification tools, skills that are perceived as valuable by industry professionals. Our future effort will assess more assignments to provide a comprehensive statistical analysis of the benefits of using remote laboratories.

BIBLIOGRAPHY

[1] S. D. Cole, K. Beabout, K. B. Turner, Z. K. Smith, V. L. Funk, S. V. Harbaugh, A. T. Liem, P. A. Roth, B. A. Geier, P. A. Emanuel, S. A. Walper, J. L. Chávez, and M. W. Lux, "Quantification of interlaboratory cell-free protein synthesis variability," ACS Synthetic Biology, vol. 8, no. 9, pp. 2080–2091, 2019. PMID: 31386355.

[2] A. A. Deshpande, A. Ramya, V. Vishweshwar, G. R. Deshpande, and A. K. Roy, "Applications of gage reproducibility & repeatability (GRR): Understanding and quantifying the effect of variations from different sources on a robust process development," Organic Process Research & Development, vol. 18, p. 1614–1621, Aug 2014.

[3] D. Gibson, L. Young, R. Chuang, J. Venter, C. Hutchison, and H. Smith, "Enzymatic assembly of dna molecules up to several hundred kilobas1," Nature methods, vol. 6, pp. 343–5, 05 2009.

[4] M. Baker, "1,500 scientists lift the lid on reproducibility," Nature, vol. 533, pp. 452–454, May 2016.

[5] M. Baker, "Irreproducible biology research costs put at $28 billion per year," Nature News, 06 2015.

[6] Y. Shu, L. Ming, F. Cheng, Z. Zhang, and J. Zhao, "Abnormal situation management: Challenges and opportunities in the big data era," Computers & Chemical Engineering, vol. 91, pp. 104–113, 2016. 12th International Symposium on Process Systems Engineering & 25th European Symposium of Computer Aided Process Engineering (PSE-2015/ESCAPE-25), 31 May - 4 June 2015, Copenhagen, Denmark.

[7] G. Weidl and A. L. Madsen, "Object oriented bayesian networks for industrial process operation," in In Proc. Workshop on Bayesian modelling, Uncertainty in AI, 2003.

[8] A. K. Samantaray and B. O. Bouamama, Model-Based Process Supervision: A Bond Graph Approach. Springer Publishing Company, Incorporated, 1st ed., 2008.

[9] V. Lapatas, M. Stefanidakis, R. Jimenez, A. Via, and M. Schneider, "Data integration in biological research: An overview," Journal of biological research (Thessalonike, Greece), vol. 22, p. 9, 09 2015.

[10] M. M. Jessop-Fabre and N. Sonnenschein, "Improving reproducibility in synthetic biology," Frontiers in Bioengineering and Biotechnology, vol. 7, p. 18, 2019.

[11] R. Meckin, "Changing infrastructural practices: Routine and reproducibility in automated interdisciplinary bioscience," Science, Technology, & Human Values, vol. 45, no. 6, pp. 1220–1241, 2020. PMID: 32904024.

[12] M. Bates, A. J. Berliner, J. Lachoff, P. R. Jaschke, and E. S. Groban, "Wet lab accelerator: A web-based application democratizing laboratory automation for synthetic biology," ACS Synthetic Biology, vol. 6, no. 1, pp. 167–171, 2017. PMID: 27529358.

[13] M. Segal, "An operating system for the biology lab," Nature, vol. 573, 09 2019.

[14] B. Keller, J. Vrana, A. Miller, G. Newman, and E. Klavins, "Aquarium: The laboratory operating system," Jan. 2019.

[15] J. Vrana, O. de Lange, Y. Yang, G. Newman, A. Saleem, A. Miller, C. Cordray, S. Halabiya, M. Parks, E. Lopez, S. Goldberg, B. Keller, D. Strickland, and E. Klavins, "Aquarium: open-source laboratory software for design, execution and data management," Synthetic Biology, 01 2021. ysab006.

[16] D. Jurafsky and J. H. Martin, Chapter 6 Vector Semantics and Embeddings, p. 1–34. Upper Saddle River, N.J.: Pearson Prentice Hall, 3 ed., 2021.

[17] J.-T. Huang, A. Sharma, S. Sun, L. Xia, D. Zhang, P. Pronin, J. Padmanabhan, G. Ottaviano, and L. Yang, "Embedding-based retrieval in facebook search," in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20, (New York, NY, USA), p. 2553–2561, Association for Computing Machinery, 2020.

[18] Y. Ma, B. M. Narayanaswamy, H. Lin, and H. Ding, "Temporal-contextual recommendation in real-time," in Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20, (New York, NY, USA), p. 2291–2299, Association for Computing Machinery, 2020.

[19] M. Grbovic and H. Cheng, "Real-time personalization using embeddings for search ranking at airbnb," in Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '18, (New York, NY, USA), p. 311–320, Association for Computing Machinery, 2018.

[20] H. Park, Y. Shiraishi, S. Imoto, and S. Miyano, "A novel adaptive penalized logistic regression for uncovering biomarker associated with anti-cancer drug sensitivity,"

IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 14, no. 4, pp. 771–782, 2017.

[21] L. Teytelman, A. Stoliartchouk, L. Kindler, and B. L. Hurwitz, "Protocols.io: Virtual communities for protocol development and discussion," PLOS Biology, vol. 14, pp. 1–6, 08 2016.

[22] B. Miles and P. L. Lee, "Achieving reproducibility and closed-loop automation in biological experimentation with an ioT-enabled lab of the future," SLAS TECHNOLOGY: Translating Life Sciences Innovation, vol. 23, no. 5, pp. 432–439, 2018. PMID: 30045649.

[23] E. Whitehead, F. Rudolf, H.-M. Kaltenbach, and J. Stelling, "Automated planning enables complex protocols on liquid-handling robots," ACS Synthetic Biology, vol. 7, no. 3, pp. 922–932, 2018. PMID: 29486123.

[24] K. Pardee, "Perspective: Solidifying the impact of cell-free synthetic biology through lyophilization," Biochemical Engineering Journal, vol. 138, pp. 91 – 97, 2018.

[25] A. D. Silverman, N. Kelley-Loughnane, J. B. Lucks, and M. C. Jewett, "Deconstructing cell-free extract preparation for in vitro activation of transcriptional genetic circuitry," ACS Synthetic Biology, vol. 8, no. 2, pp. 403–414, 2019. PMID: 30596483.

[26] Y. Sun, A. Cuesta-Infante, and K. Veeramachaneni, "Learning vine copula models for synthetic data generation," Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, pp. 5049–5057, Jul. 2019.

[27] C. Alberti, D. Andor, E. Pitler, J. Devlin, and M. Collins, "Synthetic QA corpora generation with roundtrip consistency," in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, (Florence, Italy), pp. 6168–6173, Association for Computational Linguistics, July 2019.

[28] C. M. Bowen and F. Liu, "Comparative study of differentially private data synthesis methods," arXiv: Methodology, 2016.

[29] I. Tsoukalas, P. Kossieris, and C. Makropoulos, "Simulation of non-gaussian correlated random variables, stochastic processes and random fields: Introducing the anysim r-package for environmental applications and beyond," Water, vol. 12, no. 6, 2020.

[30] C. Vollmer, N. Sanchez, S. Gondek, J. McAuliffe, T. Kent, J. Christein, and M. Callery, "A root-cause analysis of mortality following major pancreatectomy," Journal of gastrointestinal surgery : official journal of the Society for Surgery of the Alimentary Tract, vol. 16, pp. 89–102; discussion 102, 11 2011.

[31] P. Kumari, D. Lee, Q. Wang, M. N. Karim, and J. Sang-Il Kwon, "Root cause analysis of key process variable deviation for rare events in the chemical process industry," Industrial & Engineering Chemistry Research, vol. 59, no. 23, pp. 10987–10999, 2020.

