DEEP LEARNING BASED USER MODELS FOR INTERACTIVE OPTIMIZATION OF WATERSHED DESIGNS

A Dissertation

Submitted to the Faculty

of

Purdue University

by

Andrew Paul Hoblitzell

In Partial Fulfillment of the

Requirements for the Degree

of

Doctor of Philosophy

December 2019

Purdue University

West Lafayette, Indiana

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF DISSERTATION APPROVAL

Dr. Snehasis Mukhopadhyay, Chair

School of Computer and Information Science

Dr. Arjan Durresi

School of Computer and Information Science

Dr. Ananth Grama

School of Computer and Information Science

Dr. Jennifer Neville

School of Computer and Information Science

Approved by:

Clifton W. Bingham

Head of the School Graduate Program

To my parents: Paul and Elaine

To my sibling: William

Thanks for always being there for me.

ACKNOWLEDGMENTS

I would especially like to thank my mentor and advisor, Dr. Snehasis Mukhopadhyay for his encouragement and guidance during my studies. He has shown me, by his example, what a good scientist should be. I want to acknowledge my doctoral committee members Dr. Arjan Durresi, Dr. Ananth Grama, and Dr. Jennifer Neville for reviewing and providing feedback on my dissertation. Each of the members of my Dissertation Committee has provided me professional guidance and taught me a great deal about scientific research.

Nobody has been more important to me in the pursuit of this journey than the members of my family. I am grateful to my parents and sibling, who have provided me moral and emotional support in my life. I am also grateful to friends who have supported me along the way.

Thank you all for all your encouragement!

TABLE OF CONTENTS

Page

Page

Page

LIST OF TABLES

LIST OF FIGURES

Figure Page

SYMBOLS

κ	Cohen's kappa
ρ_c	concordance correlation coefficient
∞	infinity
$\int$	integral
μ	mean
σ	standard deviation
$\sum$	sum

ABBREVIATIONS

ACM	Association for Computing Machinery
ANN	Artificial neural network
AUROC	Area under receiver operating characteristic
AveP	Average precision
CNTK	Cognitive Toolkit
CRP	Conservation Reserve Program
CSP	Conservation of Security Program
CUDA	Compute Unified Device Architecture
cuDNN	CUDA Deep Neural Network library
DCG	Discounted cumulative gain
ECW	Eagle Creek Watershed
EQIP	Environmental Quality Incentive Program
FPR	False Positive Rate
GIS	Geographical Information System
GPU	Graphical processing unit
HPC	High Performance Computing
IEEE	Institute of Electrical and Electronics Engineers
IGAMI2	Interactive genetic algorithm with mixed-initiative 2
IM	Introspection Manager
INFORMS	Institute for Operations Research and the Management Sciences
MAE	Mean Absolute Error
MCDM	Multi-Criteria Decision-Making
MIM	Mixed-Initiative Manager
MRR	Mean reciprocal rank

MSCDM	Multi-Stakeholder Consensus Decision-Making
MSE	Mean Squared Error
NDCG	Normalized discounted cumulative gain
NRCS	Natural Resources Conservation Services
NSGA	Non-dominated sorting genetic algorithm
OM	Optimization Manager
P@K	Prediction at k
pip	PIP installs packages
RL	Reinforcement Learning
RMSE	Root Mean Squared Error
SDM	Simulated Decision Maker
SDMM	Simulated Decision Maker Manager
SWAT	Soil and Water Assessment Tool
TPR	True Positive Rate
USDA	United States Department of Agriculture
WHIP	Wildlife Habitat Incentives Program
WRESTORE	Watershed REstoration using Spatio-Temporal Optimization of REsources
WRP	Wetlands Reserve Program

ABSTRACT

Hoblitzell, Andrew Ph.D., Purdue University, December 2019. Deep Learning Based User Models for Interactive Optimization of Watershed Designs. Major Professor: Snehasis Mukhopadhyay.

This dissertation combines stakeholder and analytical intelligence for consensus decision-making via an interactive optimization process. This dissertation outlines techniques for developing user models of subjective criteria of human stakeholders for an environmental decision support system called WRESTORE. The dissertation compares several user modeling techniques and develops methods for incorporating such user models selectively for interactive optimization, combining multiple objective and subjective criteria.

This dissertation describes additional functionality for our watershed planning system, called WRESTORE (Watershed REstoration Using Spatio-Temporal Optimization of REsources) (http://wrestore.iupui.edu). Techniques for performing the interactive optimization process in the presence of limited data are described. This work adds a user modeling component that develops a computational model of a stakeholder's preferences and then integrates the user model component into the decision support system.

Our system is one of many decision support systems and is dependent upon stakeholder interaction. The user modeling component within the system utilizes deep learning, which can be challenging with limited data. Our work integrates user models with limited data with application-specific techniques to address some of these challenges. The dissertation describes steps for implementing accurate virtual stakeholder models based on limited training data.

Another method for dealing with limited data, based upon computing training data uncertainty, is also presented in this dissertation. Results presented show more stable convergence in fewer iterations when using an uncertainty-based incremental sampling method than when using stability based sampling or random sampling. The technique is described in additional detail.

The dissertation also discusses non-stationary reinforcement-based feature selection for the interactive optimization component of our system. The presented results indicate that the proposed feature selection approach can effectively mitigate against superfluous and adversarial dimensions which if left untreated can lead to degradation in both computational performance and interactive optimization performance against analytically determined environmental fitness functions.

The contribution of this dissertation lays the foundation for developing a framework for multi-stakeholder consensus decision-making in the presence of limited data.

1. INTRODUCTION

1.1 Overview of Problem

> *"In an extreme view, the world can be seen as only connections, nothing else. We think of a dictionary as the repository of meaning, but it defines words only in terms of other words. I liked the idea that a piece of information is really defined only by what it's related to, and how it's related. There really is little else to meaning. The structure is everything. There are billions of neurons in our brains, but what are neurons? Just cells. The brain has no knowledge until connections are made between neurons. All that we know, all that we are, comes from the way our neurons are connected."- Tim Berners Lee*

With complex problems involving multiple stakeholders, there is not always a single objectively correct solution. Many users collaborate together on the optimal solution, each with their own partial perspective and viewpoint, synthesizing these perspectives into a single decision can become complicated. Providing a viewpoint fusion mechanism is a non-trivial task involving negotiation and consensus building. In addition, each stakeholder may have individual decisions that could serve their group's best interests so a proper fusion mechanism should also encourage decisions which will provide other characteristics such as fairness.

Joint machine-user decision making allows machines to recommend analytically sound designs while also allowing stakeholders to provide more information to the decision-making process based on their knowledge and subjective preferences. Multi-stakeholder consensus decision-making involves gathering several solutions, evaluating and sorting them objectively, and then providing the best competing alternatives to stakeholders. The viewpoints of stakeholders and stakeholder groups can be learned

over time to bootstrap the multi-objective interactive optimization process, leading to consensus negotiations which require less input and negotiation from stakeholders whose time can often be a resource constraint.

1.2 Background

> *"Real cognitive science, however, is necessarily based on experimental investigation of actual humans or animals. We will leave that for other books, as we assume the reader has only a computer for experimentation."- Peter Norvig, Artificial Intelligence: A Modern Approach*

This dissertation discusses work performed for an environmental planning system named WRESTORE, which performs modeling on the Eagle Creek Watershed in Indiana. Eagle Creek Watershed is divided into total 2,953 potential wetlands. Droughts and flooding can be common which can decrease the quality of drinking water, impact crop production, and cause other undesirable outcomes. WRESTORE enables stakeholders to visualize the wetland space and conduct improved environmental planning to alleviate some of these issues.

Interactive optimization can be used as a method of solving problems where the search space is very large. Machines using the Soil and Water Assessment tool can keep the system in a Pareto optimum, while stakeholder's preferences can capture additional subjective criteria and domain knowledge.

Some issues arise in human-guided search, one of the more important being the issue of interaction fatigue. There are multiple methods of relieving interaction fatigue, some of which will be examined in greater detail later in the dissertation. A Simulated Decision Maker (SDM), or virtual user model, can be trained to learn the preferences of a stakeholder, but training can be difficult when the environment being learned is stochastic and non-stationary.

A distributed system of several machines over multiple sites is utilized in WRESTORE, with clusters to run resource-intensive environmental models. The system

has been designed with the resource characteristics of the problem in mind. More description of the distributed system is available in previous work. More information about the WRESTORE system is available later in the dissertation.

The goal of this dissertation is to contribute to an interactive decision support system that can arrive at stakeholder-optimal solutions in shorter timeframes. This dissertation thus compares different methods for building user models, suggests reliable ways for training user models with limited input, integrates user models into an interactive multi-objective system, and allows for the selection of specific user models when many different user models are present.

1.3 Scope, Expected Outcome, and Limitations

> *"We will make machines that can reason, think and do things better than we can."- Sergey Brin*

This dissertation focuses on developing a framework that supports multi-stakeholder consensus building. Human and machine involvement is used in the decision-making, with the machine guiding stakeholders during negotiations. This dissertation expects to develop user models which are reliable with less input, integrate these methods into the existing system, and develop methods for user selection during the optimization process.

This dissertation builds a simulator for stakeholder simulation and consensus building. This dissertation does not include performing an analysis of individual stakeholder's personal best interests, and some of the proposed solutions may not always adequately handle concepts like fairness and trustworthiness. There is work from others in our group that examines some of the ideas which are outside the scope of this individual dissertation.

1.4 Remaining Dissertation Structure

"Zero information is preferred to misleading or false information"- Jimmy Wales

- Chapter 1 "Introduction". Chapter 1 provides an overview of the dissertation, background, goals of the dissertation, the scope of the dissertation, and a structure for the remainder of the dissertation.
- Chapter 2 "Evaluation of Machine Learning Approaches". Chapter 2 discusses some of the many approaches to evaluating machine learning tasks so that we can consider what proper evaluation should look like for our system.
- Chapter 3 "WRESTORE and IGAMI2". Chapter 3 provides background information about the existing use case which will be expanded upon through the remainder of the dissertation.
- Chapter 4 "Comparison of Neural Methods for User Modeling in Wetland Design". Chapter 4 builds a user modeling component employing a deep learning neural network approach and examines why working with limited data is a complex task.
- Chapter 5 "Fuzzy and Deep Learning Approaches for User Modeling in Wetland Design". Chapter 5 presents approaches to dealing with limited data in the user modeling task. The originality of this approach involves application-specific data augmentation and the human-computer collaborative problem-solving approach.
- Chapter 6 "Uncertainty-Based Deep Learning for Wetland Design". Chapter 6 presents a method for dealing with limited data in deep networks based on the sampling of remaining training data with uncertainty based modeling.
- Chapter 7 "Non-Stationary Reinforcement-Learning Based Dimensionality Reduction for Multi-objective Optimization of Wetland Design". Chapter 7 studies

non-stationary reinforcement-based learning for feature selection, with the goal of performing fusion on multiple stakeholder groups to simplify the interactive optimization process.

- Chapter 8 Conclusions and Future Work. Chapter 8 describes the findings of the dissertation and provides areas for future work and extensions to the work.

2. EVALUATION OF MACHINE LEARNING APPROACHES

2.1 Abstract

"Most improved things can be improved."- Mokokoma Mokhonoana

Many approaches exist for machine learning for solving problems of different types. Objective functions, also known as cost functions or loss functions, offer one method of quantifying the performance of machine learning methods on some of these tasks. This chapter identifies some of the ways to measure machine learning algorithms on common tasks so that we can consider the way to measure the best outcomes for our system.

2.2 Classification measures

"An algorithm must be seen to be believed."- Donald Knuth, Leaders in Computing: Changing the digital world

Classification is the statistical problem of identifying the type label to apply to an unlabeled instance, for example providing a medical diagnosis. In machine learning and mathematical optimization, classifier measures are typically loss functions representing the cost of an inaccurate prediction, considering factors like the likelihood of an incorrect decision and subsequent costs of an incorrect prediction.

2.2.1 Classical measures

"The only truly secure system is one that is powered off, cast in a block of concrete and sealed in a lead-lined room with armed guards."- Gene Spafford

Several classical measures exist for measuring classification accuracy. The miss rate tells us the number of false negatives divided by the number of total positive instances and represents a false negative rate. The fall-out represents the number of false positives divided by the total number of total negative instances and represents the false positive rate. The number of false positives divided by all positives is known as the false discovery rate, while the number of false negatives over the number of total negatives is known as the false omission rate.

The number of true positives divided by all known positives is known as the true positive rate, or also as the recall. The number of true negatives divided by all of the known negatives is known as the true negative rate, or also as the selectivity. The number of true positives divided by all predicted positives is known as the positive predictive value or precision, while the number of true negatives divided by all known negatives is known as the negative predictive value.

Accuracy is defined as the number of true positives and true negatives divided by all samples. The F1 score is a widely used metric which is the harmonic average of the precision and recall. Informedness is a single metric which summarizing the performance of a diagnostic test for a multi-class case. The sum of the positive predictive value and negative predictive value minus one gives us the markedness metrics. When we are dealing with binary classification problems, we can also look at the Matthews correlation coefficient as a class imbalance resistant correlation coefficient.

2.2.2 AUROC

"We have to stop optimizing for programmers and start optimizing for users."- Jeff Atwood

miss rate (FNR)

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{P}} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TP}}$$

fall-out (FPR)

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{N}} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}$$

false discovery rate (FDR)

$$\mathrm{FDR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TP}}$$

false omission rate (FOR)

$$\mathrm{FOR} = \frac{\mathrm{FN}}{\mathrm{FN} + \mathrm{TN}}$$

Figure 2.1.: Classical measures

Area under the ROC curve (usually called AUROC) provides a metric for whether a classifier will rank a randomly chosen positive instance higher than a randomly chosen negative one. [1] This is formally given by

$$\begin{aligned} A &= \int_{x=0}^{1} \mathrm{TPR}(\mathrm{FPR}^{-1}(x))\,dx = \int_{\infty}^{-\infty} \mathrm{TPR}(T)\mathrm{FPR}'(T)\,dT \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} I(T' > T) f_1(T') f_0(T)\,dT'\,dT = P(X_1 > X_0) \end{aligned}$$

where X_1 is a positive instance score, X_0 is a negative instance score, f_0 and f_1 are probability densities, FPR is the false positive rate (on the independent axis) and TPR is the true positive rate (on the dependent axis).

sensitivity/recall (TPR)

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{P}} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

specificity/selectivity (TNR)

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{N}} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

precision/positive predictive value (PPV)

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

negative predictive value (NPV)

$$\mathrm{NPV} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FN}}$$

Figure 2.2.: Additional classical measures

2.2.3 Cohen's kappa

"Where is the 'any' key?"- Homer Simpson, in response to the message, "Press any key"

Cohen's kappa considers the agreement between two classifiers rating N items being classified into C categories. κ is formally described by:

$$\kappa \equiv \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e},$$

accuracy (ACC)

$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}$$

F1 score

$$F_1 = 2 \times \frac{PPV \times TPR}{PPV + TPR} = \frac{2 \times TP}{2 \times TP + FP + FN}$$

Informedness (BM)

$$BM = TPR + TNR - 1$$

Markedness (MK)

$$MK = PPV + NPV - 1$$

Matthews correlation coefficient (MCC)

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

Figure 2.3.: Additional classical measures

where p_0 represents relative observed agreement among the classifiers and p_e represents the hypothetical probability of chance agreement between the classifiers. When $\kappa = 1$, the raters are in complete agreement, when $\kappa = 0$ it means the agreement between classifiers is purely random, and when $\kappa < 0$ [2] it means the agreement between classifiers is worse than random. Some issues with Cohen's κ include indices of agreement interpretability, oversimplification with a single metric, and the fact that it can become undefined when the denominator evaluates to zero.

2.3 Regression measures

"Before software should be reusable, it should be usable."- Ralph Johnson

Regression is the statistical problem of understanding how one or more dependent variables change in the presence of one or more independent variables, for example sales forecasting. Regression measures in machine learning are used as a benchmark of the predicted error between prediction and reality. They are optimized against to find more ideal results. Many assumptions are made in the case of regression problems, including consistent, unbiased, and efficient estimators, uncorrelated and homoscedastic error, etc. that may not always translate well to actual datasets.

2.3.1 RMSE

"Make everything as simple as possible, but not simpler."- Albert Einstein

Root mean squared error is a regression measure that compares forecasting errors of models for a singular dataset.

The RMSE of an estimator $\hat{\theta}$ for an estimated parameter θ is given by the square root of the mean square error:

$$\mathrm{RMSE}(\hat{\theta}) = \sqrt{\mathrm{MSE}(\hat{\theta})} = \sqrt{\mathrm{E}((\hat{\theta} - \theta)^2)}.$$

The RMSE is never negative and lower values are considered ideal. A few issues exist with RMSE, including the fact that it is scale-dependent and also sensitive to outliers [3].

2.3.2 MSE

"On two occasions I have been asked, 'If you put into the machine wrong figures, will the right answers come out?' I am not able rightly to apprehend the kind of confusion of ideas that could provoke such a question."- Charles Babbage

The mean squared error (MSE) regression metric assesses the quality of a model with respect to the difference between the predicted and observed values. The most general form of the MSE is called a predictor and compares a function mapping arbitrary inputs to samples from a random variable.

MSE is formally given by

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2.$$

where n is the number of predictions on all variables and Y is the vector of observed values of the variable being predicted.

If q points are held back for cross-validation, we can compute the mean squared prediction error (MSPE) as

$$\text{MSPE} = \frac{1}{q}\sum_{i=n+1}^{n+q}(Y_i - \hat{Y}_i)^2.$$

MSE is particularly notable in the machine learning community because it perfectly illustrates the trade-off between bias and variance; further, in the case of unbiased estimators, the MSE and variance are equivalent [4]. The relation between MSE and bias and variance is given below:

$$\begin{aligned}
\text{MSE}(\hat{\theta}) &= \text{E}_\theta\left[(\hat{\theta}-\theta)^2\right] \\
&= \text{E}_\theta\left[\left(\hat{\theta}-\text{E}_\theta[\hat{\theta}]\right)^2 + 2\left(\hat{\theta}-\text{E}_\theta[\hat{\theta}]\right)\left(\text{E}_\theta[\hat{\theta}]-\theta\right) + \left(\text{E}_\theta[\hat{\theta}]-\theta\right)^2\right] \\
&= \text{E}_\theta\left[\left(\hat{\theta}-\text{E}_\theta[\hat{\theta}]\right)^2\right] + 2\left(\text{E}_\theta[\hat{\theta}]-\theta\right)\text{E}_\theta\left[\hat{\theta}-\text{E}_\theta[\hat{\theta}]\right] + \left(\text{E}_\theta[\hat{\theta}]-\theta\right)^2 \quad {}^{1} \\
&= \text{E}_\theta\left[\left(\hat{\theta}-\text{E}_\theta[\hat{\theta}]\right)^2\right] + \left(\text{E}_\theta[\hat{\theta}]-\theta\right)^2 \\
&= \text{Var}_\theta(\hat{\theta}) + \text{Bias}_\theta(\hat{\theta},\theta)^2
\end{aligned}$$

MSE is very widespread in its usage, but this is more due to its notable and convenient mathematical properties than its translation to empirical use cases. Like many other metrics it is somewhat arbitrary outside of this mathematical convenience, and it also has drawbacks with its usage including but not limited to an even higher sensitivity to outliers than RMSE.