[32] D. R. Ferreira and E. Vasilyev, "Using logical decision trees to discover the cause of process delays from event logs," Comput. Ind., vol. 70, p. 194–207, June 2015.

[33] D. J. Dean, H. Nguyen, and X. Gu, "Ubl: Unsupervised behavior learning for predicting performance anomalies in virtualized cloud systems," in Proceedings of the 9th International Conference on Autonomic Computing, ICAC '12, (New York, NY, USA), p. 191–200, Association for Computing Machinery, 2012.

[34] C. Cosgriff, L. Celi, S. Ko, T. Sundaresan, M. Armengol de la Hoz, A. Kaufman, D. Stone, O. Badawi, and R. Deliberato, "Developing well-calibrated illness severity scores for decision support in the critically ill," npj Digital Medicine, vol. 2, 12 2019.

[35] R. Serban, A. Kupraszewicz, and G. Hu, "Predicting the characteristics of people living in the south usa using logistic regression and decision tree," in 2011 9th IEEE International Conference on Industrial Informatics, pp. 688–693, 2011.

[36] P. K. Dalvi, S. K. Khandge, A. Deomore, A. Bankar, and V. A. Kanade, "Analysis of customer churn prediction in telecom industry using decision trees and logistic regression," in 2016 Symposium on Colossal Data Analysis and Networking (CDAN), pp. 1–4, 2016.

[37] H. Kavade, "A logistic regression model to predict incident severity using the human factors analysis and classification system," Master's thesis, Clemson University, 12 2009.

[38] K. Thiru, P. Donnan, and F. Sullivan, "A validated logistic regression model to identify coronary heart disease patients (CHD) within primary care databases in the united kingdom," AMIA Annual Symposium proceedings, vol. 2003, p. 1030, 02 2003.

[39] T. Hastie, J. Friedman, and R. Tisbshirani, 10. Boosting and Additive Trees, p. 337–384. Springer, 2 ed., 2017.

[40] T. Chen and C. Guestrin, "Xgboost: A scalable tree boosting system," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '16, (New York, NY, USA), p. 785–794, Association for Computing Machinery, 2016.

[41] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," in Advances in Neural Information Processing Systems (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[42] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "Catboost: unbiased boosting with categorical features," in Advances in Neural Information Processing Systems (S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, eds.), vol. 31, Curran Associates, Inc., 2018.

[43] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, (Red Hook, NY, USA), p. 1025–1035, Curran Associates Inc., 2017.

[44] R. Marshall and V. Noireaux, "Quantitative modeling of transcription and translation of an all e. coli cell-free system," Scientific Reports, vol. 9, 2019.

[45] E. Rubin and A. A. Levy, "A Mathematical Model and a Computerized Simulation of PCR Using Complex Templates," Nucleic Acids Research, vol. 24, pp. 3538–3545, 09 1996.

[46] A. L. Blum and R. L. Rivest, "Training a 3-node neural network is np-complete," Neural Networks, vol. 5, no. 1, pp. 117–127, 1992.

[47] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '16, (New York, NY, USA), pp. 785–794, ACM, 2016.

[48] A. Abate, L. Cardelli, M. Kwiatkowska, L. Laurenti, and B. Yordanov, "Experimental biological protocols with formal semantics," in Computational Methods in Systems Biology (M. Češka and D. Šafránek, eds.), (Cham), pp. 165–182, Springer International Publishing, 2018.

[49] P. Pandey, "Understanding the roc and auc metrics.," Sep 2020.

[50] K. B. Mullis, F. Francois, R. A. Gibbs, and J. D. Watson, The polymerase chain reaction. Birkhäuser, 1 ed., 1994.

[51] N. E. Gregorio, M. Z. Levine, and J. P. Oza, "A user's guide to cell-free protein synthesis," Methods and Protocols, vol. 2, 2019.

[52] E. D. Carlson, R. Gan, C. E. Hodgman, and M. C. Jewett, "Cell-free protein synthesis: Applications come of age," Biotechnology Advances, vol. 30, no. 5, pp. 1185–1194, 2012.

[53] M. Vilkhovoy, N. Horvath, C.-H. Shih, J. A. Wayman, K. Calhoun, J. Swartz, and J. D. Varner, "Sequence specific modeling of e. coli cell-free protein synthesis," ACS Synthetic Biology, vol. 7, no. 8, pp. 1844–1857, 2018. PMID: 29944340.

[54] R. Hussein and D. Wilson, "Remote versus in-hand hardware laboratory in digital circuits courses remote versus in-hand hardware laboratory in digital circuits courses," in 2021 ASEE Virtual Annual Conference Content Access, 07 2021.

[55] D. Lewin, W. Seider, and J. Seader, "Integrated process design instruction," Computers & Chemical Engineering, vol. 26, no. 2, pp. 295–306, 2002.

[56] J. Brinson, "Learning outcome achievement in non-traditional (virtual and remote) versus traditional (hands-on) laboratories: A review of the empirical research," Computers and Education, vol. 87, pp. 218–237, 07 2015.

[57] T. Jong, M. Linn, and Z. Zacharia, "Physical and virtual laboratories in science and engineering education," Science (New York, N.Y.), vol. 340, pp. 305–308, 04 2013.

[58] L. Feisel, G. Peterson, O. Arnas, L. Carter, A. Rosa, and W. Worek, "Learning objectives for engineering education laboratories," in 32nd Annual Frontiers in Education, vol. 2, pp. F1D–, 2002.

[59] I. Gustavsson, J. Zackrisson, L. H*r*akansson, I. Claesson, and T. Lagö, "The visir project - an open source software initiative for distributed online laboratories," in REV 2007, 2007.

[60] F. Morgan, S. Cawley, F. Callaly, S. Agnew, P. Rocke, M. O'Halloran, N. Drozd, K. Kepa, and B. McGinley, "Remote fpga lab with interactive control and visualisation interface," in 2011 21st International Conference on Field Programmable Logic and Applications, pp. 496–499, 2011.

[61] S. Peter, F. Momtaz, and T. Givargis, "From the browser to the remote physical lab: Programming cyber-physical systems," in 2015 IEEE Frontiers in Education Conference (FIE), pp. 1–7, 2015.

[62] J. Garcia-Zubia, D. L. deIpina, and P. Orduna, "Accessing weblabs from cellular phones," in IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics, pp. 3779–3781, 2006.

[63] S. Odeh, S. Abushanab, M. Anabtawi, and R. Hodrob, "Remote augmented reality engineering labs," IEEE Global Engineering Education Conference, EDUCON, pp. 1–6, 04 2012.

[64] P. Orduña, L. Rodriguez-Gil, J. Garcia-Zubia, I. Angulo, U. Hernandez, and E. Azcuenaga, "Increasing the value of remote laboratory federations through an open sharing platform: Labsland," in Online Engineering & Internet of Things (M. E. Auer and D. G. Zutin, eds.), (Cham), pp. 859–873, Springer International Publishing, 2018.

[65] C. A. Mayoz, A. L. da Silva Beraldo, A. Villar-Martinez, L. Rodriguez-Gil, W. F. M. de Souza Seron, and P. Orduña, "Fpga remote laboratory: experience of a shared laboratory between upna and unifesp," in 2020 XIV Technologies Applied to Electronics Teaching Conference (TAEE), pp. 1–8, 2020.

[66] M. Cooper, "Remote laboratories in teaching and learning - issues impinging on widespread adoption in science and engineering education," Int. J. Online Eng., vol. 1, 2005.

[67] S. Gröber, M. Vetter, B. Eckert, and H. J. Jodl, "Experimenting from a distance–determination of speed of light by a remotely controlled laboratory (rcl).," European Journal of Physics, vol. 31, pp. 563–572, 2010.

[68] J. Basey, L. Sackett, and N. Robinson, "Optimal science lab design: Impacts of various components of lab design on students' attitudes toward lab," International Journal Scholarship of Teaching and Learning, vol. 2, 01 2008.

[69] A. Hofstein and V. N. Lunetta, "The laboratory in science education: Foundations for the twenty-first century," Science Education, vol. 88, no. 1, pp. 28–54, 2004.

[70] M. J. Callaghan, G. Bengloan, J. Ferrer, L. Cherel, M. A. El Mostadi, A. Gomez Eguíluz, and N. McShane, "Voice driven virtual assistant tutor in virtual reality for electronic engineering remote laboratories," in Smart Industry & Smart Education (M. E. Auer and R. Langmann, eds.), (Cham), pp. 570–580, Springer International Publishing, 2019.

[71] A. Ak, V. Topuz, Z. A. ALTIKARDES, and B. Oral, "Development of a remote laboratory infrastructure and lms for mechatronics distance education," Eurasia Journal of Mathematics, Science and Technology Education, vol. 14, pp. 2493–2508, 04 2018.

[72] O. Zine, M. Errouha, O. Zamzoum, A. Derouich, and A. Talbi, "Seiti rmlab: A costless and effective remote measurement laboratory in electrical engineering," The International Journal of Electrical Engineering & Education, vol. 56, no. 1, pp. 3–23, 2019.

[73] A. K. Mohammed, H. M. El Zoghby, and M. M. Elmesalawy, "Remote controlled laboratory experiments for engineering education in the post-covid-19 era: Concept and example," in 2020 2nd Novel Intelligent and Leading Emerging Sciences Conference (NILES), pp. 629–634, 2020.

[74] C. Monzo, G. Cobo, J. A. Morán, E. Santamaría, and D. García-Solórzano, "Remote laboratory for online engineering education: The rlab-uoc-fpga case study," Electronics, vol. 10, no. 9, 2021.

[75] J. Grodotzki, S. Upadhya, and A. E. Tekkaya, "Engineering education amid a global pandemic," Advances in Industrial and Manufacturing Engineering, vol. 3, p. 100058, 2021.

[76] S. Asgari, J. Trajkovic, M. Rahmani, W. Zhang, R. C. Lo, and A. Sciortino, "An observational study of engineering online education during the covid-19 pandemic," PLOS ONE, vol. 16, pp. 1–17, 04 2021.

[77] B. Aktan, C. Bohus, L. Crowl, and M. Shor, "Distance learning applied to control engineering laboratories," IEEE Transactions on Education, vol. 39, no. 3, pp. 320–326, 1996.

[78] D. Miele, B. Potsaid, and J. Wen, "An internet-based remote laboratory for control education," in Proceedings of the 2001 American Control Conference. (Cat. No.01CH37148), vol. 2, pp. 1151–1152 vol.2, 2001.

[79] J. M. Andujar, A. Mejias, and M. A. Marquez, "Augmented reality for the improvement of remote laboratories: An augmented remote laboratory," IEEE Transactions on Education, vol. 54, no. 3, pp. 492–500, 2011.

[80] A. K. M. Azad, M. E. Auer, and V. J. Harward, "Internet accessible remote laboratories: Scalable e-learning tools for engineering and science disciplines," International Journal of Online Engineering (ijoe), vol. 6, pp. 34–35, 2011.

[81] M. Tawfik, E. Sancristobal, S. Martin, G. Diaz, and M. Castro, "State-of-the-art remote laboratories for industrial electronics applications," in 2012 Technologies Applied to Electronics Teaching (TAEE), pp. 359–364, 2012.

[82] R. S. Herrera, M. A. Márquez, A. Mejías, R. Tirado, and J. M. Andújar, "Exploring the usability of a remote laboratory for photovoltaic systems," IFAC-PapersOnLine, vol. 48, no. 29, pp. 7–12, 2015. IFAC Workshop on Internet Based Control Education IBCE15.

[83] F. Esquembre, "Facilitating the creation of virtual and remote laboratories for science and engineering education," IFAC-PapersOnLine, vol. 48, no. 29, pp. 49–58, 2015. IFAC Workshop on Internet Based Control Education IBCE15.

[84] C. Bohus, L. A. Crowl, B. Aktan, and M. H. Shor, "Running control engineering experiments over the internet," IFAC Proceedings Volumes, vol. 29, no. 1, pp. 2919–2927, 1996. 13th World Congress of IFAC, 1996, San Francisco USA, 30 June - 5 July.

[85] J. L. Hardison, K. DeLong, P. H. Bailey, and V. J. Harward, "Deploying interactive remote labs using the ilab shared architecture," in 2008 38th Annual Frontiers in Education Conference, pp. S2A–1–S2A–6, 2008.

[86] H. A. Lahoud and J. P. Krichen, "Networking labs in the online environment: Indicators for success.," The Journal of Technology Studies, vol. 36, pp. 31–40, 2010.

[87] A. Coble, A. Smallbone, A. Bhave, R. Watson, A. Braumann, and M. Kraft, "Delivering authentic experiences for engineering students and professionals through e-labs," in IEEE EDUCON 2010 Conference, pp. 1085–1090, 2010.

[88] B. Pitzer, S. Osentoski, G. Jay, C. Crick, and O. C. Jenkins, "Pr2 remote lab: An environment for remote development and experimentation," in 2012 IEEE International Conference on Robotics and Automation, pp. 3200–3205, 2012.

[89] K. P. Ayodele, L. O. Kehinde, L. O. Kehinde, and B. I. Ishola, "Ac 2012-4622: Remote realistic interface experimentation using the emona datex board," in American Society for Engineering Education, 2012.

[90] P. Orduña, L. Rodriguez-Gil, D. López-de Ipiña, and J. García-Zubia, "Sharing the remote laboratories among different institutions: A practical case," in 2012 9th International Conference on Remote Engineering and Virtual Instrumentation (REV), pp. 1–4, 2012.

[91] D. Kumar, H. Sasidharakurup, R. Radhamani, N. Nizar, K. Achuthan, B. Nair, and S. Diwakar, "Mobile learning and biotechnology education via remote labs: Deployment-based study on real time shared resources," in 2015 International Conference on Interactive Mobile Communication Technologies and Learning (IMCL), pp. 39–43, 2015.

[92] J. Sáenz, J. Chacón, L. De La Torre, A. Visioli, and S. Dormido, "Open and low-cost virtual and remote labs on control engineering," IEEE Access, vol. 3, pp. 805–814, 2015.

[93] M. Alavi, "Shared remote lab," in Visions and Concepts for Education 4.0 (M. E. Auer and D. Centea, eds.), (Cham), pp. 392–399, Springer International Publishing, 2021.

[94] D. Gravano, U. Chakraborty, I. Pesce, and M. Thomson, "Solutions for shared resource lab remote quality control and instrument troubleshooting during a pandemic," Cytometry Part A, vol. 99, 11 2020.

[95] G. Sullivan and A. Artino, "Analyzing and interpreting data from likert-type scales," Journal of graduate medical education, vol. 5, pp. 541–2, 12 2013.

[96] O. Mazumder, D. Roy, S. Bhattacharya, A. Sinha, and A. Pal, "Synthetic ppg generation from haemodynamic model with baroreflex autoregulation: a digital twin of cardiovascular system," in 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), pp. 5024–5029, 2019.

[97] A. Reiner-Benaim, "Analyzing medical research results based on synthetic data and their relation to real data results: Systematic comparison from five observational studies," JMIR Medical Informatics, vol. 8, p. 16492, 02 2020.

[98] A. Gon*c*calves, P. Ray, B. Soper, J. Stevens, L. Coyle, and A. Sales, "Generation and evaluation of synthetic patient data," BMC Medical Research Methodology, vol. 20, 05 2020.