2.3.3 MAE

"I think computer viruses should count as life. I think it says something about human nature that the only form of life we have created so far is purely destructive. We've created life in our own image."- Stephen Hawking

Mean absolute error (MAE) is a regression metric given by the difference between two continuous variables. MAE is formally described as

$$\text{MAE} = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}. \text{ [5]}$$

where n is the number of samples, e_i is the error for this particular instance, y_i is the observed value, x_i is the predicted value, and i is the currently considered instance. If we wish to allow the error to become negative for our use case, the Mean Error (ME) is given by:

$$\text{ME} = \frac{\sum_{i=1}^{n} y_i - x_i}{n}. \text{ [6]}$$

MAE can be used as a forecast error in time series analysis, offering a way to compare forecasts with eventual outcomes. One of the main advantages of MAE over MSE and RMSE is the simplicity it is calculated with makes it much more interpretable.

2.3.4 F1 Score

"In an information economy, the most valuable company assets drive themselves home every night. If they are not treated well, they do not return the next morning."- Peter Chang

The F1 score, also sometimes referred to as the F-score or F-measure, is a metric for considering the accuracy of classification tasks. The F1 score is given by

$$\text{F1} = \frac{2 * p * r}{p + r}.$$

where the precision p is the number of correct positive results divided by the number of all positive results and the recall r is the number of correct positive results divided by the number of samples which should have been identified as positive. The F1 score is a harmonic average between these two numbers. An F1 score of 1 indicates perfect precision and recall, while an F1 score of 0 indicates the worst case F1 score. Common criticisms of F1 score include ease of interpretability, the equal weighting of precision and recall which may not be appropriate for certain use cases, and the focus on a singular number rather than a more comprehensive outlook of how users will utilize the classifier and its output.

2.4 Clustering measures

"The most important property of a program is whether it accomplishes the intention of its user."- C.A.R. Hoare

In machine learning and data science, clustering involves grouping points or data instances into two or more groupings where the similarity within groups is higher than the similarity across groups. Common metrics for clustering can include external evaluation, internal evaluation, and cluster tendency. Clustering can be used for different rationales, thus evaluating clustering is highly subjective and in some instances even thought of as a multi-objective optimization problem.

2.4.1 Rand index

"The question of whether a computer can think is no more interesting than the question of whether a submarine can swim."- Edsger W. Dijkstra

The Rand index is a clustering metric which measures the similarity between two clusters. The Rand index R [7] can formally be defined as

$$\mathrm{R} = \frac{a+b}{\binom{n}{2}}.$$

where we have n elements in set $S = \{o_1, \ldots, o_n\}$ and two partitions of S to compare, $X = \{X_1, \ldots, X_m\}$, a partition of S into m subsets, and $Y = \{Y_1, \ldots, Y_l\}$, a partition of S into l subsets, a is the number of pairs of in the same subset in X and in the same subset in Y and b is the number of pairs in different subsets in X and in different subsets in Y.

A Rand index of 0 means clusters do not agree on any pair of points while 1 indicates that the two clusters are exactly the same. Corrected and adjusted values of the Rand index also exist.

2.4.2 Mutual Information

> *"What one programmer can do in one month, two programmers can do in two months."- Fred Brooks*

Mutual information is a clustering metric which quantifies the amount of information obtained about one variable by observing the other variable. The mutual information can be formally given as [8]

$$\mathrm{I}(X;Y) = \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} p(x,y) \log \left(\frac{p(x,y)}{p(x)\, p(y)} \right)$$

where X and Y are two discrete random variables, $p(x,y)$ is the joint probability function of X and Y, and $p(x)$ and $p(y)$ are the marginal probability distribution functions of X and Y.

For continuous random variables, this can be rewritten as: [8]

$$\mathrm{I}(X;Y) = \int_{\mathcal{Y}} \int_{\mathcal{X}} p(x,y) \log \left(\frac{p(x,y)}{p(x)\, p(y)} \right) dx\, dy,$$

Several variations of mutual information exist. Mutual information has found usage in many applications, including search engines and physics.

2.5 Information retrieval measures

"The business we're in is more sociological than technological, more dependent on workers' abilities to communicate with each other than their abilities to communicate with machines."- Tom DeMarco, Peopleware: Productive Projects and Teams

Information retrieval (IR) is the task of obtaining resources from a collection and often involves text searching and content-based indexing. IR metrics attempt to quantify the ability of an IR system to meets the needs of its users and have a simplified notion of ground truth relevancy built in. Other metrics for IR systems can be concerned with popular queries, revenue generated, latency, efficiency, usability, or even trustworthiness.

2.5.1 Average precision

"In theory there is no difference between theory and practice. In practice there is."- Yogi Berra

In information retrieval tasks, precision is the fraction of relevant resources retrieved divided by the number of retrieved documents. We can formally define average precision over a continuous space as

$$\text{AveP} = \int_0^1 p(r)dr$$

but because our documents or resources are often discrete we can rewrite this as

$$\text{AveP} = \sum_{i=1}^{n} P(k)\Delta r(i)$$

where i is the rank in the sequence of retrieved documents, n is the number of retrieved documents, and $P(i)$ is the precision at cutoff i in the list. This function also has other representations within literature [9] with some authors interpolating the probability

function to simplify the curve obtained [10, 11] while others assume decision values follow a Gaussian distribution [12].

When considering a set of queries, the mean average precision is given by

$$\text{MAP} = \frac{\sum_{q=1}^{Q} \text{AveP(q)}}{Q}$$

where Q is the number of queries.

R-Precision is a special kind of precision where precision is equal to recall at the R-th position [13]. It is usually correlated strongly with mean average precision. [11]

2.5.2 Precision at K

"The purpose of computing is insight, not numbers."- Richard Hamming

In systems where there massive amounts of resources, it may be that a user will put significantly higher importance on the top k resources than on the bottom (n-k) resources. Precision at k (P@k) is useful for these types of problems because it does not provide weighting to resources a user will never view or be interested in. Drawbacks of P@k are that it ignores position within the top k and for queries with fewer relevant results than k obtaining a score of 1 becomes impossible. [13]

2.5.3 Discounted cumulative gain

"Considering the current sad state of our computer programs, software development is clearly still a black art, and cannot yet be called an engineering discipline."- William Jefferson Clinton

Discounted cumulative gain (DCG) is an information retrieval metric motivated by the premise that the further up a search list you go the more important the results are to a user and the lower down the search list you go the less important the results are to the user. DCG utilizes a graded relevance to give the gain of a resource based on its result position. DCG is formally defined by

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}.$$

The normalized version of DCG (nDCG) sorts documents of a result list by relevance, producing an ideal DCG at position p ($IDCG_p$). nDCG is given by

$$\mathrm{nDCG_p} = \frac{DCG_p}{IDCGp}.$$

nDCG values can average several queries to obtain rankings for an IR algorithm. An nDCG of 1 indicates an ideal IR algorithm, while an nDCG of 0 indicates a worst-case scenario. nDCG is motivated by the idea that different queries and systems will return result sets of different sizes. One of the principal challenges with nDCG is that sometimes ideal ordering of results is not always available.

2.5.4 Mean Reciprocal Rank

> *"Controlling complexity is the essence of computer programming."- Brian Kernighan*

The mean reciprocal rank (MRR) is an information retrieval metric for evaluating processes that produce a list of responses to a sample of queries. Mean reciprocal rank is formally given by

$$\mathrm{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\mathrm{rank}_i}.$$

where rank_i is the rank position of the first relevant document of the i-th query.

The reciprocal rank of a query is 1 divided by the rank of the first correct answer, or the multiplicative inverse of the correct answer (thus, it is undefined when the correct answer is not returned). The MRR's reciprocal value corresponds to the harmonic mean of the ranks. [14, 15]

2.6 Other measures

"It is possible to invent a single machine which can be used to compute any computable sequence."- Alan Turing

2.6.1 Concordance and discordance

"Computers are good at following instructions, but not at reading your mind."- Donald Knuth

Concordance measures the agreement between two variables, with the concordance correlation coefficient ρ_c given by

$$\rho_c = \frac{2\rho\sigma_x\sigma_y}{\sigma_x^2 + \sigma_y^2 + (\mu_x - \mu_y)^2},$$

where μ_x and μ_y are the means for the two variables and σ_x^2 and σ_y^2 are the corresponding variances. ρ gives us the correlation coefficient between the two variables.

While an ordinary correlation coefficient is unaffected by whether biased or unbiased estimations of variance are used, the concordance correlation coefficient is not. Different methods exist for normalizing the coefficient exist [16] [17].

3. WRESTORE AND IGAMI2

3.1 Abstract

"AI is akin to building a rocket ship. You need a huge engine and a lot of fuel. The rocket engine is the learning algorithms but the fuel is the huge amounts of data we can feed to these algorithms."- Andrew Ng

This chapter provides background information about the existing use case which will be expanded upon through the remainder of the dissertation. Watershed REstoration using Spatio-Temporal Optimization of Resources (WRESTORE) is a decision support system for the interactive optimization of watershed designs. The second version of the Interactive Genetic Algorithm with Mixed-Initiative (IGAMI2) is a distributed algorithm used in WRESTORE. This chapter outlines the decision-making framework and may be useful in increasing understanding and comprehension of the remainder of the dissertation. Additional information about these systems is available in previous work.

3.2 WRESTORE

"A deep learning system doesn't have any explanatory power."- Geoffrey Hinton

WRESTORE is a user-friendly, interactive, decision support system that helps landowners and other stakeholders in the implementation of conservation practices. The interactive nature of the system provides decision-makers the ability to learn about the possible physical and socio-economic impacts of implementing different solutions on their watershed.

Eagle Creek Watershed (ECW) is a flat-land covering around 162 square miles about 10 miles northwest of downtown Indianapolis. Among many uses, ECW is used as a water drinking supply. It stretches over four different counties: Marion, Hendricks, Boone, and Hamilton. It is located in an agriculture-dominant area with mostly the growth of corn and soybeans. The watershed is divided into approximately 130 sub-basins connected to one another, although these can be further subdivided.

3.2.1 Wetlands

> *"More importantly, it is difficult to study minds because we are mental beings. We have our own minds to maintain and protect, and may not wish to discover facts that force us to change, or make us question our own being in the world, or conflict with our sense of right and wrong. We have not discussed belief systems known as religions to any extent in this book. However, particularly threatening are facts that run counter to our religious beliefs, especially if those beliefs are strongly held. Further, scientists have hopes, standards, and ethical beliefs, and they... like anybody... are not eager to find that their beliefs are invalid."- James Kennedy, Swarm Intelligence*

Wetlands are usually situated on the edge of the aquatic and terrestrial landscape and act as a unique bionetwork. They support local crops and the vegetation by keeping the soil saturated. Their ecological characteristics may include hydrologic, soil and biotic conditions, and they can produce soil that can support hydrophytic plantation. Wetland soil is low in oxygen and disrupts the growth of microorganisms.

Wetlands provide several ecological benefits including storing flood waters, reducing peak flow, dissipating wave energy and stabilizing shorelines, and absorbing excess fertilizers, septics, eroded soil particles, and pollutants. Wetlands also have intangible benefits to the community, such as hunting, fishing, hiking, and nature photography.

Wetlands also provide an ecological research opportunity and have been shown to boost the value of nearby land.

3.2.2 Filter Strips

> *"In the history of ideas, it's repeatedly happened that an idea, developed in one area for one purpose, finds an unexpected application elsewhere. Concepts developed purely for philosophy of mathematics turned out to be just what you needed to build a computer. Statistical formulae for understanding genetic change in biology are now applied in both economics and in programming."- Patrick Grim*

Filter strips are located at the lower edges of a landscape and remove residue, organic materials, and various other pollutants from wastewater. They act as a buffer between the water and the fields so that the pesticides and other pollutants do not spread as far. Filter strips decrease the overall maintenance of a wetland and improve the aesthetic value. Filter strips also reduce stream-bank cutting and provide access all year long. The effects of filter strips are captured in Figure 3.1.

3.2.3 Grassed Waterways

> *"A common mistake that people make when trying to design something completely foolproof is to underestimate the ingenuity of complete fools."- Douglas Adams*

Grassed waterways are naturally constructed channels used to carry the runoff from rigorous flow without causing soil erosion. They act as habitats for quails, pheasants, rabbits, and other wildlife. They provide additional surface water from natural drainage systems and reduce pesticides and other soluble nutrients in surface water. They also decrease sediments and other chemicals in surface water; thereby, increasing

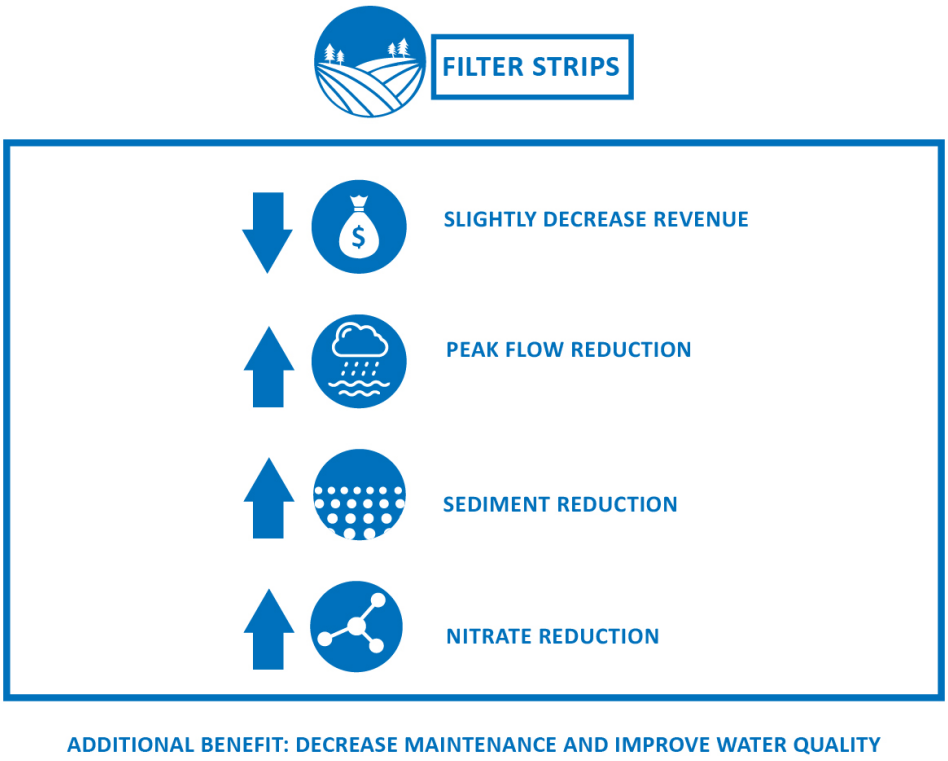

Figure 3.1.: Filter Strips

the dissolved oxygen for the growth of aquatic life. The effects of grassed waterways are captured in Figure 3.2.

3.2.4 Crop Rotation

> *"The rise of Google, the rise of Facebook, the rise of Apple, I think are proof that there is a place for computer science as something that solves problems that people face every day."- Eric Schmidt*

Crop rotation involves the planting of crops in a planned sequence, usually a period of just two to three years. It involves rotating food crops with various field crops. For example, barley can be grown after wheat, grain crops can be grown after pulses, etc. Crop rotation reduces runoff, erosion, and pests, and increases organic matter,

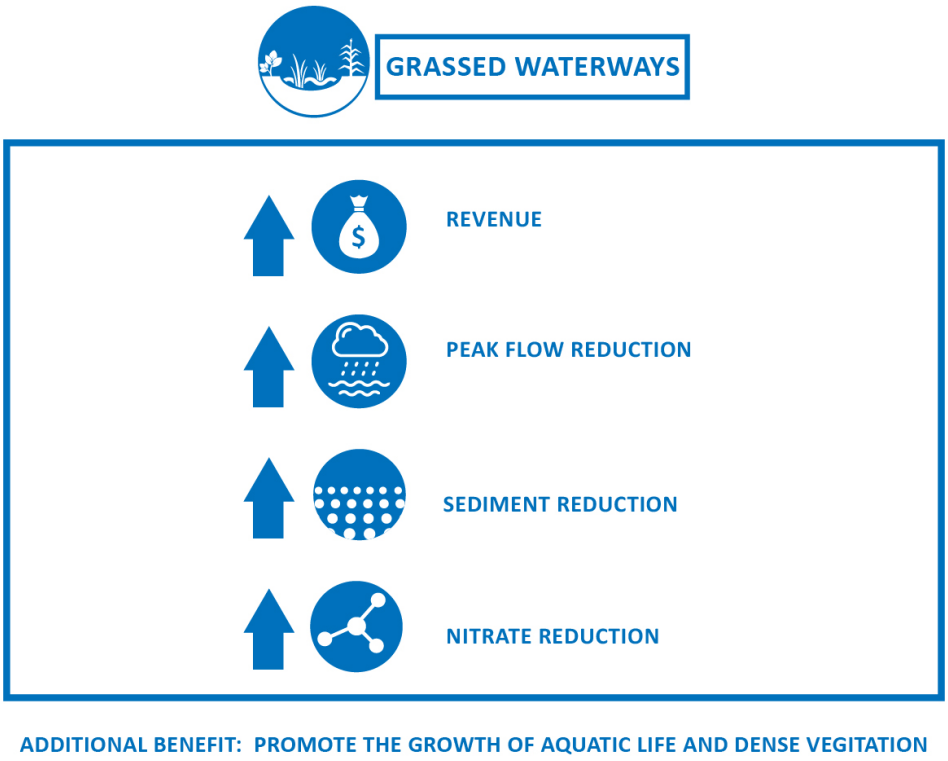

Figure 3.2.: Grassed Waterways

soil tilth, moisture efficiency, and yield. The effects of crop rotation are captured in Figure 3.3.