[99] A. Torfi, E. A. Fox, and C. K. Reddy, "Differentially private synthetic medical data generation using convolutional gans," Information Sciences, vol. 586, pp. 485–500, 2022.

[100] R. McKenna, G. Miklau, and D. Sheldon, "Winning the nist contest: A scalable and general approach to differentially private synthetic data," Journal of Privacy and Confidentiality (JPC) special issue on data challenges, vol. 11, 08 2021.

[101] J. Goldfeder and H. Kugler, "Temporal logic based synthesis of experimentally constrained interaction networks," in Molecular Logic and Computational Synthetic Biology (M. Chaves and M. A. Martins, eds.), (Cham), pp. 89–104, Springer International Publishing, 2019.

[102] L. Li, J. Sun, Y. Liu, M. Sun, and J.-S. Dong, "A formal specification and verification framework for timed security protocols," IEEE Transactions on Software Engineering, vol. 44, no. 8, pp. 725–746, 2018.

[103] A. Jurcut, T. Coffey, and R. Dojen, "A novel security protocol attack detection logic with unique fault discovery capability for freshness attacks and interleaving session

attacks," IEEE Transactions on Dependable and Secure Computing, vol. 16, no. 6, pp. 969–983, 2019.

[104] F. Yang, S. Escobar, C. Meadows, J. Meseguer, and S. Santiago, "Strand spaces with choice via a process algebra semantics," in Proceedings of the 18th International Symposium on Principles and Practice of Declarative Programming, PPDP '16, (New York, NY, USA), p. 76–89, Association for Computing Machinery, 2016.

[105] G. Chen, M. Liu, and J. Chen, "Frequency-temporal-logic-based bearing fault diagnosis and fault interpretation using bayesian optimization with bayesian neural networks," Mechanical Systems and Signal Processing, vol. 145, p. 106951, 2020.

[106] Y. Jiang, M. Wang, Z. Su, Y. Yang, and H. Wang, "Formal design of multi-function vehicle bus controller," IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 6, pp. 3880–3889, 2021.

[107] L. Cardelli, M. Kwiatkowska, and L. Laurenti, "A language for modeling and optimizing experimental biological protocols," Computation, vol. 9, no. 10, 2021.

[108] M. Callaghan, G. Bengloan, J. Ferrer, L. Cherel, M. Mostadi, A. Gómez Eguíluz, and N. McShane, Voice Driven Virtual Assistant Tutor in Virtual Reality for Electronic Engineering Remote Laboratories: Proceedings of the 15th International Conference on Remote Engineering and Virtual Instrumentation, pp. 570–580. Springer, 01 2019.

[109] M. K. Takahashi, C. A. Hayes, J. Chappell, Z. Z. Sun, R. M. Murray, V. Noireaux, and J. B. Lucks, "Characterizing and prototyping genetic networks with cell-free transcription–translation reactions," Methods, vol. 86, pp. 60 – 72, 2015. Bacterial and Archaeal Transcription.

[110] L. Breiman, "Bias, variance , and arcing classifiers," Technical Report 460, Statistics Department, University of California, 11 2000.

[111] K. Yilmaz, "Prediction of consumer behavior regarding purchasing remanufactured products: A logistics regression model," International Journal of Business and Social Research, vol. 06, pp. 1–10, 02 2016.

[112] V. Tsoukalas and N. Fragiadakis, "Prediction of occupational risk in the shipbuilding industry using multivariable linear regression and genetic algorithm analysis," Safety Science, vol. 83, pp. 12 – 22, 2016.

[113] P. Peduzzi, J. Concato, E. Kemper, T. R. Holford, and A. R. Feinstein, "A simulation study of the number of events per variable in logistic regression analysis," Journal of Clinical Epidemiology, vol. 49, pp. 1373–1379, 2020/08/11 1996.

[114] C. W. Seymour, V. X. Liu, T. J. Iwashyna, F. M. Brunkhorst, T. D. Rea, A. Scherag, G. Rubenfeld, J. M. Kahn, M. Shankar-Hari, M. Singer, C. S. Deutschman, G. J. Escobar, and D. C. Angus, "Assessment of Clinical Criteria for Sepsis: For the Third International Consensus Definitions for Sepsis and Septic Shock (Sepsis-3)," JAMA, vol. 315, pp. 762–774, 02 2016.

[115] S. Palei and S. Das, "Logistic regression model for prediction of roof fall risks in bord and pillar workings in coal mines: An approach," Safety Science - SAF SCI, vol. 47, pp. 88–96, 01 2009.

[116] M. Strano and B. Colosimo, "Logistic regression analysis for experimental determination of forming limit diagrams," International Journal of Machine Tools and Manufacture, vol. 46, no. 6, pp. 673 – 682, 2006.

[117] K. Murphy, Machine Learning: A Probabilistic Perspective. Adaptive Computation and Machine Learning series, MIT Press, 2012.

[118] M. Strano, "Technological representation of forming limits for negative incremental forming of thin aluminum sheets," Journal of Manufacturing Processes, vol. 7, no. 2, pp. 122 – 129, 2005.

[119] A. Ramírez, "Dynamic variable selection in dynamic logistic regression: an application to internet subscription," Empirical Economics, 03 2019.

[120] D. Malmarugan, "Strategic model for predicting customer's intention to purchase apparel online," Innovative Marketing, 05 2008.

[121] P. Kiran and V. Shanmugam, "A logistic regression model to identify the key attributes considered by consumers for purchasing a car," International Journal of Economic Research, vol. 14, pp. 161–172, 01 2017.

[122] R. Yeung and W. Yee, "Logistic regression: An advancement of predicting consumer purchase propensity," The Marketing Review, vol. 11, 03 2011.

[123] T. Rusch, I. Lee, K. Hornik, W. Jank, and A. Zeileis, "Influencing elections with statistics: Targeting voters with logistic regression trees," The Annals of Applied Statistics, vol. 7, 03 2013.

[124] B. Weeks and R. K. Garrett, "Electoral consequences of political rumors: Motivated reasoning, candidate rumors, and vote choice during the 2008 u.s. presidential election," International Journal of Public Opinion Research, vol. 26, pp. 401–422, 12 2014.

[125] W. Xu, Y. Zhao, S. Nian, L. Feng, X. Bai, X. Luo, and F. Luo, "Differential analysis of disease risk assessment using binary logistic regression with different analysis strategies," Journal of International Medical Research, vol. 46, p. 030006051877717, 06 2018.

[126] C. Zhu, C. U. Idemudia, and W. Feng, "Improved logistic regression model for diabetes prediction by integrating pca and k-means techniques," Informatics in Medicine Unlocked, vol. 17, p. 100179, 2019.

[127] B. Tabaei and W. Herman, "A multivariate logistic regression equation to screen for diabetes : Development and validation," Diabetes care, vol. 25, pp. 1999–2003, 11 2002.

[128] J. Tolles and W. J. Meurer, "Logistic Regression: Relating Patient Characteristics to Outcomes," JAMA, vol. 316, pp. 533–534, 08 2016.

[129] S. Biondo, E. Ramos, M. Deiros, J. Ragué, J. Oca, P. Moreno, L. Farran, and E. Jaurrieta, "Prognostic factors for mortality in left colonic peritonitis: A new scoring system," Journal of the American College of Surgeons, vol. 191, pp. 635–42, 01 2001.

[130] R. D. McKelvey and W. Zavoina, "A statistical model for the analysis of ordinal level dependent variables.," Journal of Mathematical Sociology, vol. 4, p. 103–112, 1975.

[131] N. J. D. NAGELKERKE, "A note on a general definition of the coefficient of determination," Biometrika, vol. 78, pp. 691–692, 09 1991.