3.2.5 No-till

> *"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable."- Leslie Lamport*

No-till leaves stems, leaves, seed pods, etc. unhindered from harvesting through planting and can be beneficial with most crops if managed properly. No-till can reduce labor costs, greenhouse gases, and soil erosion. No-till increases organic matter in the soil and increases earthworm population, soil quality, and water infiltration. No-till

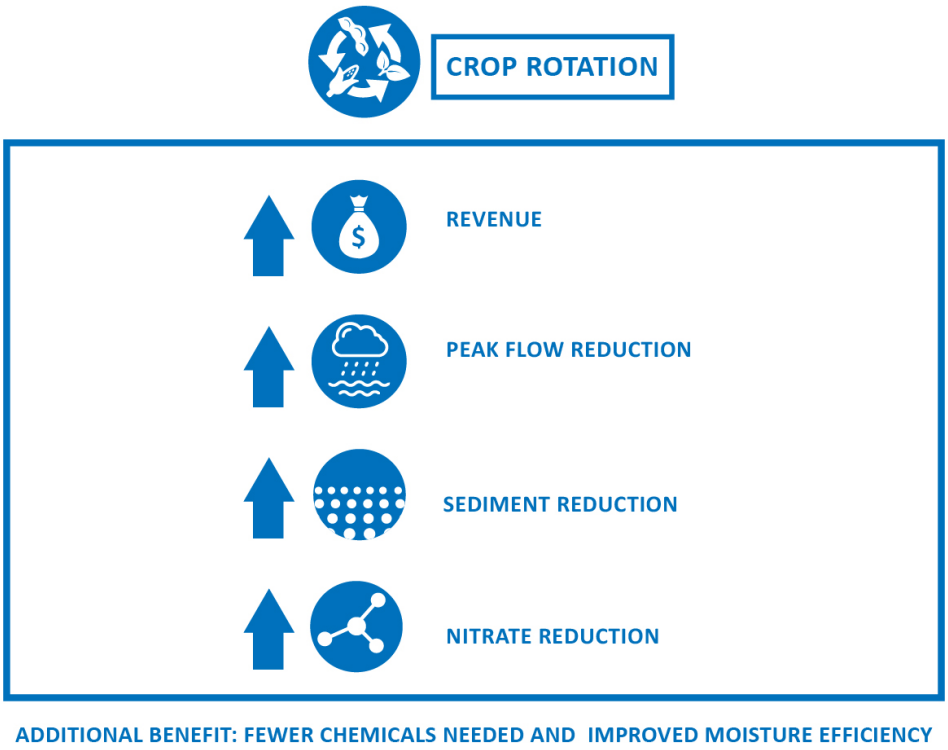

Figure 3.3.: Crop Rotation

reduces runoff due to increased organic matter and also optimizes soil moisture for crop growth in dry seasons. The effects of no-till are captured in Figure 3.4.

3.2.6 Strip Cropping

> *"Computer science is no more about computers than astronomy is about telescopes."- Edsger Dijkstra*

Strip cropping involves the systematic arrangement of strips across fields in order to lessen soil erosion by water and the wind. It is mainly used on recreational and farming land. Irregular strips of grass are replaced and strips are laid out with a correspondence to the slope of the land to help with erosion. To deal with wind erosion, strips are laid out perpendicularly to the dominant wind direction. Strip

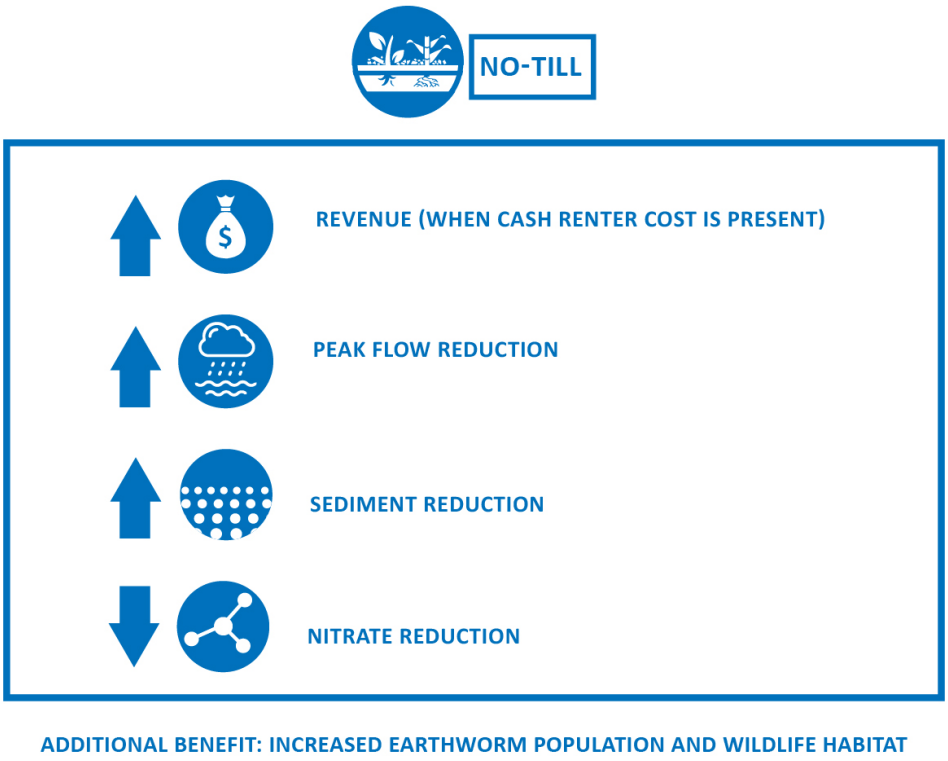

Figure 3.4.: No-till

cropping also reduces dust emissions in the air and increases soil moisture and water quality. The effects of strip cropping are captured in Figure 3.5.

3.2.7 Cover Crops

> *"Artificial intelligence is nowhere near attaining actual sentience or awareness. And without awareness it's simply a mechanical device, which may pretend to show emotions and sentience, if it is programmed to do so, and thus it may be able to fool the humans as being alive, but in its own internal circuitry, it'd simply be following its preprogrammed tasks through the flowchart of an algorithm."- Abhijit Naskar*

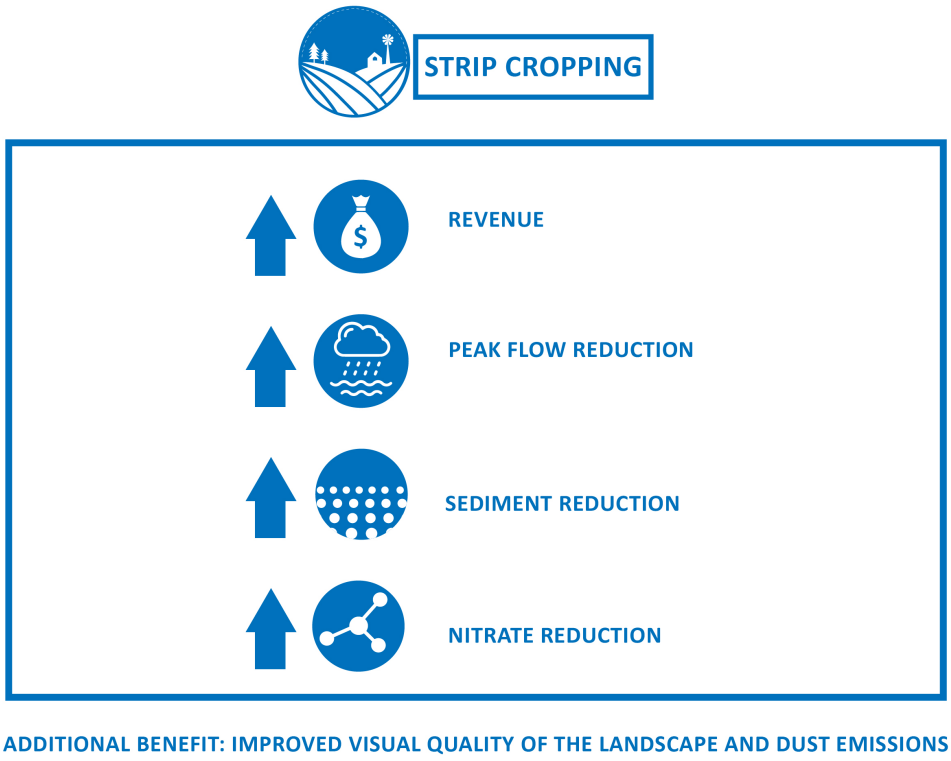

Figure 3.5.: Strip Cropping

Cover crops include rye-grass, crimson, oats, etc., and are grown between regular crops like corn and soybeans to protect the soil. Cover crops are especially useful during summer when primary crops are damaged. Cover crops improve soil structure by increasing organic matter and root penetration. They use nitrogen left in the soil and increase nutrient availability for corn, soybeans, and other crops. Cover crops also increase the biodiversity of soil and suppress weeds. The effects of cover crops are captured in Figure 3.6.

3.2.8 Conservation Programs

> *"People think that computer science is the art of geniuses but the actual reality is the opposite, just many people doing things that build on each other, like a wall of mini stones."- Donald Knuth*

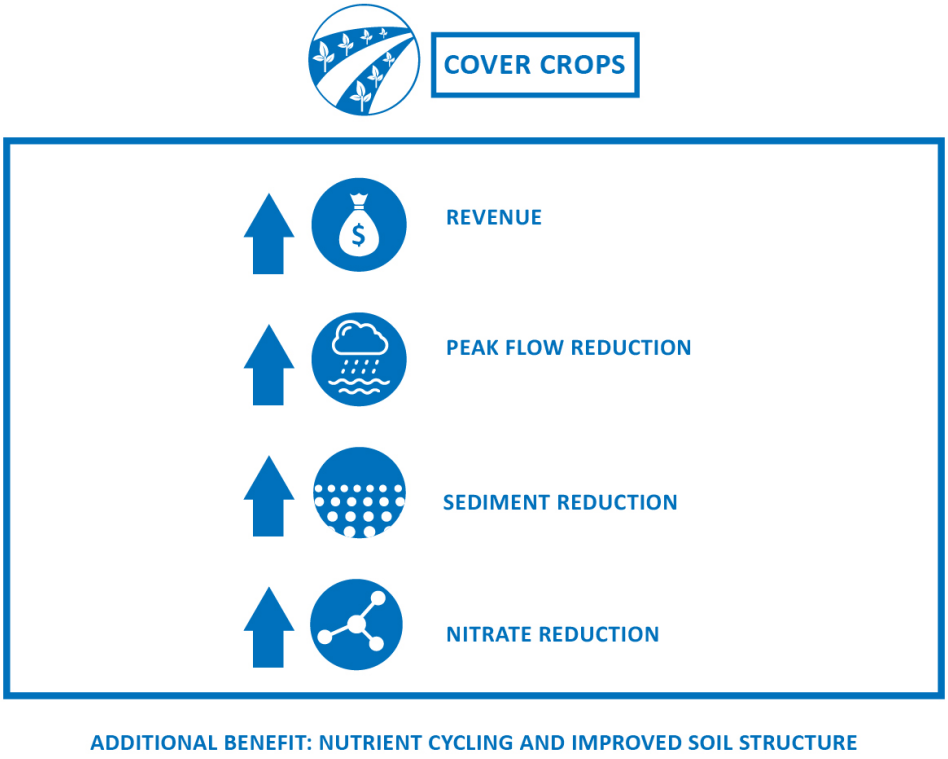

Figure 3.6.: Cover Crops

The United State Department of Agriculture (USDA) and Natural Resources Conservation Services (NRCS) have several programs for improving water quality and preventing erosion. Some of these related programs include

- Conservation Reserve Program (CRP): CRP assists landowners with cost assistance for preventing soil erosion.
- Environmental Quality Incentive Program (EQIP): EQIP assists landowners with planning, implementation, and maintenance of agricultural areas Wetlands Reserve Program (WRP): WRP assists landowners who converted their private wetlands to agricultural areas before 1985.

- Wildlife Habitat Incentives Program (WHIP): WHIP is a voluntary program for people who want to improve the wildlife habitat in areas not under any other program.
- Conservation of Security Program (CSP): CSP encourages producers to implement conservation activities on their land.

3.3 IGAMI2

> *"To my knowledge, nobody, no one who is publishing papers in the main field of AI, is even working on consciousness. I think there are some neuroscientists who are trying to understand it, but I'm not aware that they've made any progress. As far as AI people, nobody is trying to build a conscious machine, because no one has a clue how to do it, at all. We have less clue about how to do that than we have about build a faster-than-light spaceship."- Stuart Russell*

IGAMI2, the second version of the Interactive Genetic Algorithm with Mixed-Initiative, is a distributed algorithm used in our system to run environmental models, with feedback from stakeholders, to perform interactive multi-objective optimization. Feedback from stakeholders is used to train the virtual user models or simulated decision makers (SDMs). Feedback from stakeholders is fed into the interactive optimization process and the process loops iteratively until satisfactory results have been settled upon.

The IGAMI2 Kernel manages different components of the system and includes a Mixed-Initiative Manager (MIM), an SDM Manager (SDMM), an HPC Controller, an Optimization Manager (OM), and an Introspection Manager (IM). The database server is responsible as acting for a data layer for the entire system. Both the kernel and the database server run on the master computer.

The external view of the system involves stakeholders signing in through a web application and starting their search by specifying their preferences. The kernel initiates searches for each user with default designs being used for the first search. Stakeholders

provide ratings for generated designs, and then non-dominated sorting and an initial search begins. When evaluation finishes, results are given back to stakeholders for rounds of feedback. Interactions are saved in the database server for future reference and for learning in subsequent iterations.

The distributed system contains several machines with different roles. The system contains a high performance computing component which is used to run analytical environmental models for determining the quantifiable characteristics of different designs. The HPC controller runs on the same machine as the kernel, while virtual agents (VAs) run Soil and Water Assessment Tool evaluations on each of the designs. The capacity of the system has varied over time, but currently supports thousands of concurrent model evaluations running, allowing dozens of users to use the system at a time. The system has been designed in a way that it can be scaled to handle additional users and environmental model runs. More details about IGAMI2 and the components mentioned above are available in previous work.

The algorithm and system allow users to test competing designs in a simulated environment. Stakeholders can identify designs that are matched to their individual needs. An iterative search method is used so that stakeholders who wish to experiment can discover the impact of several different choices on the overall watershed. The tool is useful from several angles: driving stronger environmental outcomes, increasing understanding and awareness of the importance of watersheds in the community, and driving environmentally sound and consensus-driven decision-making.

4. COMPARISON OF NEURAL METHODS FOR LIMITED DATA USER MODELING IN WETLAND DESIGN

4.1 Abstract

The WRESTORE (Watershed REstoration using Spatio-Temporal Optimization of REsources) (http://wrestore.iupui.edu) online environmental decision support system allows stakeholders to design solutions for preventing or reducing pollution on a watershed landscape. This is accomplished by utilizing an interactive optimization algorithm that breaks down the problem-solving process into several stages. WRESTORE uses the Soil and Water Assessment Tool hydrologic model for simulating the impact of landscape decisions on watershed methods. WRESTORE is one of many interactive systems that changes in direct relation to the quality and quantity of user feedback data. The user modeling component employs a deep learning neural network approach. Limited data in user modeling is a common issue, one we will examine to improve the outcomes of our interactive optimization process. The innovation of our method is combining user models with insufficient data with application-specific augmentation. This chapter has three goals: i) explain the WRESTORE system, ii) describe current work in user modeling with insufficient data, and iii) describe our work in implementing specific virtual stakeholder models based on limited training samples.

4.2 Introduction

Even though researchers have debated the effectiveness of big data solutions, some problems are inherently restricted to limited-size datasets [18] [19] [20]. Small data appears in domains such as small enterprise solutions and black swan event modeling.

When processing small data, common problems can arise like how to deal with outliers and anomalies, over-fitting to training or test or validation data, and the cost of acquiring new data.

Possible strategies for dealing with limited data are to find experts familiar with small sample experiments in that field, to consult a statistician, to reduce the degrees of freedom, to reduce the number of possible hypotheses, or to reduce the models' complexity. More advanced techniques for dealing with reduced-size datasets include using pre-trained models, attempting to transfer models from other domains [21], finding application-specific ways to perturb and augment data [22], or creating synthetic minority class examples [23].

When working with small data, it is important to keep certain techniques in mind, such as: data cleansing, augmenting data with external sources, removing irrelevant characteristics, and being aware of the uncertainty associated with the predictions and communicating this uncertainty to the intended audience. Reporting uncertainty is important because stakeholders may incorrectly place too much faith in the system.

The WRESTORE web application helps communities of stakeholders to get user-centric solutions for the allocation of conservation practices in a watershed landscape. User models, interactive optimization problem formulation, and the distributed hydrology model based on the USDA's Soil and Water Assessment Tool [24] were built for the entire test bed site of Eagle Creek Watershed. The system identifies the optimal locations to apply best management practices in the watershed.

4.2.1 Related Work

Corani developed an alternative Naive Bayes approach [25] which allows a general and flexible treatment of incomplete data, while others [26] have proposed a context tree generalization because it provides more structural flexibility. Researchers at Intel [27] used an A-S learning process and the guard-band concept to achieve a sample size reduction of 10 to 20 times. Onisko proposed a method [28] that uses

Noisy-OR gates to reduce the data requirements in learning conditional probabilities. Shaikhina has worked with multiple runs of neural networks and decision trees to refine accuracy on small datasets in the medical domain [29].

In order to identify and mitigate outliers without removing relevant data, some researchers have dealt with limited data in high-dimensional feature spaces using ordered if-then-else lists of rules [30]. Other approaches have experimented with the adaptive local hyperplane algorithm on small medical datasets [31]. Another technique used data-fuzzifying, domain range expansion, and adaptive-network-based fuzzy inference systems (ANFIS) to handle limited-size scheduling datasets [32]. In another study, researchers used a mega-trend-diffusion technique to estimate the domain range of a small dataset and produce artificial samples for training artificial neural networks [33].