[132] B. Efron, "Regression and anova with zero-one data: Measures of residual variation," Journal of the American Statistical Association, vol. 73, no. 361, pp. 113–121, 1978.

[133] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[134] M. R. Anderson, M. Cafarella, Y. Jiang, G. Wang, and B. Zhang, "An integrated development environment for faster feature engineering," Proc. VLDB Endow., vol. 7, p. 1657–1660, Aug. 2014.

[135] C. Park, D. Kim, J. Oh, and H. Yu, "Predicting user purchase in e-commerce by comprehensive feature engineering and decision boundary focused under-sampling," in Proceedings of the 2015 International ACM Recommender Systems Challenge, RecSys '15 Challenge, (New York, NY, USA), Association for Computing Machinery, 2015.

[136] V. Fehst, H. C. La, T.-D. Nghiem, B. E. Mayer, P. Englert, and K.-H. Fiebig, "Automatic vs. manual feature engineering for anomaly detection of drinking-water quality," in Proceedings of the Genetic and Evolutionary Computation Conference Companion, GECCO '18, (New York, NY, USA), p. 5–6, Association for Computing Machinery, 2018.

[137] M. Demetgul, "Fault diagnosis on production systems with support vector machine and decision trees algorithms," The International Journal of Advanced Manufacturing Technology, vol. 67, 08 2012.

[138] E. J. Snell and D. R. Cox, Applied statistics: a handbook of BMDP analyses. Springer Science Business Media, B.V., 1987.

[139] T. A. Domencich and D. McFadden, Urban travel demand a behavioral analysis. North-Holland Publishing Co., 1975.

[140] A. Kim, Y. Song, M. Kim, K. Lee, and J. H. Cheon, "Logistic regression model training based on the approximate homomorphic encryption," BMC Medical Genomics, vol. 11, 2018.

[141] R. Ruslan and G. Hemakumara, "Models comparison to assess emerging of low income housing in colombo suburbs," in European Proceedings of Social and Behavioural Sciences, pp. 543–553, Future Academy, 09 2019.

[142] C. Lan, L. Joseph, and D. Wolfson, "Bayesian sample size determination for binomial proportions," Bayesian Analysis, vol. 3, pp. 269–296, 06 2008.

[143] P. Green, K. Latuszynski, M. Pereyra, and C. Robert, "Bayesian computation: a summary of the current state, and samples backwards and forwards," Statistics and Computing, vol. 25, 06 2015.

[144] R. Chao, S. Mishra, T. Si, and H. Zhao, "Engineering biological systems using automated biofoundries," Metabolic Engineering, vol. 42, pp. 98 – 108, 2017.

[145] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko, "Revisiting deep learning models for tabular data," CoRR, vol. abs/2106.11959, 2021.

[146] R. Zhu, K. Zhao, H. Yang, W. Lin, C. Zhou, B. Ai, Y. Li, and J. Zhou, "Aligraph: A comprehensive graph neural network platform," Proc. VLDB Endow., vol. 12, p. 2094–2105, Aug. 2019.

[147] J. A. Schoeneberger, "The impact of sample size and other factors when estimating multilevel logistic models," The Journal of Experimental Education, vol. 84, no. 2, pp. 373–397, 2016.

[148] N. E. Biolabs, "Gibson assembly®."

[149] K. LARSEN, "Sorry arima, but i'm going bayesian," Apr 2016.

[150] Q. Guang, Hierarchical Bayesian Modeling of Health Insurance Claims. PhD thesis, Simon Fraser University, 2015.

[151] J. Schroeder, R. Karkar, J. Fogarty, J. A. Kientz, S. A. Munson, and M. Kay, "A patient-centered proposal for bayesian analysis of self-experiments for health," Journal of Healthcare Informatics Research, vol. 3, no. 1, p. 124–155, 2018.

[152] R. O. Ness, K. Sachs, P. Mallick, and O. Vitek, "A bayesian active learning experimental design for inferring signaling networks," Journal of Computational Biology, vol. 25, no. 7, p. 709–725, 2018.

[153] A. Hauser and P. Bühlmann, "Two optimal strategies for active learning of causal models from interventional data," International Journal of Approximate Reasoning, vol. 55, p. 926–939, Jun 2014.

[154] E. M. Lindgren, M. Kocaoglu, A. G. Dimakis, and S. Vishwanath, "Experimental design for cost-aware learning of causal graphs," 2018.

[155] K. Chaloner and I. Verdinelli, "Bayesian experimental design: A review," Statistical Science, vol. 10, no. 3, pp. 273–304, 1995.

[156] J. Vanlier, C. A. Tiemann, P. A. J. Hilbers, and N. A. W. van Riel, "A bayesian approach to targeted experiment design," Bioinformatics, vol. 28, no. 8, pp. 1136–1142, 2012.

[157] S. Scott and H. Varian, "Predicting the present with bayesian structural time series," Int. J. of Mathematical Modelling and Numerical Optimisation, vol. 5, pp. 4 – 23, 01 2014.

[158] R. A. Bauder and T. M. Khoshgoftaar, "A probabilistic programming approach for outlier detection in healthcare claims," in 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 347–354, Dec 2016.

[159] W. Meng, W. Li, Y. Xiang, and K.-K. R. Choo, "A bayesian inference-based detection mechanism to defend medical smartphone networks against insider attacks," Journal of Network and Computer Applications, vol. 78, pp. 162 – 169, 2017.

[160] I. Scheel, E. Ferkingstad, A. Frigessi, O. Haug, M. Hinnerichsen, and E. Meze-Hausken, "A bayesian hierarchical model with spatial variable selection: the effect of weather on insurance claims," Journal of the Royal Statistical Society: Series C (Applied Statistics), vol. 62, no. 1, pp. 85–100, 2013.

[161] L. Bermúdez, J. Pérez, M. Ayuso, E. Gómez, and F. Vázquez, "A bayesian dichotomous model with asymmetric link for fraud in insurance," Insurance: Mathematics and Economics, vol. 42, no. 2, pp. 779 – 786, 2008.

[162] D. Excell, "Bayesian inference – the future of online fraud protection," Computer Fraud & Security, vol. 2012, no. 2, pp. 8 – 11, 2012.

[163] J. Wang and Y. Zhang, "Opportunity model for e-commerce recommendation: Right product; right time," in Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '13, (New York, NY, USA), p. 303–312, Association for Computing Machinery, 2013.

[164] C. Kreutz, A. Raue, and J. Timmer, "Likelihood based observability analysis and confidence intervals for predictions of dynamic models," BMC Systems Biology, vol. 6, p. 120, Sep 2012.

[165] D. W. Scott and M. P. Wand, "Feasibility of multivariate density estimates," Biometrika, vol. 78, no. 1, pp. 197–205, 1991.

[166] M. I. Jordan, Z. Ghahramani, and L. K. Saul, "Hidden markov decision trees," in Advances in Neural Information Processing Systems 9 (M. C. Mozer, M. I. Jordan, and T. Petsche, eds.), pp. 501–507, MIT Press, 1997.

[167] E. C. Hayden, "The automated lab," Nature, vol. 516, no. 7529, p. 131–132, 2014.

[168] R. Eman, "Having hard times reproducing your experiments?."

[169] J. Salvatier, T. V. Wiecki, and C. Fonnesbeck, "Probabilistic programming in Python using PyMC3," PeerJ Computer Science, vol. 2, p. e55, Apr. 2016.

[170] J. W. Schooler, "Turning the lens of science on itself," Sep 2014.

[171] D. Gibson, L. Young, R. Chuang, J. Venter, C. Hutchison, and H. Smith, "Enzymatic assembly of dna molecules up to several hundred kilobas1," Nature methods, vol. 6, pp. 343–5, 05 2009.

[172] J. Joyce, "Bayes' theorem," Jun 2003.