In a 2014 paper, researchers constructed virtual samples using mega-trend diffusion functions and using back-propagation neural networks [34]. Researchers have also discussed several techniques including attribute construction, the bootstrap method, the incremental method, and different diffusion functions [35] and concluded that data augmentation can boost classification accuracy. Other researchers have studied human face recognition and proposed perturbations and mutations to create additional test data when datasets are limited [36]. Simulation-based data has been applied to reinforcement learning-based approaches with success as well [37].

Some researchers have created new learning samples to fill gaps in their original datasets and used a neural network learning method based on the posterior probability [38]. One study compared the learning surfaces of several algorithms in an attempt to discover which feature selection methods and models should be used [39], and found that careful visualization of the learning surface's different regions is key to discovering the correct combination. A budget-sensitive progressive sampling algorithm for selecting points from the training set to deal with class imbalance has also been proposed [40]. Meignan presented a review of interactive optimization includ-

ing interactive reoptimization, interactive multi-objective optimization, interactive evolutionary algorithms, and other approaches [41].

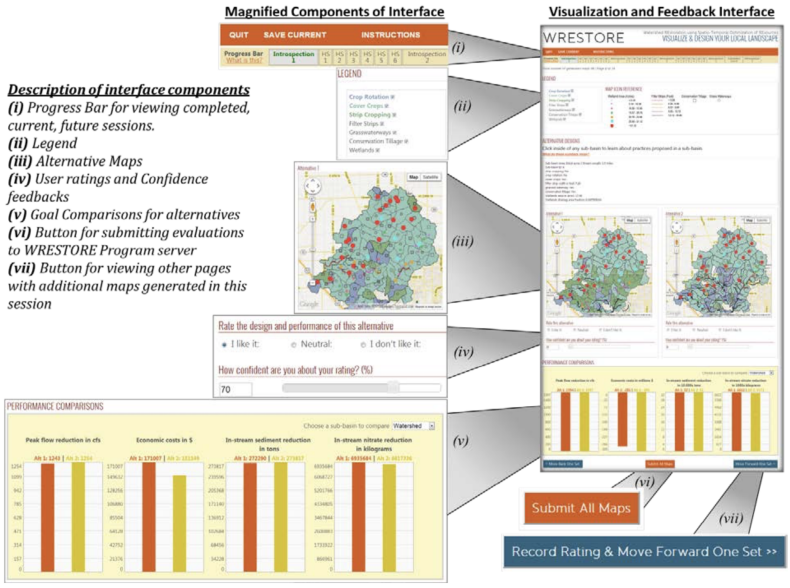

Figure 5. Visualization and Feedback Interface in WRESTORE

Figure 4.1.: WRESTORE system

Figure 4.1 shows part of the human-computer interface of the WRESTORE system. Since the human brain is skilled at visualizing data for analysis [42], it is only reasonable to utilize human-guided search based on interactive optimization. Tasks in consulting positions should be performed by humans when the problems require visualization of large search spaces [43]. However, one common limitation with this approach is user fatigue [44] which can be addressed using a simulated decision maker (SDM) based on neural networks and other linear/non-linear modeling techniques. Because of the small number of inputs that a typical user can provide, training SDMs is a limited-data problem.

4.3 Methodology

We examined three predictive models for user modeling: neural networks, fuzzy logic, and deep learning.

4.3.1 Neural Networks

Our artificial neural network uses a feed-forward design defined by inputs sent through hidden layers to a final output layer [45]. The general neural network design is depicted in Figure 4.2. The input layer is comprised of wetlands design data and our output layer is the prediction of the user model based on the input.

The activation state of each unit within each layer, X_i, was given by a value of 0 for not activated and a value of 1 for activated. The weight given between unit i and unit j is given by weight W_{ij}. The activation of a unit is given by the sum of the products of output Y_j and weight W_{ij} along with the bias term, b_j:

$$X_i = \sum_j W_{ij} Y_j + b_j.$$

Given activation X_i, output Y_i is computed using the activation function. We used the logistic sigmoid function which takes the activation state X_i for each unit to generate output Y_i given by

$$Y_i = \frac{1}{1 + e^{-X_i}}.$$

Output Y_i is propagated in a feedforward fashion to the next layer until it reaches the output layer.

Pseudo-code for creating one NN configuration in Matlab is found below:

Adaptive Neuro-fuzzy Inference System

Fuzzy logic is based upon Zadeh's theory of fuzzy sets [46]. An adaptive neuro-fuzzy inference system is based on a Takagi-Sugeno fuzzy inference system. The technique comprises integrating fuzzy logic and artificial neural networks. We trained our system using a hybrid algorithm with a forward and backward pass. The least-squares method finds consequent parameters in the forward pass; gradient descent optimizes error values and premise parameters in the backward pass [47]. The ANFIS architecture is depicted in Figure 4.3.

Algorithm create Neural Net Classifier

procedure createNeuralNetClassifier($trainData, testData, hiddenLayerSize, numEpochs, maxFails$)
    $numTrainingDimensions = length(trainData(:,1)) - 1$
    $trainingDataPoints = trainData(2 : numTrainingDimensions, :)$
    $trainingLabels = trainData(1, :)$
    $testDataPoints = testData(2 : numTrainingDimensions, :)$
    $testLabels = testData(1, :)$
    $trainingLabelPoints = length(trainingLabels)$
    $numberOfTrainingLabels = max(trainingLabels)$
    $trainingLabelMatrix = zeros(numberOfTrainingLabels, trainingLabelPoints)$
    $testLabelPoints = length(testLabels)$
    $numberOfTestLabels = numberOfTrainingLabels$
    $testLabelMatrix = zeros(numberOfTestLabels, testLabelPoints)$
    for $(i = 1 : trainingLabelPoints)$ **do**
        $trainingLabelMatrix(trainingLabels(i), i) = 1$
    end for
    for $(i = 1 : testLabelPoints)$ **do**
        $testLabelMatrix(testLabels(i), i) = 1$
    end for
    $net = patternnet()$
    $net = patternnet(hiddenLayerSize)$
    $net.divideParam.trainRatio = 75/100$
    $net.divideParam.valRatio = 15/100$
    $net.divideParam.testRatio = 15/100$
    $net.trainParam.epochs = numEpochs$
    $net.trainParam.show = 5$
    $net.trainParam.min_grad = 1e-8$
    $net.trainParam.max_fail = maxFails$
    $net.trainParam.sigma = 5.0e-7$
    $net.trainParam.lambda = 5.0e-9$
    $net.performFcn = 'mse'$
    $net.performParam.regularization = 0.01$
    $net = train(net, trainingDataPoints, trainingLabelMatrix)$
    $outputdata = net(testDataPoints)$
    $perf = mse(net, testLabelMatrix, outputdata)$
    $mserror = perf$
    $classes = vec2ind(outputdata)$
    $bestnet = net$
    $errorsSet = perf$
    $errorAbsNN = perf$
end procedure

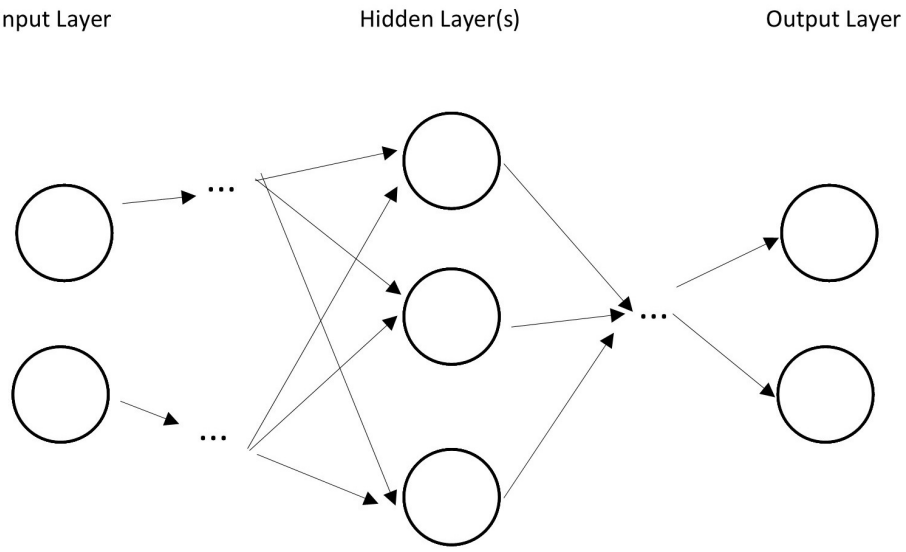

Figure 4.2.: NN Design

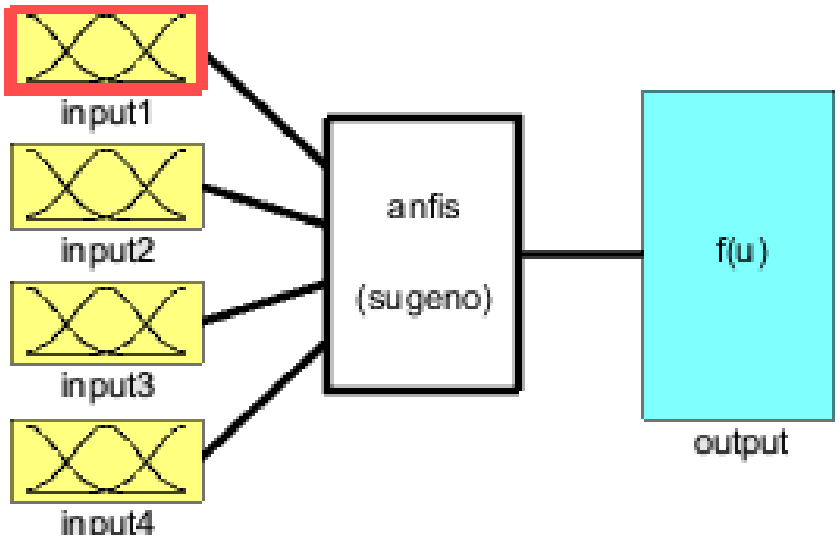

Figure 4.3.: ANFIS architecture

The output of our fuzzy logic process can be written as

$$
\begin{aligned}
f &= \frac{w_1}{w_1+w_2}f_1 + \frac{w_2}{w_1+w_2}f_2 \\
f &= \overline{w}_1(p_1x+q_1y+r_1) + \overline{w}_2(p_2x+q_2y+r_2) \\
f &= (\overline{w}_1x)p_1 + (\overline{w}_1y)q_1 + (\overline{w}_1)r_1i + \\
&\quad (\overline{w}_2x)p_2 + (\overline{w}_2y)q_2 + (\overline{w}_2)r_2
\end{aligned}
$$

where f is assumed to be linear over the parameters p_i, q_i, r_i.

Pseudo-code for creating one ANFIS configuration in Matlab is found below:

Algorithm creating one ANFIS configuration in Matlab

procedure CREATEADAPTIVENEUROFUZZYINFERENCESYSTEM($trainData, testData, epochs, targetError, stepSize$)
  $cols = length(trainData(1,:))$
  $rows = length(trainData(:,1))$
  $temp = trainData(:,1)$
  $trainData(:,1) = trainData(:,cols)$
  $trainData(:,cols) = temp$
  $minimumTrainClass = min(trainData(:,cols))$
  $maximumTrainClass = max(trainData(:,cols))$
  $trainData(:,cols) = (trainData(:,cols) - minimumTrainClass)/ (maximumTrainClass - minimumTrainClass)$
  $temp = testData(:,1)$
  $testData(:,1) = testData(:,cols)$
  $testData(:,cols) = temp$
  $testData(:,cols) = (testData(:,cols) - minimumTrainClass)/ (maximumTrainClass - minimumTrainClass)$
  $mfType =' gbellmf'$
  $trnOpt = [epochstargetErrorstepSize0.71.1]$
  $in\_fis = genfis1(trainData, 4, mfType, mfType)$
  $outfis = anfis(trainData, in_fis, trnOpt)$
  $[anfisPredictions] = evalfis(testData(:,1:(cols-1)), outfis)$
  $anfisPredictions = (anfisPredictions * (maximumTrainClass - minimumTrainClass)) + minimumTrainClass$
  for $i = 1 : length(anfisPredictions)$ **do**
    if $(anfisPredictions(i) < minimumTrainClass)$ **then**
      $anfisPredictions(i) = minimumTrainClass$
    end if
    if $(anfisPredictions(i) > maximumTrainClass)$ **then**
      $anfisPredictions(i) = maximumTrainClass$
    end if
    $anfisPredictions(i) = round(anfisPredictions(i))$
  end for
  $testData(:,cols) = (testData(:,cols) * (maximumTrainClass - minimumTrainClass)) + minimumTrainClass$
  $testData(:,cols)$
  $anfisPredictions$
  $anfisRmse = sqrt(sum((testData(:,cols) - anfisPredictions(:)).^2)/ numel(testData(:,cols)))$
end procedure

Adaptive neuro-fuzzy inference systems use a neural network based on a Takagi–Sugeno fuzzy inference system. The technique involves the integration of techniques from both fuzzy logic as well as artificial neural networks. Our system is

trained using a forward and backward pass in a hybrid algorithm. The least-squares method finds consequent parameters in the forward pass; error values and premise parameters are optimized with gradient descent in the backward pass.

4.3.2 Deep Learning

Deep neural networks (i.e., neural networks with over one hidden layer) are used for mail classification [48]. Popular approaches with deep neural networks learn several layers of representation [49].

Deep neural networks, neural networks with more than one hidden layer, have been used successfully with several use cases. There has been much work on dealing with training sets of constrained size, some of which we adapted here, primarily through data cleaning and augmentation. Two very important things for researchers and practitioners to do is to make sure they validate the integrity of the design of their experiments and make sure they do not lose any valid noise which is present in the data. Comparing the learning surfaces implementations is another way to boost the understandability and accuracy of the models which are utilized.

Pseudo-code for creating one DL configuration in Matlab is found below:

Algorithm create Deep Data Sets

procedure CREATEDEEPDATASETS(data)
    $X = getDeepNormalizedWithRank(data)$
    $c = length(data(1,:))$
    $data1 = zeros(120, c)$
    $data2 = zeros(120, c)$
    $data3 = zeros(120, c)$
    $data4 = zeros(240, c)$
    $data5 = zeros(360, c)$
    for $i = 1 : 120$ **do**
        $data1(i,:) = X(i,:)$
    end for
    for $i = 121 : 240$ **do**
        $id = i - 120$
        $data2(id,:) = X(i,:)$
    end for
    for $i = 241 : 360$ **do**
        $id = i - 240$
        $data3(id,:) = X(i,:)$
    end for
    $data4 = [data1; data2]$
    $data5 = [data4; data3]$
    $dataDeep = zeros(length(X(:,1)), c - 1)$
    for $i = 1 : c - 1$ **do**
        $dataDeep(:,i) = X(:,i)$
    end for
end procedure

Algorithm Analyze Data Deep

procedure AnalyseDataDeep($dataDeep, X$)
  $nout = max(nargout, 1) - 1$
  $errorsSetDeep = zeros(7, 1)$
  $pos = length(X(1, :))$
  $len = length(X(:, 1))$
  $targetNN = zeros(len, 3)$
  $target = zeros(len, 1)$
  $traindata = zeros(len, pos - 1)$
  for $i = 1 : len$ **do**
    $posRank = X(i, pos)$
    if then$(posRank == 0)$
      $posRank = 1$
    end if
    $targetNN(i, posRank) = 1$
    $target(i) = posRank$
  end for
  for $i = 1 : pos - 1$ **do**
    $traindata(:, i) = X(:, i);$
  end for
  $inputData = transpose(traindata)$
  $targetData = transpose(target)$
  $testData = inputData$
  $testDataRnk = targetData$
  $targetNN = transpose(targetNN)$
  $dataDeep = transpose(dataDeep)$
  $SOMNets = cell(7, 1)$
  $DeepNets = cell(7, 1)$
  $SOMNodes = zeros(7, 1)$
  $SOMNodes(1, 1) = 10$
  $SOMNodes(2, 1) = 25$
  $SOMNodes(3, 1) = 50$
  $SOMNodes(4, 1) = 100$
  $SOMNodes(5, 1) = 200$
  $SOMNodes(6, 1) = 250$
  $SOMNodes(7, 1) = 500$
  $trialNos = 10$
  for $i = 1 : 7$ **do**
    $[SOMNets\{i\}] = createSOM(dataDeep, SOMNodes(i))$
    $varargout\{i + 7\} = SOMNets\{i\}$
    $[rankData\{i\}] = createDeepNeuralNetClassifier(inputData, targetNN, testData, testDataRnk, SOMNets\{i\}, trialNos)$
  end for
  for $i = 1 : 7$ **do**
    $varargout\{i\} = rankData\{i\}$
  end for
end procedure

Algorithm create Deep Neural Net Classifier

```
1: procedure createDeepNeuralNetClassifier(inputData, targetData, testData, testDataRnk, nnsom, trialNos)
2:     trainPer = 75
3:     valPer = 15
4:     testPer = 10
5:     len = length(targetData)
6:     bestnet = 0
7:     errorsmin = length(inputData(1, :))
8:     errorsSet = zeros(trialNos, 1)
9:     x1 = nnsom(inputData)
10:    for k = 1 : trialNos do
11:        net = patternnet()
12:        net.trainParam.max_fail = 200
13:        net.divideParam.trainRatio = trainPer/100
14:        net.divideParam.valRatio = valPer/100
15:        net.divideParam.testRatio = testPer/100
16:        net.trainParam.showWindow = false
17:        [net] = train(net, x1, targetData)
18:        x2 = nnsom(testData)
19:        outputdata = net(x2)
20:        len = length(x1(1, :))
21:        rankdata = zeros(1, len)
22:        for i = 1 : len do
23:            maxx = 0
24:            for j = 1 : 3 do
25:                if (maxx < outputdata(j, i))) then
26:                    maxx = outputdata(j, i)
27:                    rankdata(1, i) = j
28:                end if
29:            end for
30:        end for
31:        errorRMS = 0
32:        for i = 1 : len do
33:            val = testDataRnk(i) − rankdata(i)
34:            errorRMS = errorRMS + abs(val)
35:        end for
36:        errorRMS = errorRMS/len
37:        errorRMS = errorRMS/2
38:        if (errorRMS < errorsmin) then
39:            bestnet = net
40:            errorsmin = errorRMS
41:        end if
42:        errorsSet(k) = errorRMS
43:    end for
44: end procedure
```

Deep learning models are similar to the representation of neural networks which was already presented. For representing deep learning models, we simply have two or more layers of nodes or neurons $N = \{u_1, u_2, \ldots, \}$ and a finite set $H \subseteq N \times N$ of directed edges or connections between nodes which we represent as an acyclic graph. The i-th layer ($i > 1$) is the set of all nodes $u \in N$ such that there is an edge path of length $i - 1$ (but no longer path) between some input unit and u where shortcuts may exist from a previous layer to a subsequent layer. The model's behaviour is determined by a set of real-valued weights w_i ($i = 1, \ldots, n$) which are determined empirically by the context of the learning problem presented.