[173] P. F. Wilson, G. F. Anderson, and L. D. Dell, Root cause analysis: a tool for total quality management. Milwaukee, WI, 1993.

[174] G. M. Allenby, P. E. Rossi, and R. E. McCulloch, "Hierarchical Bayes Models: A Practitioners Guide," SSRN Electronic Journal, pp. 1–44, 2005.

[175] M. Vanoni, M. Vai, and G. Frascotti, "Effects of temperature on the yeast cell cycle analyzed by flow cytometry," Cytometry, vol. 5, no. 5, p. 530–533, 1984.

[176] D. A. Freedman, "On the asymptotic behavior of bayes' estimates in the discrete case," The Annals of Mathematical Statistics, vol. 34, no. 4, pp. 1386–1403, 1963.

[177] M. Hernández de Menéndez, A. Jr, and R. Morales-Menendez, "Virtual reality laboratories: a review of experiences," International Journal on Interactive Design and Manufacturing (IJIDeM), vol. 13, 09 2019.

[178] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin, Bayesian data analysis. CRC Press, 2015.

[179] L. Pronzato, "Optimal experimental design and some related control problems," Automatica, vol. 44, no. 2, pp. 303–325, 2008.

[180] S. Bottaro and K. Lindorff-Larsen, "Biophysical experiments and biomolecular simulations: A perfect match?," Science, vol. 361, pp. 355–360, 07 2018.

[181] L. Zhu, T. Dasgupta, and Q. Huang, "A D-optimal design for estimation of parameters of an exponential-linear growth curve of nanostructures," Technometrics, vol. 56, no. 4, pp. 432–442, 2014.

[182] F. MENTRÉ, A. MALLET, and D. BACCAR, "Optimal design in random-effects regression models," Biometrika, vol. 84, no. 2, pp. 429–442, 1997.

[183] D. Barber, Bayesian Reasoning and Machine Learning. University Cambridge Press, 2012.

[184] K. Chaloner and K. Larntz, "Optimal bayesian design applied to logistic regression experiments," Journal of Statistical Planning and Inference, vol. 21, no. 2, pp. 191 – 208, 1989.

[185] V. D. B. Jojanneke, C. Andrew, and T. Jeannot, "Optimal nonlinear bayesian experimental design: an application to amplitude versus offset experiments," Geophysical Journal International, vol. 155, no. 2, pp. 411–421, 2003.

[186] D. E. Knuth, The TEX book. Addison-Wesley, 1984.

[187] D. E. Knuth, TEX: The Program. Addison-Wesley, 1986.

[188] D. E. Knuth, The Metafont book. Addison-Wesley, 1986.

[189] D. E. Knuth, Computer Modern Typefaces. Addison-Wesley, 1986.

[190] L. Lamport, LATEX: A Document Preparation System. Addison-Wesley, 2nd ed., 1994.

[191] M. Goossens, F. Mittelbach, and A. Samarin, The LATEX Companion. Addison-Wesley, 1994.

[192] Editor, "Hyphenation exception log," TUGboat, vol. 7, no. 3, p. 145, 1986.

[193] W. Shakespeare, Hamlet. NY: F.S. Crofts & Co., Inc., 1946. Act I, Scene 3, Lines 70-72, are apropos.

[194] Spivak, M.D., Ph.D., The Joy of TEX. RI: American Mathematical Society, 1986.

[195] Spivak, M.D., Ph.D., PCTEX Manual. CA: Personal TEX, Inc., 1985.

[196] G. Pal, X. Hong, Z. Wang, H. Wu, G. Li, and K. Atkinson, "Lifelong machine learning and root cause analysis for large-scale cancer patient data," Journal of Big Data, vol. 6, 12 2019.

[197] B. Sharma, P. Jayachandran, A. Verma, and C. R. Das, "Cloudpd: Problem determination and diagnosis in shared dynamic clouds," in Proceedings of the 2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), DSN '13, (USA), p. 1–12, IEEE Computer Society, 2013.

[198] T. Wang, W. Zhang, J. Wei, and H. Zhong, "Workload-aware online anomaly detection in enterprise applications with local outlier factor," in 2012 IEEE 36th Annual Computer Software and Applications Conference, pp. 25–34, 2012.

[199] K. Moloi, B. T. Abe, A. F. Nnachi, and J. A. Jordaan, "Root cause analysis and perfomance enhancement for power system network: A case study," in 2019 IEEE AFRICON, pp. 1–5, 2019.

[200] P. K. Tan, M. K. Dawood, G. R. Low, H. H. Yap, R. He, S. J. Moon, H. Feng, H. Tan, Y. M. Huang, D. D. Wang, Y. Z. Zhao, Y. Zhou, S. James, C. Q. Chen, J. Lam, and Z. H. Mai, "Nanoprobing ebac technique to reveal the failure root cause of gate oxide reliability issues of an ic process," in 2014 IEEE International Integrated Reliability Workshop Final Report (IIRW), pp. 10–15, 2014.

[201] Z. He, Y. He, F. Liu, and Y. Zhao, "Big data-oriented product infant failure intelligent root cause identification using associated tree and fuzzy dea," IEEE Access, vol. 7, pp. 34687–34698, 2019.

Appendix A
BIOLOGICAL WORKFLOWS

This chapter describes four biological workflows that serve as test cases for the proposed framework.

A.1 Polymerase Chain Reaction (PCR)

We apply the proposed framework to a complex multi-step biological workflow, namely the Polymerase Chain Reaction (PCR). PCR is a common biological technique to make many copies of a specific region of DNA in a test tube. To acquire the specific DNA region, we need five core ingredients.

1. The DNA template to be copied

2. Primers: a short sequence of nucleotides that initiate DNA synthesis. Two primers in each PCR reaction: forward primer and backward primer, are used.

3. Taq polymerase, which is a thermostable DNA polymerase, to add in the new DNA bases.

4. DNA nucleotide bases (A, C. G, and T) are fundamental building blocks of DNA. PCR relies on DNA nucleotide bases to construct the new strand.

5. Buffer is used to ensure the right conditions of the PCR reaction.

When copying the target regions of DNA, the PCR reaction will cycle through a series of temperature changes. For each time, we double the number of DNA copies. After PCR

is completed, we run electrophoresis to check the quantity and size of the DNA fragments produced.

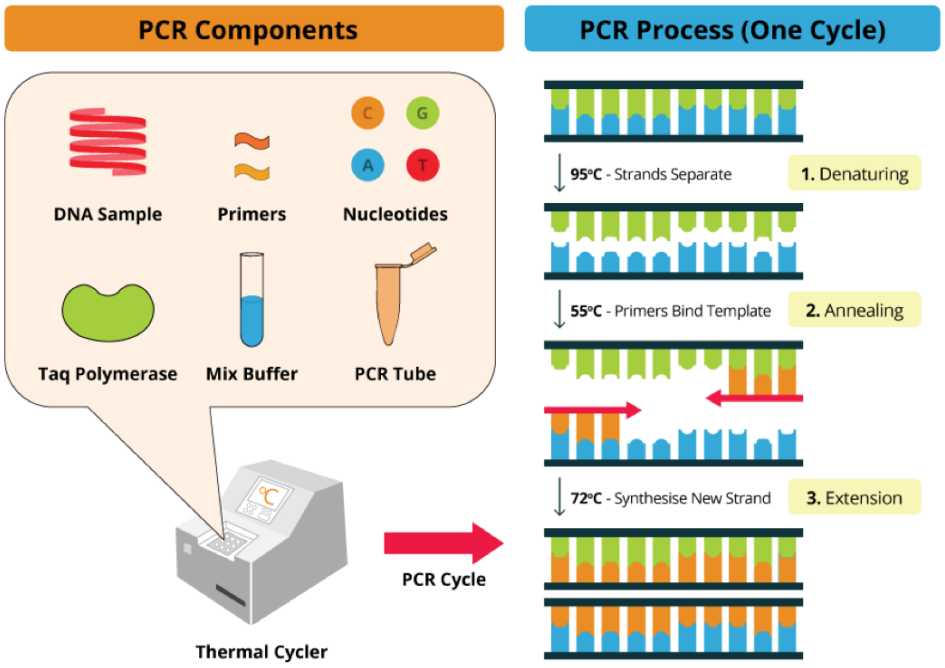


Figure A.1: The general steps and required inputs (reagents, primers, and templates) of a polymerase chain reaction (PCR) cycle. The figure is found in [2].