4.3.3 Experimental setup

Our data came from SQL Server instances which are further described in the appendix. In a previous workshop, 20 participants volunteered to find watershed-scale plans that agreed with their individual subjective preferences, including 14 from Indiana University and Oregon State University and six stakeholder users consisting of state/federal agency personnel, nongovernmental organization personnel, and watershed individuals.

One limitation of the approach is that the 14 stakeholders are not personally involved in the watershed, but all of them are from a community which is directly impacted by the watershed. To address this, several stakeholders more closely associated with Eagle Creek Watershed via land ownership or professional responsibilities were also included in the workshop. Several challenges were encountered in testing and implementing novel decision support tools in real-world conditions, some of which will be addressed through the remainder of the dissertation.

We preprocessed the data to remove near zero variance users and users who showed negative or small correlation with one or more of the objective fitness functions previously described. In addition, we also created synthetic user data with known behaviors

and preferences to validate that our models would have as expected when given users with known goals.

4.3.4 Inputs and outputs of the models

Four physically based environmental objective functions (Peak Flow Reduction, Sediment Reduction, Nitrates Reduction, and Cost Revenue function) were estimated by the Soil and Water Assessment Tool at the entire watershed scale and at the local level for each sub-basin, these acted as the input for each model.

Peak flow reduction represents is the maximum difference between the peak flows of the baseline model and the peak flow of the new model which has been found, it represents a decrease in flooding for the watershed. Sediment Reduction represents the loss of fertile soil from the landscape across all sub-basins. Nitrate Reduction represents loss in nitrates via runoff across all sub-basins. Finally, the Cost Revenue function considers the costs and revenues generated by the conservation practice over model time period.

The output layer here consists of the predicted user rating for a virtual user model or simulated decision maker. The architecture and configuration of each model (size of the hidden layers, regularization parameters, etc.) was determined empirically through hyperparameter optimization.

4.4 Results

The performance of a linear model, neural networks, fuzzy logic, and deep learning for the user are compared with RMSE:

$$\text{Root Mean Square Error} = \sqrt{\frac{\sum \left(P_{\text{est}} - P_{\text{true}}\right)^2}{n}}.$$

The neural network obtained the results in Table 4.1.

Fuzzy logic obtained the results in Table 4.2.

Deep learning obtained the results in Table 4.3.

Table 4.1.: Performance of neural network (non-enriched)

Sample size	RMSE
N=25	0.23
N=50	0.07
N=100	0.15
N=360	0.08

Table 4.2.: Performance of fuzzy logic (non-enriched)

Sample size	RMSE
N=25	0.40
N=50	0.30
N=100	0.28
N=360	0.12

Table 4.3.: Performance of deep learning (non-enriched)

Sample size	RMSE
N=25	0.58
N=50	0.14
N=100	0.12
N=360	0.11

Table 4.4.: Performance of deep learning with enriched data

Sample size	RMSE
N=25 (1500 total)	0.11
N=50 (3000 total)	0.094
N=100 (6000 total)	0.085
N=360 (12000 total)	0.079

Deep learning with enriched training data obtained the results in Table 4.4.

Many complexities appeared during testing with stakeholders, including limited data points, significant noise in experiment data, and the variety of preferences present among the stakeholders. The holdout test data used in our validation process confirmed that our data augmentation technique offered a measurable improvement over standard techniques. The holdout test data did not go through the same augmentation technique as the incoming training data. We chose RMSE as our objective function because of its prevalence in environmental systems analysis.

RMSE is helpful in measuring the overall fit between the predicted and actual data; however, since RMSE only considers the magnitude of an error without considering the potential human cost it is ineffective at dealing with outliers and other issues. Although it was susceptible to noise, the traditional neural network performed the best without enriched data. The deep learning approach behaved accurately and consistently when adding the enriched data. Additional work that remains includes finding additional ways to deal with limited data in user modeling and integrating these results into the interactive optimization process.

4.5 Conclusion

Complex techniques can produce incorrect results due to issues coming from limited size datasets. Future areas of work include finding additional domain-specific ways to augment data, additional validation and verification, dealing with untrustworthy stakeholders, providing a way to capture uncertainty, and providing transparency into the prediction space.

5. FUZZY AND DEEP LEARNING APPROACHES FOR USER MODELING IN WETLAND DESIGN

5.1 Abstract

Several factors must be considered in the process of designing optimal watersheds. Some of these criteria come from community stakeholders that provide feedback on different designs. The Watershed Restoration Using Spatial Temporal Optimization of Resources (WRESTORE) project, at (http://wrestore.iupui.edu), allows members of the community to be a part of the watershed design process. This chapter examines the performance of several different user modeling techniques that have limited data to train upon. A deep learning user model is integrated back into our interactive optimization process to drive improve watershed design outcomes.

5.2 Introduction

WRESTORE utilizes the Soil and Water Assessment Tool hydrological model for its watershed simulations. Most of the controls are done in a virtual, interactive genetic algorithm, with deep learning user models providing input into the process. The existing user modeling component was redeveloped with additional capacity for dealing with limited data. This chapter explains our work implementing reliable and stable predictive models for the purpose of boosting system efficiency and improving interactive optimization performance for stakeholders.

5.3 Related Work

In previous work, our group used user modeling techniques and an interactive genetic algorithm to solve a multi-objective optimization problem [50]. Other work has

found it is possible to use domain experts to capture qualitative information which otherwise may not be available, while visualization can lead to a more globally optimal solution [51] [52]. Different implementations of user modeling exist [53] [54] [55] [56], some even to solve water quality modeling problems. Some of these approaches used a web interface to reach out to a wider range of stakeholders [57]. Genetic algorithms have been used to search exponential search spaces where other optimization techniques traditionally failed due to the complexity of the search space [58] and graphical representations also have a history of usage in interactive genetic algorithms [59].

Kisi used fuzzy logic and neural networks to estimate sediment concentration [60]. Researchers have found that neural networks are regarded as good models for expressing nonlinear decisions [61]. The use of neural network variants for user modeling and adaptive learning is well studied [54] [62] [63], and it also has been used in hydrology and multi-criteria decision making [64] [65] [66]. Previous work in WRESTORE also showed significant peak flow reductions with fewer sites and smaller wetlands via a GIS-simulation optimization based approach [67].

In order to perform optimization in a time efficient manner, distributed computing was utilized to adopt a divide and conquer approach. A decision support system was used due to the complexity of the problem space (i.e., environmental issues, planning, and design are difficult tasks considering the number of conflicting and/or subjective issues that can arise with different key stakeholders), an area where decision support systems traditionally show success [68]. The watershed design dilemma directly relies on space, which has an exponentially large number of possible solutions. Human-guided search is a known method of solving problems which incorporates a feature space that is not easily modeled via mathematical equations. Problems with large data search spaces involving real-time visualization of the problem are typically best managed when humans serve an advisory role. A depiction of a design shown to stakeholders is present in Figure 5.1.

Interactive optimization is performed using Simulated Decision Makers (SDMs), which act as virtual users that model actual users' behaviors and preferences. Inter-

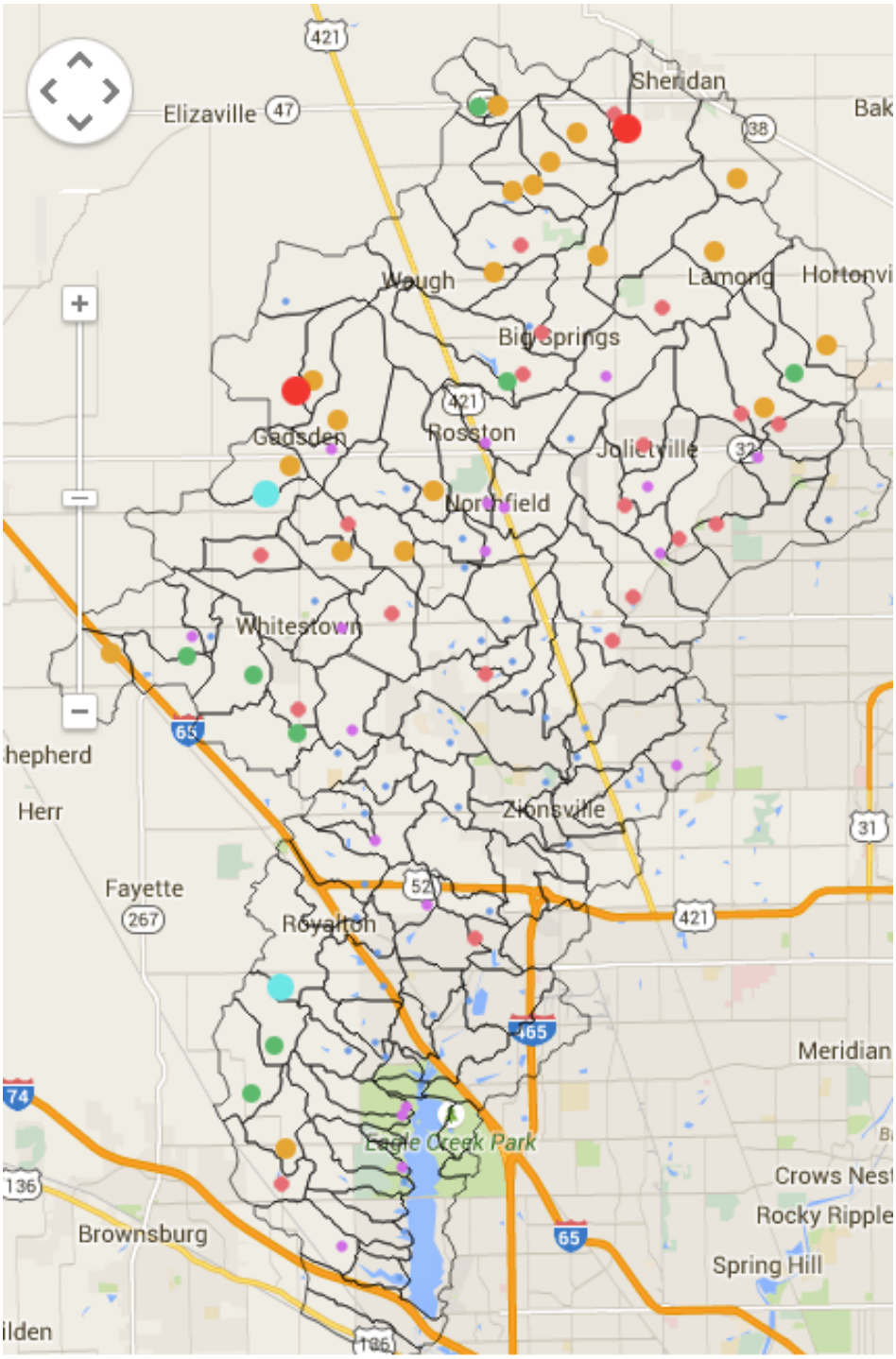

Figure 5.1.: Water basins in WRESTORE

active genetic algorithms are a modified genetic algorithm technique that we elected to use to take account of user preferences in the design process due to previous successes [69]. Neural networks, fuzzy logic, and deep learning were applied in the user modeling task of the watershed design optimization. Artificial neural networks (ANNs) are inspired by biological neural networks and are known to be universal approximators. Fuzzy logic is based upon classical logic theory but allows for a multiple-valued logic for degrees of certainty rather than a definite true or false. Deep learning usually involves layers of representation learning [46]. After considering several modeling procedures, multi-layer neural networks proved to provide the best virtual user models. A system that uses the scalable computing power of distributed computing was utilized [70].

5.4 Methodology

We examined three predictive models for user modeling: neural networks, fuzzy logic, and deep learning.

5.4.1 Artificial Neural Networks

Our neural network user model was tested with a variety of feedforward design which transferred signals on the input layer to the hidden layer and then to the output layer.

The input layer is composed of output from our SWAT models. The output unit layer represents the prediction made by the neural network. The particular activation state of each unit X_i in a layer is defined as a value between 0 for not activated and 1 for fully activated. The weight between a unit j and another unit i is represented by weight W_ij. The activation of a unit is given by the sum of the products of each unit's output Y_j and weight W_ij along with a constant bias term b_j.

The activation state X_i for each unit and the corresponding output value for a unit Y_i is computed using an activation function. Many such activation functions

are available including identity functions, binary step functions, ramp functions, sigmoidal functions, and radial basis functions. One popular activator is the logistic sigmoid function which takes the activation state X_i for each unit in order to generate the output by which Y_i is given. The output Y_i may then be propagated in a feedforward fashion to the next layer of the neural network, whether it is a hidden layer or an output layer. This process is depicted in Figure 5.2. The pseudo-code for the neural network here matches the pseudo-code which is found in Chapter 4.

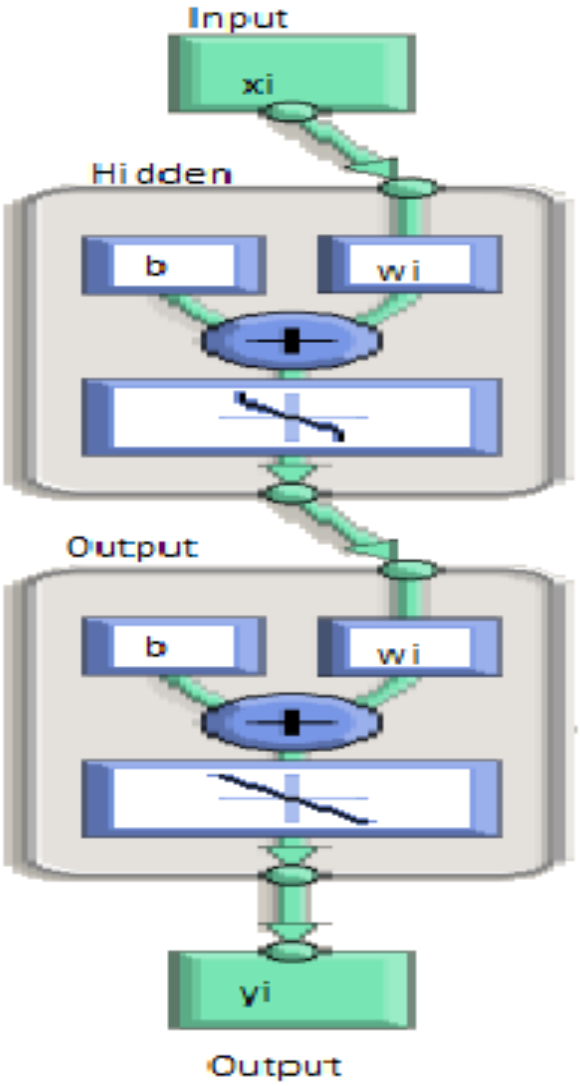

Figure 5.2.: Feedforward network

5.4.2 Fuzzy Logic

Fuzzy logic extends classical logic theory with degrees of certainty rather than a certain true or false output value. Adaptive neuro-fuzzy inference systems have been used before in ecology and water resource management problems [71]. Our ANFIS implementation used a Kalman filter and gradient descent algorithm [72]. Adaptive neuro-fuzzy inference systems are a type of fuzzy logic that involve the integration of methods from both fuzzy logic and artificial neural networks.

5.4.3 Deep Learning

Deep learning is a structured platform of learning that usually involves methods based on learning representations of data. These models use several non-linear transformations. Several researchers have used deep learning with different implementations and different use cases [73] [74] [75]. In our implementation, a multilayer feedforward artificial neural network was used with back-gradient propagation. Pseudo-code for the deep learning hyperparameter optimization is found below:

5.4.4 Experimental setup

As previously discussed, data came from SQL Server instances described in the appendix. 20 participants volunteered to find watershed-scale plans that agreed with their individual subjective preferences. One limitation of the approach is that 14 of the stakeholders were not personally involved in the watershed, but all of them are from a community which is directly impacted by the watershed. To address this, several stakeholders more closely associated with Eagle Creek Watershed via land ownership or professional responsibilities were also included in the workshop.

Algorithm Pseudo-code for the deep learning hyperparameter optimization

```
hiddenSizes ← c(1, 10, 30, 50, 70, 120, 160, 200, 400, 800)
dropoutRatios ← c(0.01, 0.05, 0.1, 0.2, 0.3, 0.4)
l1s ← c(1e − 8, 5e − 8, 1e − 7, 5e − 7, 1e − 6, 5e − 6, 1e − 5, 5e − 5, 4e − 8)
numEpochs ← c(1, 10, 20, 40, 100, 200)
i=0
for hiddenSize in hiddenSizes do
    for dropoutRatio in dropoutRatios do
        for l1Value in l1s do
            for epoch in numEpochs do
                r ← NULL
                attempt ← 1
                i = i + 1
                while is.null(r) && attempt ≤ 7 do
                    attempt ← attempt + 1
                    tryCatch(
                    r ← deep_train_may_fail(hiddenSize, dropoutRatio, l1Value, epoch, i)
                    error = function(e)print(e)
                    )
                end while
            end for
        end for
    end for
end for
```

5.4.5 Inputs and outputs of the models

As previously discussed, our physically based environmental objective functions (Peak Flow Reduction, Sediment Reduction, Nitrates Reduction, and Cost Revenue function) were estimated by the Soil and Water Assessment Tool at the entire watershed scale and at the local level for each sub-basin, these acted as the input for each model. The output layer here consists of the predicted user rating for a virtual user model or simulated decision maker. The architecture and configuration of each model (size of the hidden layers, regularization parameters, etc.) was determined empirically through hyperparameter optimization.

5.5 Results

5.5.1 Discussion of results

Several experiments were carried out with virtual users of the system. Because our feature space was already reduced, we did not use an autoencoder [76]. A grid search was performed over the number of nodes in the hidden layers, the dropout ratio used, the L1 regularization value, and the number of epochs used in training, in order to obtain maximum performance. Complementary random search may have allowed for increased performance on optimization of the objective function [77].

There are several different activator functions available to choose from. In this case, we make use of the rectifier which is given by f(i) = max(o,i) because we believe it to be closer to a biological process [78]. Test users were given a pre-defined goal for optimization, and some were machine-written virtual users that were given a pre-defined goal for optimization. Both of these cases were shown to be trivial cases to learn. Dropout randomly removed the values of certain nodes in the network to act as a form of regularization [79]. The data gathered was split into 240 known training designs and 120 test designs for evaluation. Root mean square was used as the objective function.

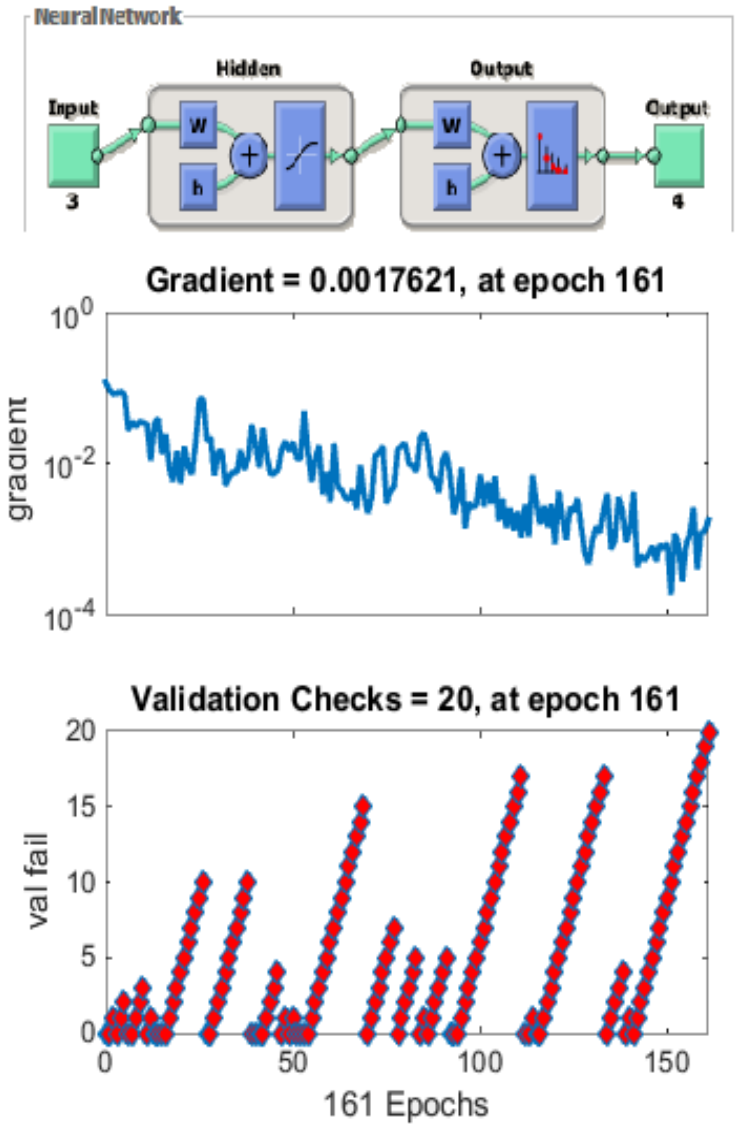

Figure 5.3.: NN Results

The network was trained with backpropagation using a hybrid approach of learning rate annealing and momentum training and may be found in Fig. 5.3. As was expected, actual users of the system were the most difficult category of prediction; on the other hand, the users who were given a predefined goal for optimization were the easiest category of prediction. The performance on a trivial virtual user may be found in Fig. 5.4.

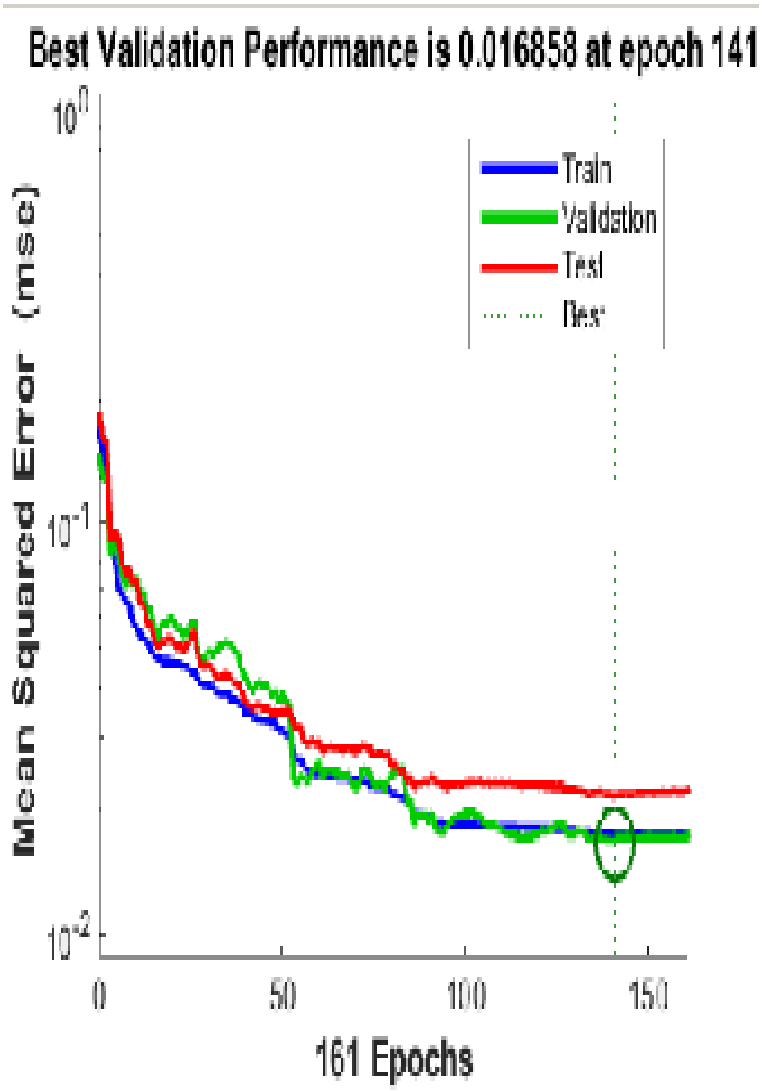

Figure 5.4.: Validation Performance

The goal of the overall WRESTORE system is to act as a web-based collaborative problem-solving tool to help stakeholders design and analyze conservation practices on the Eagle Creek Watershed. Genetic algorithms are used to search exponential search spaces where other optimization techniques traditionally fail [80]. The system offers several goals which stakeholders may optimize towards: decreasing the overall net costs, decreasing the overall flooding, decreasing the overall fertilizer losses, and

decreasing the overall erosion losses. Integrating user models back into the interactive genetic algorithm increased user satisfaction while also increasing other fitness functions. Additional work in this area would deal with using several stakeholders at once, or stakeholders with conflicting satisfaction goals or stakeholders who are not trustworthy.

5.6 Conclusion

WRESTORE helps stakeholders design the mixture of landscape practices that minimize soil erosion, fertilizer loss, and maintain the water quality of the region while also maximizing stakeholders' satisfaction and revenue. Distributed computing and human-guided search are both utilized to deal with the complex problem space. Several user modeling approaches were compared and integrated back into the interactive optimization process. Work remains to explore additional ways of dealing with limited data and to simplify the interactive optimization process when several stakeholders are present.

6. UNCERTAINTY BASED DEEP LEARNING NETWORKS FOR LIMITED DATA WETLANDS USER MODELS

6.1 Abstract

This chapter conveys a method for utilizing limited data approaches on deep networks. This is based on the calculation of the level of uncertainty that is associated with a sample of remaining training data. The method was developed for the Watershed Restoration using Spatio-Temporal Optimization of Resources (WRESTORE) system. WRESTORE acts as an interactive decision support system designed for performing multi-criteria decision analysis with a distributed system of conservation practices. The system is used on the Eagle Creek Watershed in Indiana, USA. Results show more ideal convergence when applying uncertainty-based incremental sampling.

6.2 Introduction

Existing work in the system uses a standard random incremental sampling method. The goal of this chapter is to provide an explanation of the existing method and our new method. WRESTORE is a user-friendly, virtual, interactive online decision support system that assists land-owners and other key stakeholders (e.g., the local government, investors) in watershed design. The system is applied on Eagle Creek Watershed which covers approximately 160 square miles of area in central Indiana. This area is situated more than ten miles northwest of downtown Indianapolis, USA.

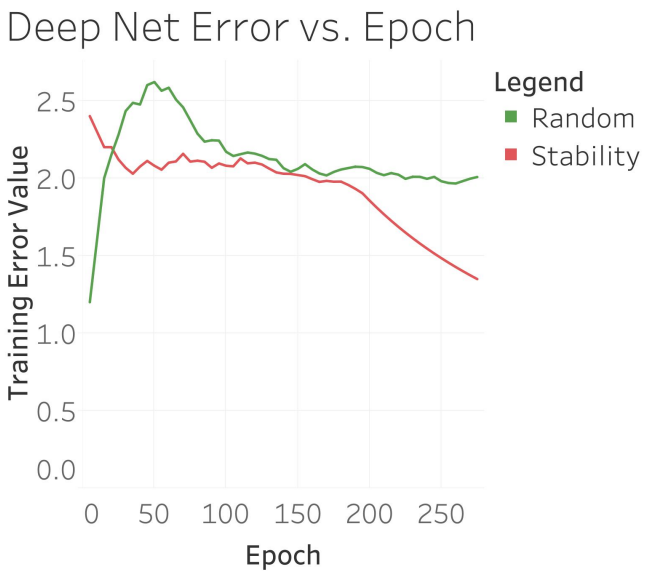

Figure 6.1.: Stability sampling vs. Random sampling training error

6.3 Background Literature

Luo used empirical eigenfunctions and neural networks to approximate optimal control [81], while others used a cerebellar model arithmetic computer neural network

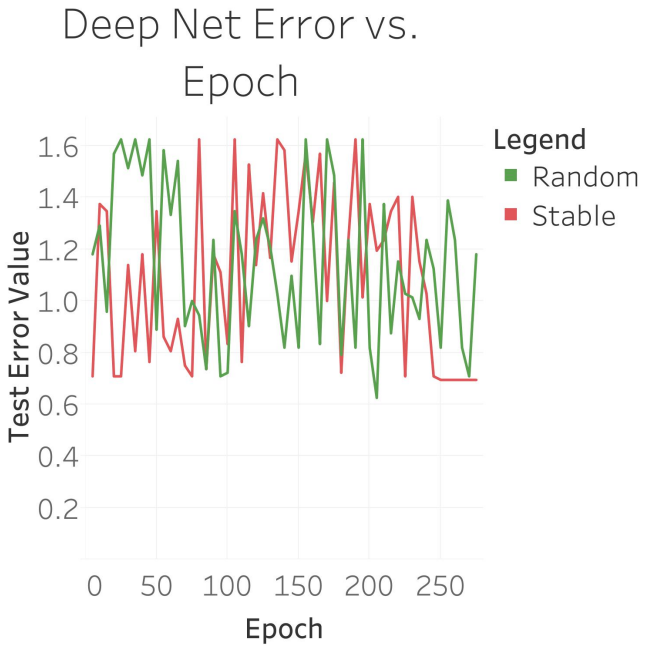

Figure 6.2.: Stability sampling vs. Random sampling test error

for optimal control [82]. Researchers have developed an interactive machine learning toolkit and applied it to several examples [83], while another study worked a number of low-latency, high-throughput uses cases with incremental model learning [84].

One paper developed a method for fuzzy neural networks which minimizes the next mean step error [85], while another compared several different models and dealt with model efficiency but not in a way that improves an individual model's efficiency [86]. Langkvist developed a human-in-the-loop intelligent system that allows the algorithm and user to collaborate [87]. Cauwenberghs demonstrated an online recursive algorithm for support vector machines (SVMs) [88]. Amershi showed the importance of studying the users of interactive machine learning systems [89].

Carpenter presented an architecture for utilizing fuzzy logic and Adaptive Resonance Theory (ART) for developing a new mini-max learning rule. Several training attempts were performed to minimize predictive error and improve accuracy, with the approach providing accuracy. The approach did encounter some issues with letter image data and also ran into issues with voting criteria [90].

6.4 Methodology

Eagle Creek Watershed is divided into 130 different sub-basins based on its geography. The utilized map assists in accurately measuring the direct flow of the water level in the district region. Wetlands are beneficial to the local environment for several reasons. For example, wetlands reduce the overall threat of flooding by reducing peak water from overflowing. The U.S. Department of Agriculture has created several programs for improving wetlands, such as the Wetlands Reserve Program (WRP).

Filter strips remove excess residue, organic materials, and various other environmental pollutants from wastewater. They also provide numerous additional benefits like removing excess residue before the water supply enters the waterway area. Grassed waterways carry run-off from excessive water flow and effectively minimizes soil erosion. Crop rotation control acts to stabilize the actively growing field crops

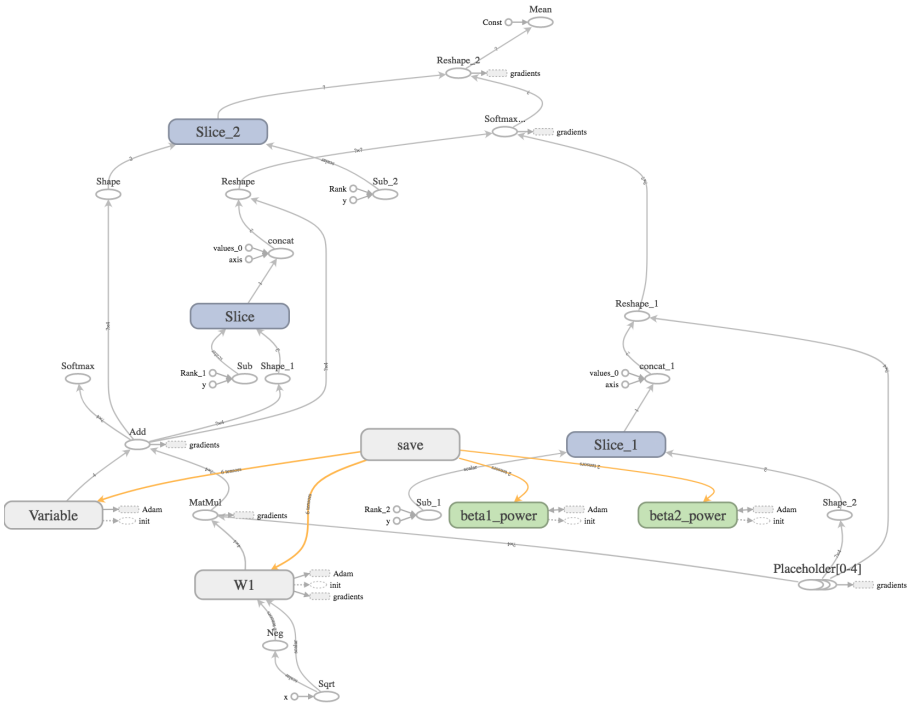

Figure 6.3.: Deep learning network architecture

and offers benefits like reduction of crop run-offs. Strip cropping provides increased soil moisture and higher water quality. Cover crops, main ryegrass and oats, and are grown between regular crops for the purpose of improving the overall productivity of farmers' crops. More information is available in the cited papers [91] [92] [50]. WRESTORE helps recommend combinations of BMPs for wetland designs.

It is frequently too expensive to obtain the amount of feedback necessary in order to resolve user-related issues (for example, user fatigue). Incremental user sampling minimizes user fatigue. Our uncertainty-based method includes a level of uncertainty which is used for determining the next point in the feature space to query. One of the most apparent challenges with this approach is limited training data; nonetheless, this challenge was partially addressed with the incorporation of domain knowledge and dataset augmentation. After integrating user models back into WRESTORE we noted a decrease in user fatigue and an increase in recommendation power [93].

The uncertainty method used could work by taking any of the following: the most uncertain training examples from the softmax function within each mini-batch, by taking the most uncertain examples from predictions which change with increasing L1 and L2 regularization over separate runs, by taking the most uncertain examples from a dropout based technique [31], and by taking the most uncertain examples from early termination over several runs. The optimal method for us seemed to best be determined empirically dependent on the use case and data set present.

6.4.1 Experimental setup

As previously discussed, data came from SQL Server instances described in the appendix. 20 participants volunteered to find watershed-scale plans that agreed with their individual subjective preferences. One limitation of the approach is that 14 of the stakeholders were not personally involved in the watershed, but all of them are from a community which is directly impacted by the watershed. To address this, several stakeholders more closely associated with Eagle Creek Watershed via land ownership or professional responsibilities were also included in the workshop.

6.4.2 Inputs and outputs of the models

As previously discussed, our physically based environmental objective functions (Peak Flow Reduction, Sediment Reduction, Nitrates Reduction, and Cost Revenue function) were estimated by the Soil and Water Assessment Tool at the entire watershed scale and at the local level for each sub-basin, these acted as the input for each model. The output layer here consists of the predicted user rating for a virtual user model or simulated decision maker. The architecture and configuration of each model (size of the hidden layers, regularization parameters, etc.) was determined empirically through hyperparameter optimization.