Figure A.2 shows a sample PCR workflow in Aquarium. Starting with *make PCR fragments*, we run *extract gel slice* and *run gel*.

Among these steps of the PCR workflows, *make PCR fragments* is an essential step. There are so many factors that determine if the operation will yield the desired band. Among these factors, the factors that dominate are:

1. incorrect or degraded template

2. wrong primer design by an inexperienced user or wrong information in Aquarium

(putting too high or too low of annealing temperature for the primer, incorrect length)

3. Kapa polymerase master mix was incorrectly made (all PCRs will fail, and this is easy to identify but still a possible source of failure)

4. technician performance error

If a PCR workflow is successful, we run *purify gel slice*, and the status of *purify gel slice* should be *done*. Otherwise, we abort *purify gel slice* instead.

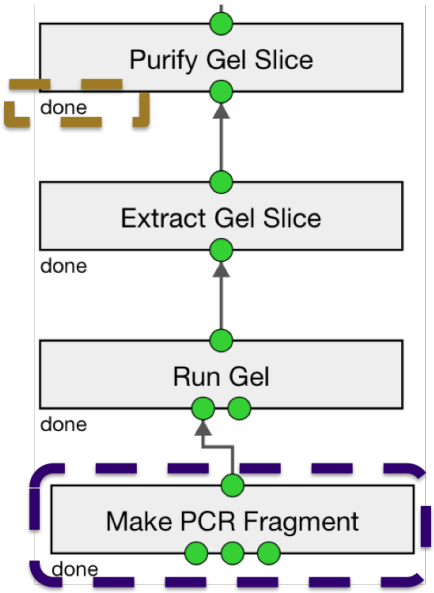


Figure A.2: Polymerase chain reaction (PCR) workflow in Aquarium.

A.2 Gibson Assembly

Gibson Assembly is a biological method to assemble DNA seamlessly [171]. This workflow is carried out under isothermal conditions using three enzymatic activities:

1. a 5' exonuclease generates long overhangs

2. a polymerase fills in the gaps of the annealed single-strand region

3. a DNA ligase seals the nicks of the annealed and filled-in gaps

As a result, different DNA fragments will join into one DNA fragment. Gibson Assembly is performed in a single reaction vessel. We combine the fragments and a master mix of enzymes and incubate the mixture solution at $50°C$ for up to one hour.

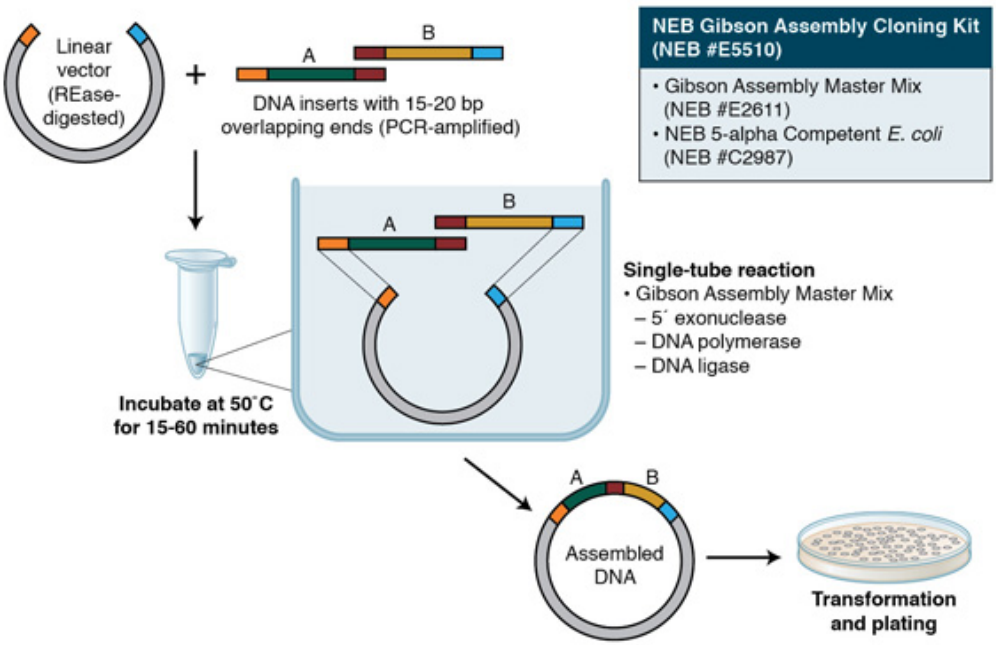


Figure A.3: Gibson Assembly workflow. [3]

We present a sample Gibson Assembly workflow in Figure A.4. This workflow starts with *Assemble Plasmid*. Following with the directed edge on the figure, the path is *transform cells*, *plate transformed cells*, *check plate*, *make overnight suspension*, *make miniprep*. Finally, we may have various *send sequencing* and *upload sequencing results*.

To decide if a Gibson Assembly workflow succeeds, we need first to check the status of *upload sequencing results* is done. Also, we need to pull *plan info* under the Aquarium system and look for an item called *item x sequencing ok?* Whenever a Gibson Assembly workflow finishes, the experiment designer checks the sequencing result. If the result is as expected, they would manually mark *yes* for *item x sequencing ok.*

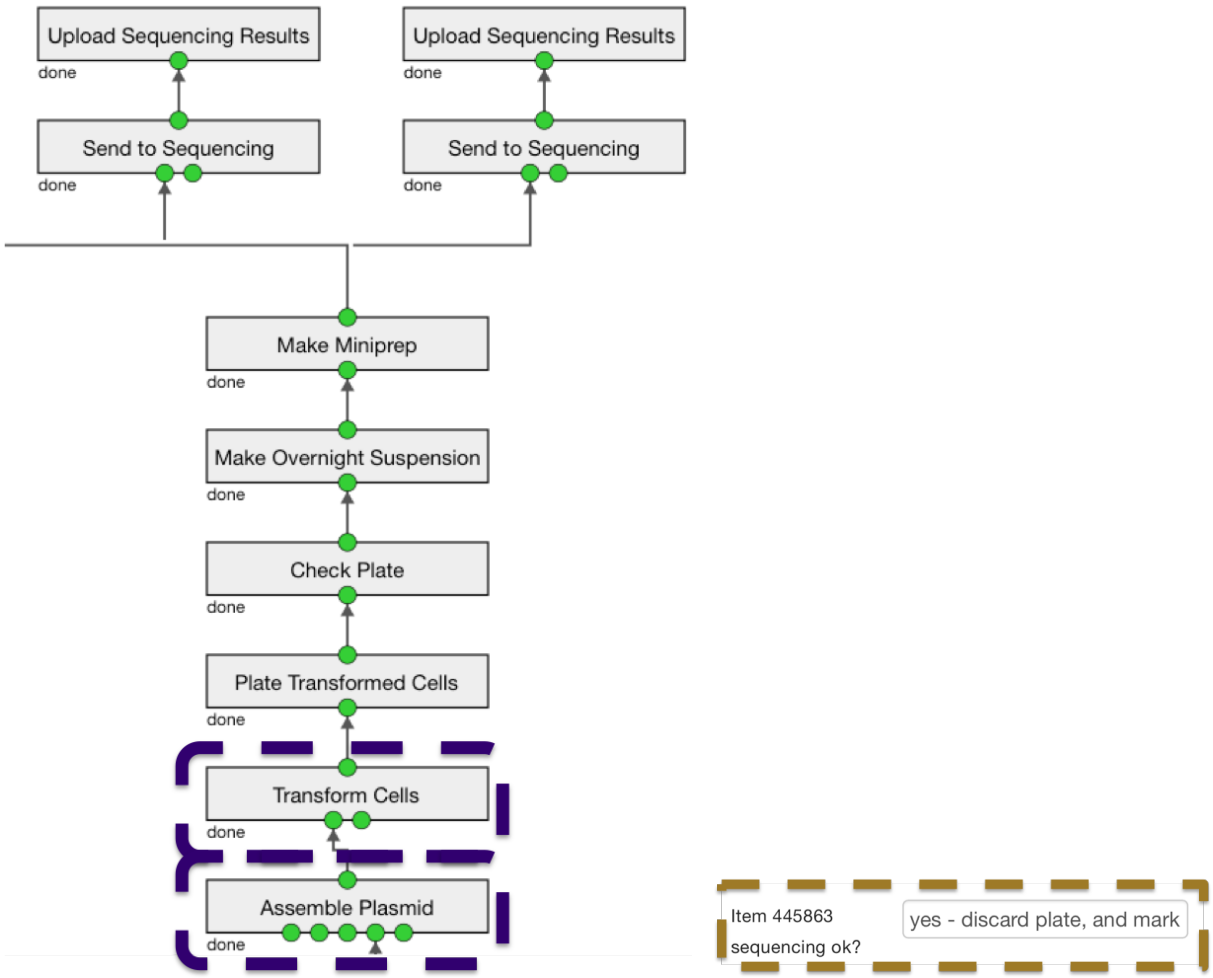
There are a few factors that determine if the desired plasmid will be produced:

1. Gibson assembly design (for example, if the fragments going into the reaction compatible with each other; if the overlap is enough between the DNA fragments for the reaction for effective work)

2. technician performance error, especially for E. coli transformation, which is a time and temperature-sensitive procedure

3. bad fragment stock coming from a PCR that was not sequence verified, since most fragments are not sequenced before assembly

4. the wrong temperature is used, or the wrong duration at the desired temperature



(a) Gibson Assembly workflow in Aquarium.

(b) The sample size is 500.

Figure A.4: Gibson Assembly result in Aquarium.

A.3 Yeast Strain Construction

Yeast is an excellent model system for studying various aspects of synthetic biology. We can control the genetics of yeast using a few sophisticated armories of molecular genetic techniques. As a result, the yeast becomes a highly approachable model system. Yeast Strain Transformation turns genetic resources into a specific strain of competent yeast cells. Lithium acetate, single-stranded carrier DNA, and polyethylene glycol are used in this yeast transformation process (PEG).

Figure A.5 denotes a typical sample *yeast strain construction* workflow in Aquarium. It starts with *yeast transformation*. Sometimes, we plate the transformed yeast cell with antibiotics. Then we run *check yeast plate*. Then, the path splits to a few branches, some of them have *yeast overnight suspension* and *yeast glycerol stock*; some of them have *yeast lysate, colony PCR*, and *Fragment analyzing*. To check if a *yeast strain construction* has been run successfully, we would need to check one input called *require QC?* under the *yeast overnight suspension*, it needed to be marked *True*. Furthermore, at least one of the statuses of *fragment analyzing* needed to be marked as *done*.

We also run some control experiments for *yeast strain construction* workflow. In this case, the operation won't create *yeast overnight suspension* or *yeast glycerol stock*. In this type of controlled workflow, we need to ensure the status of all "fragment analyzing" required to be marked as "done" to be considered a successful workflow.

Among these steps, we believe *make yeast comp cells*, *plasmid digest*, and *yeast transformation* are the most essential steps. (1) *Plasmid digest*: if the technician used an expired enzyme, and did not put DNA or buffer in the reaction, the plasmid would not be linearized, and transformation would not integrate the new gene into the yeast genome. (2) *Make yeast comp cells*: the comp cells need to be made while they are still in their log phase growth, and there is no contamination happening in the YPAD media. (3) *Yeast transformation*: this step involves multiple reagents, including PEG, Salmon sperm, LiAc, Plasmid digest, comp cells, and H2O. The success of this also depends on the user design: some auxotrophic

markers are easier to integrate into, while others are hard.

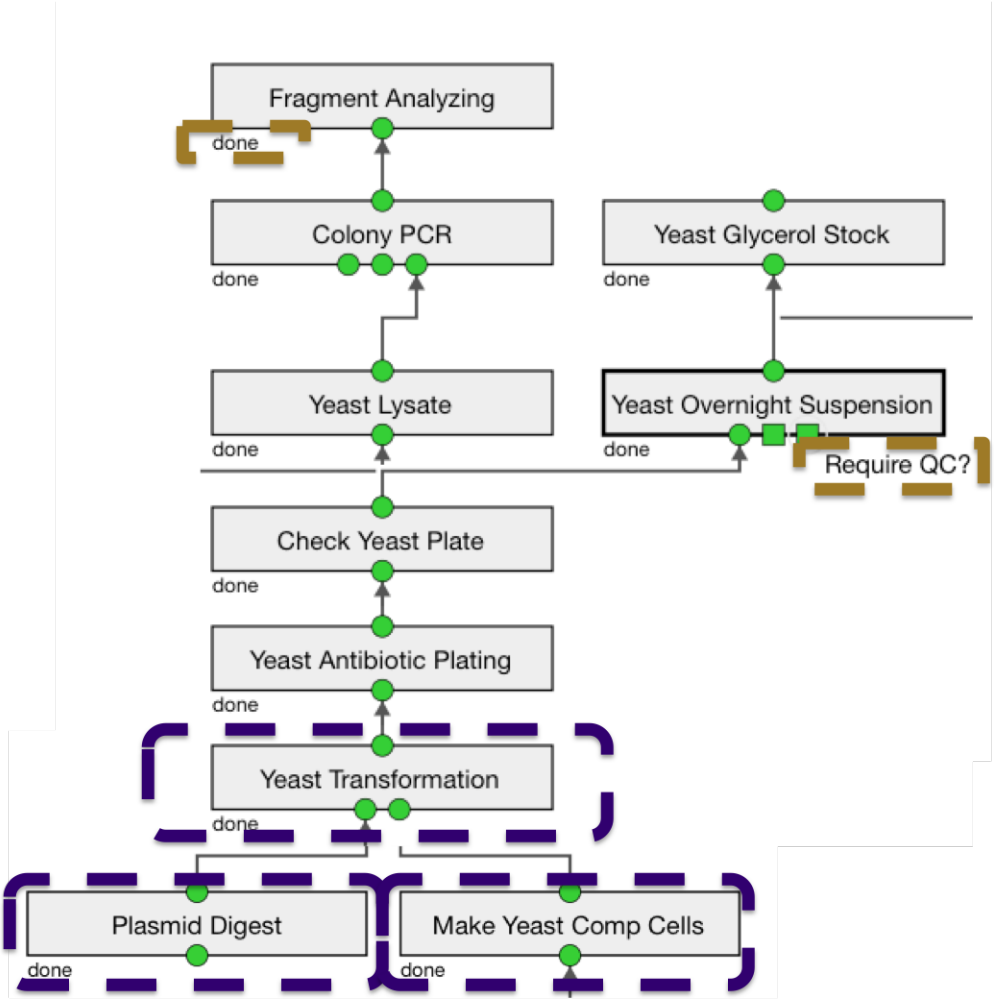


Figure A.5: Yeast Strain Construction workflow in Aquarium.