6.5 Results

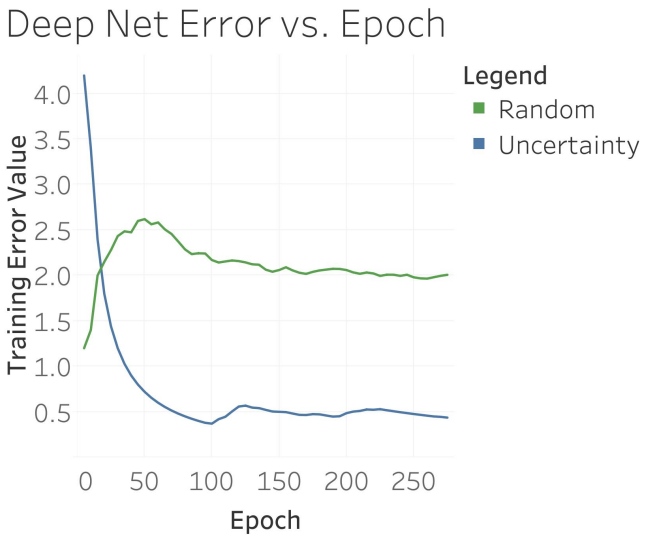

Figure 6.4.: Uncertainty sampling vs. Random sampling training error

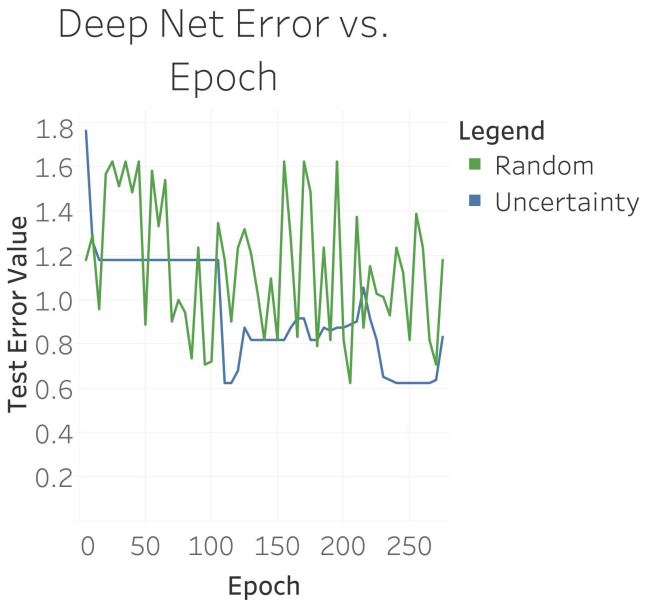

Figure 6.5.: Uncertainty sampling vs. Random sampling test error

6.5.1 Discussion of results

Our implementation makes use of a unified dataflow graph [94] to represent both computation and state. Nodes represent operations that need to be performed, while edges represent data flowing along a graph. The advantages of this approach include massive parallelization and lazy evaluation. Our implementation of stochastic gradient descent made use of the first order gradient-based Adam optimizer [95].

TensorFlow is a reference implementation commonly used for diverse use cases including computer vision, NLP, and other applications. The input layer to our network consisted of environmental fitness functions and the output layer was the prediction of the user model. The architecture utilized is pictured in Figure 6.3.

The reshape, shape, slice, and concat functions are all dimensionality manipulators utilized while RELU, gradients, sub, add, mat mul, sqrt, neg, mean, init, beta1 power, beta2 power, and Adam are all data mutators. Placeholder, variable, W1, and save are all internal development features of TensorFlow.

Zheng developed a general stability training method [96] to stabilize deep networks against extreme instability against contrived input perturbations commonly known as adversarial examples. They developed a parameterized Gaussian weighted stability noise factor which we replicate the logic of here for comparison against a new uncertainty-based sampling method inspired by similar approaches in the field [74] [97].

The mean absolute error (MAE) is given by:

$$\frac{1}{N}\sum_{i=1}^{N}|T_i - C_i|$$

where N is the number of samples in the test set, T_i is the true label of the i-th sample and C_i is the predicted label.

Mean absolute error has the benefit of offering the average absolute difference between C_i and T_i, while it has the drawback of being a scale-dependent accuracy measure which cannot be used to make comparisons between series of different scales.

As can be seen in Figures 6.1 and 6.2, stability based incremental sampling outperformed random sampling. Also, as can be seen in Figures 6.4 and 6.5, uncertainty-based incremental sampling also outperformed random sampling. Experiments were ran with micro batch size = 1 and learning rate = 0.008.

Comparing the uncertainty-based incremental sampling and the stability-based incremental sampling, our uncertainty-based sampling method was able to converge with half of the training data of the stability-based approach.

Random feature space sampling, parameterized exploration versus exploitation sampling, sampling methods which optimize prediction stability, and sampling methods that minimize prediction generalization error are all designs of experiment-based techniques that remain to be explored further. Other techniques also exist in the community which could be further compared to this work.

Practically, we also place a great emphasis on embedding these techniques back into our wetland decision support system where integration would yield faster convergence to Pareto-optimal solutions. This would drive better community and stakeholder outcomes.

6.6 Conclusion

WRESTORE more efficiently deals with the expensive task of user querying through the use of an uncertainty-based sampling technique. The proposed uncertainty-based sampling method showed faster convergence than either a random sampling technique or a stability-based sampling technique. This technique allows for faster convergence to a Pareto-optimal solution. This research ultimately drives more favorable outcomes for the community and key stakeholders. Areas remaining for future work including comparison against other sampling techniques.

7. NON-STATIONARY REINFORCEMENT-LEARNING BASED DIMENSIONALITY REDUCTION FOR MULTI-OBJECTIVE OPTIMIZATION OF WETLAND DESIGN

7.1 Abstract

This chapter presents a non-stationary, reinforcement-based method for feature selection. The primary contribution of this chapter is an RL-based feature selection technique for interactive optimization watershed designs.

7.2 Introduction

Wetlands are considered a mass area of land that have saturated soil and reduced peak water-flow levels during flooding. Best management practices for wetlands include the following: filter strips, grassed waterways, crop rotation, no-till, strip cropping, and covering crops. WRESTORE balances the interests in competing key stakeholder groups by using a combination of a strictly quantitative approach and a learned quantitative approach.

Reinforcement learning is considered a type of machine learning, with common use cases including optimal control, dynamic programming, approximation, and action selection. Reinforcement learning [98] [99] has been studied with several use cases, such as neuroscience, robotics, deep learning, etc.

7.3 Background Work

This section provides an overview of reinforcement learning and its applications to dimensionality reduction.

Mnih outlines a deep reinforcement learning algorithm [100] which is given pixels as input and trained to play a game. The algorithm uses replay memory and random sampling for performing updates. Another paper developed several improvements for deep reinforcement learning in a method called Rainbow. Rainbow addresses the high number of hyperparameters present being through a restricted calibration hyperparameter search [101].

The Arcade Learning Environment is a benchmark for some current algorithms in reinforcement learning with several game modes and best practices [102]. Russo provides a tutorial on Thompson sampling. Thompson sampling chooses actions that address the famous exploration-exploitation problem in the multi-armed bandit problem [103].

One study outlined a goal driven dimensionality reduction of objects with reinforcement learning on image pixels; however, in a simulation they found that model-based learning is superior [104]. Other research uses reinforcement learning to perform probabilistic non-linear dimensionality reduction [105].

Some research has examined a reinforcement driven dimensionality reduction (RDDR) model and combined experimental and theoretical results to show the efficacy of the model for dimensionality reduction [106]. Rusch summarized the relationship between attention and learning and shows that including attention in RL-based techniques will improve overall performance [107].

7.4 Methodology

Each of the methods provided (e.g., Random, Boltzmann, Thompson, Omniscient process) is expected to show a monotonically increasing reward. Random sampling represents a completely naive approach, while omniscient sampling represents an above optimal sampling method because it is already always aware of the optimal policy decision.

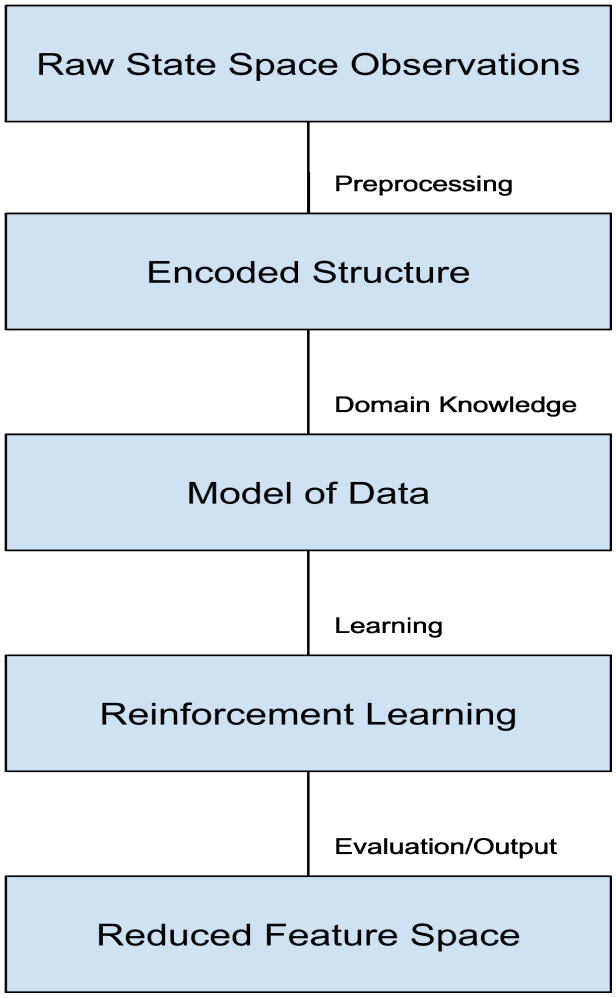

Figure 7.1.: RL-based feature selection process

Boltzmann sampling is chosen as a greedy approach, while Thompson sampling is used as more of a Bayesian approach. In addition to the total data accumulated reward, the function of the Root Mean Square Error (RMSE) between the actual utility and estimated utility can be used as a measure of exploration versus exploitation. The approach used for reinforcement based learning feature selection is illustrated in Figure 7.1.

7.4.1 Boltzmann Sampling

Boltzmann sampling draws from a Gibbs distribution in a greedy fashion. This process is commonly known as "softmax action selection." Boltzmann sampling defines the probability of a certain state S as a function of that state's energy and temperature of the system to which the distribution is applied. The Gibbs distribution is the distribution that maximizes the entropy.

Pseudo-code for Boltzmann sampling is found below:

7.4.2 Thompson Sampling

Thompson sampling has been proven to converge [108], instantaneously correcting itself. This method has received new-found interest in various applications (for example, in dueling bandit problems [109]).

Pseudo-code for Thompson sampling is found below:

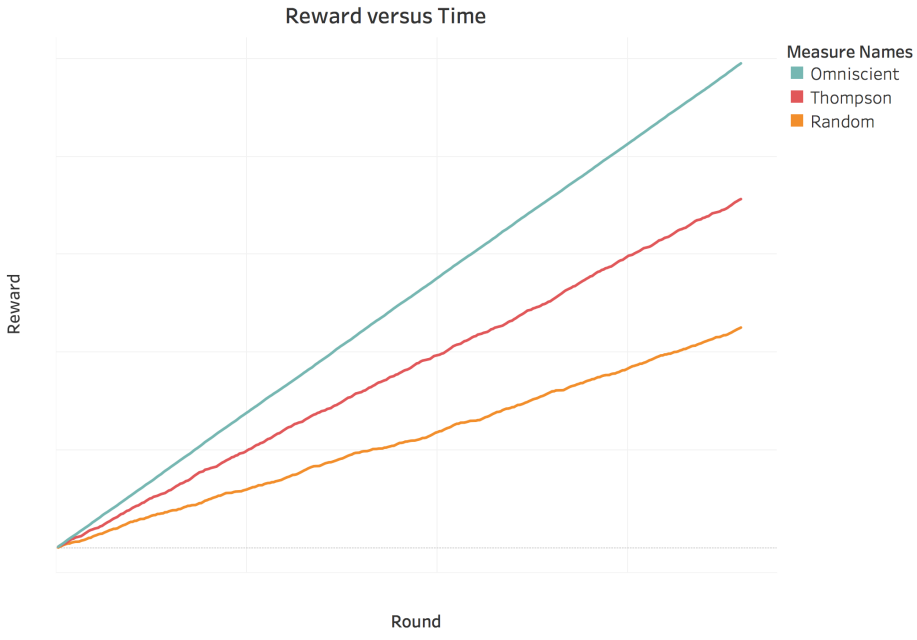

Figure 7.2.: Thompson Sampling

Algorithm Pseudo-code for Boltzmann sampling

```
procedure BOLTZMANN(x, temperature)
    exponent = np.true_divide(x − np.max(x), temperature)
    returnnp.exp(exponent)/np.sum(np.exp(exponent))
end procedure
procedure RETURN_BOLTZMANN_ACTION(temperature, reward_counter_array)
    tot_arms = reward_counter_array.shape[0]
    boltzmann_distribution = boltzmann(reward_counter_array, temperature)
    returnnp.random.choice(tot_arms, p = boltzmann_distribution)
end procedure
procedure MAIN
    reward_distribution = [0, 0.6, 0, 0, 0.1, 0.1, 0, 0, 0.05, 0.05]
    temperature_start = 0.1
    temperature_stop = 0.0001
    epsilon = 0.1
    tot_arms = 10
    tot_episodes = 1000
    tot_steps = 359
    cumulated_rewardlist = list()
    average_utilityarray = np.zeros(tot_arms)
    temperature_array = np.linspace(temperature_start, temperature_stop, num = totsteps)
    print("StartingBoltzmannagent...")
    for episode in range(tot_episodes): do
        my_bandit = WRestoreMABandit(reward_probability_list = reward_distribution)
        cumulated_reward = 0
        reward_counter_array = np.zeros(tot_arms)
        action_counter_array = np.full(tot_arms, 1.0e − 5)
        for step in range(tot_steps): do
            temperature = temperature_array[step]
            action = return_boltzmann_action(temperature, np.truedivide(reward_counter_array, action_counter_array))
            reward = mybandit.step(action)
            reward_counter_array[action]+ = reward
            actioncounterarray[action]+ = 1
            cumulatedreward+ = reward
        end for
        cumulated_reward_list.append(cumulated_reward)
        utility_array = np.true_divide(reward_counter_array, action_counter_array)
        average_utility_array+ = utility_array
        print("AccumulatedReward : " + str(cumulated_reward))
        print("UtilityRMSE : " + str(return_rmse(utility_array, reward_distribution)))
    end for
end procedure
```

Algorithm Pseudo-code for Boltzmann sampling

```
procedure RETURN_RMSE(predictions, targets)
    return np.sqrt(((predictions - targets)**2).mean())
end procedure
procedure RETURN_THOMPSON_ACTION(success_counter_array, failure_counter_array)
    beta_sampling_array = np.random.beta(success_counter_array, failure_counter_array)
    return  np.argmax(beta_sampling_array)
end procedure
procedure MAIN
    reward_distribution = [0, 0.6, 0, 0, 0.1, 0.1, 0, 0, 0.05, 0.05]
    tot_arms = 10
    tot_episodes = 1000
    tot_steps = 359
    print_every_episodes = 1
    average_utility_array = np.zeros(tot_arms)
    for episode in range(tot_episodes): do
        my_bandit = WRestoreMABandit(reward_probability_list = reward_distribution)
        cumulated_reward = 0
        success_counter_array = np.ones(tot_arms)
        failure_counter_array = np.ones(tot_arms)
        action_counter_array = np.full(tot_arms, 1.0e − 5)
        for step in range(tot_steps): do
            action = return_thompson_action(success_counter_array, failure_counter_array)
            reward = my_bandit.step(action)
            if reward > 0: then
                success_counter_array[action]+ = 1
            elsereward == 0:
                failure_counter_array[action]+ = 1
            end if
            action_counter_array[action]+ = 1
            cumulated_reward+ = reward
        end for
        cumulated_reward_list.append(cumulated_reward)
        utility_array = np.true_divide(success_counter_array, action_counter_array)
        average_utility_array+ = utility_array
        print("AccumulatedReward : " + str(cumulated_reward))
        print("UtilityRMSE : " + str(return_rmse(utility_array, reward_distribution)))
    end for
end procedure
```

Figure 7.3.: Boltzmann Sampling

7.5 Results

7.5.1 Design of experiment

As previously discussed, our data came from SQL Server instances described in the appendix. User data came from previously held workshops and we preprocessed the data to remove near zero variance users and users who showed negative or small correlation with one or more of the objective fitness functions previously described. Synthetic user data with known behaviors and preferences was also utilized to validate that our models would behave as expected when given users with known goals.

7.5.2 Discussion of results

The study has provided four different user RL-based techniques: Random sampling, Thompson sampling, Boltzmann sampling, and Omniscient sampling. Random

sampling was utilized by implementing a strategic plan, such as an unknowing gambler choosing each lever from a stationary and uniform distribution. Thompson sampling and the Boltzmann sampling were implemented with the characteristics described in the previous section. Omniscient sampling is directed to an oracle platform which acts how an ideal agent would execute the given command.

Ten agents were initially utilized, but the goal was to eventually reduce it down to three agents respectively. Hyper-parameter optimization was performed with a mixture of grid search and random search. As seen in Figure 7.2, Thompson sampling outperformed random selection and underperformed against omniscient sampling. As seen in Figure 7.3, Boltzmann sampling also noticeably outperformed random selection but still underperformed against omniscient sampling. Both sampling methods were able to generate a significant reward.

Root Mean Square Error is given by

$$\text{Root Mean Square Error} = \sqrt{\frac{\sum \left(P_{\text{est.ut.dist.}} - P_{\text{tr.ut.dist.}}\right)^2}{n}}.$$

in which we are taking the difference between the optimal utility distribution and the average estimation of the data utility distribution. RMSE is used here as an indirect measure of exploration versus exploitation. The random agent had a negligible RMSE while Thompson sampling and Boltzmann sampling methods returned larger values for the RMSE due to the fact that they performed much more exploitation than the random sampling.

7.6 Conclusion

Thompson sampling and Boltzmann sampling provided a significant improvement over random sampling. The Boltzmann sampling method outperformed Thompson sampling, although it was possible for either technique to outperform the other, depending on the choice of distributions and parameter initialization. This chapter developed a functional, non-stationary RL-based feature data selection technique for an interactive, multi-objective optimization system.

8. CONCLUSION

This dissertation developed a framework for multi-stakeholder consensus decision-making taking into consideration the subjective preferences of stakeholders as well as quantifiable environmental fitness functions.

8.1 Conclusions

"The promise of artificial intelligence and computer science generally vastly outweighs the impact it could have on some jobs in the same way that, while the invention of the airplane negatively affected the railroad industry, it opened a much wider door to human progress."- Paul Allen.

- Chapter 1 "Introduction". In Chapter 1, an overview of the problem of the dissertation is provided, helping multiple stakeholders collaborate effectively on designing optimal solutions and synthesizing their perspectives into group decisions. It also provides additional background information about the dissertation as well as information about research goals and assumptions and limitations. Finally, the chapter provides an outline for the remainder of the dissertation.

- Chapter 2 "Evaluation of Machine Learning Approaches". Several approaches exist in machine learning for solving different use cases. Cost functions quantify your performance on a task so the problem can be modeled as an optimization process to minimize or maximize against. This chapter examines several different metrics to help learn lessons for arriving at a set of suitable metrics for our study.

- Chapter 3 "WRESTORE and IGAMI2". Chapter 3 provided additional background about the use case, WRESTORE, and IGAMI2. The information was

useful for the remainder of the dissertation. Additional background information is also found within the appendices.

- Chapter 4 "Comparison of Neural Methods for User Modeling in Wetland Design" builds a user modeling component and shows why working with limited data is a difficult task. Complex techniques can produce incorrect results due to issues coming from limited size datasets, this chapter examines several approaches and finds that deep networks performed the best.

- Chapter 5 "Fuzzy and Deep Learning Approaches for User Modeling in Wetland Design". Chapter 5 presents an original approach to dealing with limited data involving application-specific data augmentation. Additional work remained to explore additional ways of dealing with limited data and to deal with the presence of several stakeholders.

- Chapter 6 "Uncertainty-Based Deep Learning for Wetland Design". Chapter 6 presented a method for deep networks that samples remaining training data with uncertainty based modeling. The technique showed faster convergence than either a random sampling technique or a stability-based sampling technique and allowed faster converge to a Pareto optimum.

- Chapter 7 "Non-Stationary Reinforcement-Learning Based Dimensionality Reduction for Multi-objective Optimization of Wetland Design". Chapter 7 studies non-stationary reinforcement-based learning for interactive optimization feature selection. Thompson sampling and Boltzmann sampling are shown to provide an effective method of feature selection.

8.2 Contribution

"Real knowledge is to know the extent of one's ignorance."- Confucius

This dissertation describes contributions that were made to an existing interactive decision support system. Contributions to the system include techniques for increasing

the reliability of user models in the presence of limited data, more efficient learning in the presence of limited data via an uncertainty based sampling technique, and a reinforcement-based learning technique for choosing user models that are known to lead to more objectively measurable optimal outcomes. The limited data techniques developed here can be applied to other use cases, while the reinforcement-based user model selection developed here could also be given a different reward function to optimize for other characteristics such as novelty or diversity.

8.3 Future Work

> *"Making AI more sensitive to the full scope of human thought is no simple task. The solutions are likely to require insights derived from fields beyond computer science, which means programmers will have to learn to collaborate more often with experts in other domains."- Fei-Fei Li.*

This dissertation addressed issues of both limited data and verifying validity with a variety of stakeholders. Future work will tackle users with conflicting goals and can also spend time developing ways to deal with untrustworthy stakeholders.

Additional work for dealing with limited data could include integrating external domain knowledge, examining application-specific ways of reducing degrees of freedom, transferring models from other domains, and additional data augmentation techniques.

A key consideration for human-in-the-loop machine learning systems is making sure that qualitative information and preferences are captured, that predictions are accurate, and that stakeholders understand and acknowledge the uncertainty associated with the system's current predictions. Transparency and uncertainty quantification are additional areas of future work.

When dealing with multiple stakeholders, multiple issues arise including fairness, negotiation, prediction complexity, trust, etc. Additional work for dimensionality reduction of user models in interactive optimization includes an examination of dif-

ferent fusion techniques, including linear and nonlinear combinations of users, and the impact on the issues already cited.

REFERENCES

REFERENCES

[1] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, 2006.

[2] J. Sim and C. C. Wright, "The kappa statistic in reliability studies: Use, interpretation, and sample size requirements," *Physical Therapy*, vol. 85, no. 3, pp. 257–268, 03 2005.

[3] R. J. Hyndman and A. B. Koehler, "Another look at measures of forecast accuracy," *International Journal of Forecasting*, vol. 22, pp. 679–688, 2006.

[4] D. Wackerly, W. Mendenhall, and R. Scheaffer, *Mathematical Statistics with Applications*, ser. Mathematical Statistics with Applications. Thomson Higher Education, 2008. [Online]. Available: https://books.google.gr/books?id=d6IMnwEACAAJ

[5] C. J. Willmott and M. Kenji, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate Research*, vol. 30, pp. 79–82, Dec 2005.

[6] R. G. Pontius, O. Thontteh, and H. Chen, "Components of information for multiple resolution comparison between maps that share a real variable," *Environmental and Ecological Statistics*, vol. 15, no. 2, pp. 111–142, Jun 2008. [Online]. Available: https://doi.org/10.1007/s10651-007-0043-y

[7] L. Hubert and P. Arabie, "Comparing partitions," *Journal of Classification*, vol. 2, no. 1, pp. 193–218, Dec 1985. [Online]. Available: https://doi.org/10.1007/BF01908075

[8] T. Cover and J. Thomas, *Elements of Information Theory*. Wiley, 2006. [Online]. Available: https://books.google.gr/books?id=Pgr3uAEACAAJ

[9] A. Turpin and F. Scholer, "User performance versus precision measures for simple search tasks," in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '06. New York, NY, USA: ACM, 2006, pp. 11–18. [Online]. Available: http://doi.acm.org/10.1145/1148170.1148176

[10] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, Jun 2010. [Online]. Available: http://pascallin.ecs.soton.ac.uk/challenges/VOC/pubs/everingham10.pdf

[11] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge University Press, 2008. [Online]. Available: http://nlp.stanford.edu/IR-book/html/htmledition/evaluation-of-ranked-retrieval-results-1.html

[12] K. Brodersen, C. Ong, K. Stephan, and J. Buhmann, "The binormal assumption on precision-recall curves," in *Proceedings of the 20th International Conference on Pattern Recognition*, 2010, pp. 4263–4266. [Online]. Available: url=https://web.archive.org/web/20121208201457

[13] C. D. Manning, P. Raghavan, and H. Schütze. (2009) Chapter 8: Evaluation in information retrieval. [Online]. Available: https://nlp.stanford.edu/IR-book/pdf/08eval.pdf

[14] E. Voorhees, "Proceedings of the 8th text retrieval conference," in *TREC-8 Question Answering Track Report*, 1999, pp. 77–82.

[15] D. R. Radev, H. Qi, H. Wu, and W. Fan, "Evaluating web-based question answering systems," in *Proceedings of LREC*, 2002.

[16] L. I.-K. Lin, "A concordance correlation coefficient to evaluate reproducibility," *Biometrics (journal)*, vol. 45, pp. 255–268, 3 1989.

[17] C. A. E. Nickerson, "A note on "a concordance correlation coefficient to evaluate reproducibility," *Biometrics (journal)*, vol. 53, pp. 1503–1507, 12 1997.

[18] V. N. Gudivada, R. Baeza-Yates, and V. V. Raghavan, "Big data: Promises and problems," *Computer*, no. 3, pp. 20–23, Mar 2015.

[19] M. Banko and E. Brill, "Scaling to Very Very Large Corpora for Natural Language Disambiguation," in *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ser. ACL '01. Stroudsburg, PA, USA: Association for Computational Linguistics, Jul 2001, pp. 26–33. [Online]. Available: https://doi.org/10.3115/1073012.1073017

[20] A. Halevy, P. Norvig, and F. Pereira, "The Unreasonable Effectiveness of Data," *IEEE Intelligent Systems*, vol. 24, no. 2, pp. 8–12, Mar 2009.

[21] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, "Analysis of representations for domain adaptation," in *Advances in neural information processing systems*, Jan 2007, pp. 137–144.

[22] H. Bunke and T. M. Ha, "Off-Line, Handwritten Numeral Recognition by Perturbation Method," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, vol. 19, pp. 535–539, May 1997. [Online]. Available: doi.ieeecomputersociety.org/10.1109/34.589216

[23] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, Jun 2002.

[24] Texas A & M University, "Soil and Water Assessment Tool," 2017, http://blackland.tamu.edu/files/2012/09/SWAT.2012.pdf. Accessed Jan 17, 2019. [Online]. Available: http://blackland.tamu.edu/files/2012/09/SWAT.2012.pdf.

[25] G. Corani and M. Zaffalon, "Learning reliable classifiers from small or incomplete data sets: the naive credal classifier 2," *Journal of Machine Learning Research*, vol. 9, no. Apr, pp. 581–621, Apr 2008.

[26] R. Eggeling, M. Koivisto, and I. Grosse, "Dealing with small data: On the generalization of context trees," in *International Conference on Machine Learning*, Jun 2015, pp. 1245–1253.

[27] W. Lee and S.-C. Ong, "Learning from small data sets to improve assembly semiconductor manufacturing processes," in *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, vol. 2, Feb 2010, pp. 50–54.

[28] A. Oniśko, M. J. Druzdzel, and H. Wasyluk, "Learning Bayesian network parameters from small data sets: Application of Noisy-OR gates," *International Journal of Approximate Reasoning*, vol. 27, no. 2, pp. 165–182, Aug 2001.

[29] T. Shaikhina, D. Lowe, S. Daga, D. Briggs, R. Higgins, and N. Khovanova, "Machine learning for predictive modelling based on small data in biomedical engineering," *IFAC-PapersOnLine*, vol. 48, no. 20, pp. 469–474, Jan 2015.

[30] A. Tengli, A. Dubrawski, and L. Chen, "Learning Predictive Models from Small Sets of Dirty Data," in *International Conference on Information and Automation*, 2005.

[31] T. Yang and V. Kecman, "Adaptive local hyperplane algorithm for learning small medical data sets," *Expert Systems*, vol. 26, no. 4, pp. 355–359, Sep 2009.

[32] D.-C. Li, C. Wu, and F. M. Chang, "Using data-fuzzification technology in small data set learning to improve FMS scheduling accuracy," *The International Journal of Advanced Manufacturing Technology*, vol. 27, no. 3, pp. 321–328, Dec 2005. [Online]. Available: https://doi.org/10.1007/s00170-003-2184-y

[33] D.-C. Li, C.-S. Wu, T.-I. Tsai, and Y.-S. Lina, "Using mega-trend-diffusion and artificial samples in small data set learning for early flexible manufacturing system scheduling knowledge," *Computers & Operations Research*, vol. 34, no. 4, pp. 966 – 982, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0305054805001693

[34] D.-C. Li and I.-H. Wen, "A genetic algorithm-based virtual sample generation technique to improve small data set learning," *Neurocomputing*, vol. 143, pp. 222–230, Nov 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925231214007462

[35] N. H. Ruparel, N. M. Shahane, and D. P. Bhamare, "Learning from small data set to build classification model: A survey," in *Proc. IJCA Int. Conf. Recent Trends Eng. Technol.(ICRTET)*, May 2013, pp. 23–26.

[36] P. Niyogi, F. Girosi, and T. Poggio, "Incorporating prior information in machine learning by creating virtual examples," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2196–2209, Nov 1998.

[37] V. Rieser and O. Lemon, "Learning and Evaluation of Dialogue Strategies for New Applications: Empirical Methods for Optimization from Small Data Sets," *Comput. Linguist.*, vol. 37, no. 1, pp. 153–196, Mar 2011. [Online]. Available: http://dx.doi.org/10.1162/coli_a_00038

[38] R. Mao, H. Zhu, L. Zhang, and A. Chen, "A New Method to Assist Small Data Set Neural Network Learning," in *Sixth International Conference on Intelligent Systems Design and Applications*, vol. 1, Oct 2006, pp. 17–22.

[39] G. Forman and I. Cohen, "Learning from little: Comparison of classifiers given little training," in *Knowledge Discovery in Databases: PKDD 2004*, J.-F. Boulicaut, F. Esposito, F. Giannotti, and D. Pedreschi, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, Sep 2004, pp. 161–172.

[40] G. M. Weiss and F. Provost, "Learning when training data are costly: The effect of class distribution on tree induction," *Journal of Artificial Intelligence Research*, vol. 19, pp. 315–354, Oct 2003.

[41] D. Meignan, S. Knust, J.-M. Frayret, G. Pesant, and N. Gaud, "A Review and Taxonomy of Interactive Optimization Methods in Operations Research," *ACM Trans. Interact. Intell. Syst.*, vol. 5, no. 3, pp. 17:1–17:43, Sep 2015. [Online]. Available: http://doi.acm.org/10.1145/2808234

[42] M. Tory and T. Moller, "Human factors in visualization research," *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 1, pp. 72–84, Jan 2004.

[43] H. Takagi, "Interactive evolutionary computation: fusion of the capabilities of EC optimization and human evaluation," *Proceedings of the IEEE*, vol. 89, no. 9, pp. 1275–1296, Sep 2001.

[44] X. Llorà, K. Sastry, D. E. Goldberg, A. Gupta, and L. Lakshmi, "Combating user fatigue in iGAs: partial ordering, support vector machines, and synthetic fitness," in *Proceedings of the 7th annual conference on Genetic and evolutionary computation*. ACM, Jun 2005, pp. 1363–1370.

[45] D. Kneller, F. Cohen, and R. Langridge, "Improvements in protein secondary structure prediction by an enhanced neural network," *Journal of Molecular Biology*, vol. 214, no. 1, pp. 171 – 182, Jul 1990. [Online]. Available: http://www.sciencedirect.com/science/article/pii/002228369090154E

[46] Stanford, "Fuzzy Logic," *Stanford encyclopedia of philosophy, Stanford University*, 2010.

[47] J. R. Jang, "ANFIS: adaptive-network-based fuzzy inference system," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, May 1993.

[48] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural Computation*, vol. 1, no. 4, pp. 541–551, Dec 1989. [Online]. Available: https://doi.org/10.1162/neco.1989.1.4.541

[49] G. E. Hinton, "Learning multiple layers of representation," *Trends in Cognitive Sciences*, vol. 11, no. 10, pp. 428 – 434, Oct 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364661307002173

[50] M. Babbar-Sebens and B. S. Minsker, "Interactive Genetic Algorithm with Mixed Initiative Interaction for multi-criteria ground water monitoring design," *Applied Soft Computing*, vol. 12, no. 1, pp. 182–195, Jan 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494611003371

[51] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, Apr 2002.

[52] D. Anderson, E. Anderson, N. Lesh, J. Marks, K. Perlin, D. Ratajczak, and K. Ryall, "Human-guided simple search: combining information visualization and heuristic search," in *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*. ACM, Nov 1999, pp. 21–25.

[53] A. Kobsa, "User modeling: Recent work, prospects and hazards," *Human Factors in Information Technology*, vol. 10, pp. 111–111, 1993.

[54] A. Jennings and H. Higuchi, "A personal news service based on a user model neural network," *IEICE Transactions on Information and Systems*, vol. 75, no. 2, pp. 198–209, Mar 1992.

[55] J. A and H. Higuchi, "A user model neural network for a personal news service," *User Modeling and User-Adapted Interaction*, vol. 3, no. 1, pp. 1–25, Mar 1993. [Online]. Available: https://doi.org/10.1007/BF01099423

[56] R. Zou, W.-S. Lung, and J. Wu, "An adaptive neural network embedded genetic algorithm approach for inverse water quality modeling," *Water Resources Research*, vol. 43, no. 8, Aug 2007. [Online]. Available: https://doi.org/10.1029/2006WR005158

[57] J. Putnam, "Genetic Programming of Music," New Mexico Institute of Mining and Technology, Tech. Rep., 1994.

[58] D. Goldberg and R. Lingle, "Proceedings of the Second International Conference on Genetic Algorithms," in *Mahwah, NJ: Lawrence Eribaum Associates*, 1985.

[59] P. Beyls, "Selectionist musical automata: Integrating explicit instruction and evolutionary algorithms," in *IX Brazilian Symposium on Computer Music*. Brazilian Computing Society, 2003.

[60] K. Özgür, "Evolutionary fuzzy models for river suspended sediment concentration estimation," *Journal of Hydrology*, vol. 372, no. 1, pp. 68–79, Jun 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0022169409002170

[61] I. Zukerman and D. W. Albrecht, "Predictive Statistical Models for User Modeling," *User Modeling and User-Adapted Interaction*, vol. 11, no. 1, pp. 5–18, Mar 2001. [Online]. Available: https://doi.org/10.1023/A:1011175525451

[62] Y. Ishiwaka, H. Yokoi, and Y. Kakazu, "Adaptive learning interface used physiological signals," in *Smc 2000 conference proceedings, 2000 ieee international conference on systems, man and cybernetics*, vol. 1, Oct 2000, pp. 32–37.

[63] Q. Chen and A. F. Norcio, "A neural network approach for user modeling," in *Conference Proceedings 1991 IEEE International Conference on Systems, Man, and Cybernetics*, Oct 1991, pp. 1429–1434 vol.2.

[64] F. Chiari, M. Delhom, J. F. Santucci, and J. B. Filippi, "Prediction of the hydrologic behavior of a watershed using artificial neural networks and geographic information systems," in *Smc 2000 conference proceedings, 2000 ieee international conference on systems, man and cybernetics*, vol. 1, Oct 2000, pp. 382–386.

[65] R. Yasdi, "A Literature Survey on Applications of Neural Networks for Human-Computer Interaction," *Neural Computing & Applications*, vol. 9, no. 4, pp. 245–258, Dec 2000. [Online]. Available: https://doi.org/10.1007/s005210070002

[66] G. Ghinea, G. D. Magoulas, and C. Siamitros, "Multicriteria decision making for enhanced perception-based multimedia communication," *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, vol. 35, no. 6, pp. 855–866, Nov 2005.

[67] M. Babbar-Sebens, R. C. Barr, L. P. Tedesco, and M. Anderson, "Spatial identification and optimization of upland wetlands in agricultural watersheds," *Ecological Engineering*, vol. 52, pp. 130–142, Mar 2013. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0925857412004478

[68] R. H. Bonczek, C. W. Holsapple, and A. B. Whinston, *Foundations of decision support systems*. Academic Press, 2014.

[69] H.-S. Kim and S.-B. Cho, "Application of interactive genetic algorithm to fashion design," *Engineering Applications of Artificial Intelligence*, vol. 13, no. 6, pp. 635 – 644, Dec 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0952197600000452

[70] C. A. Stewart, "Big Data, Big Red II, Data Capacitor II, Wrangler, Jetstream, and Globus Online," *Presented to Microsoft, Inc. visiting group, Indiana University, Bloomington IN*, 2015.

[71] F.-J. Chang and Y.-T. Chang, "Adaptive neuro-fuzzy inference system for prediction of water level in reservoir," *Advances in Water Resources*, vol. 29, no. 1, pp. 1 – 10, Jan 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0309170805001338

[72] J.-S. R. Jang *et al.*, "Fuzzy Modeling Using Generalized Neural Networks and Kalman Filter Algorithm," in *AAAI*, vol. 91, Jul 1991, pp. 762–767.

[73] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436 EP –, May 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[74] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, Jan 2014.

[75] M. D. Zeiler, "ADADELTA: An Adaptive Learning Rate Method," *CoRR*, vol. abs/1212.5701, 2012. [Online]. Available: http://arxiv.org/abs/1212.5701

[76] Y. Bengio, "Learning Deep Architectures for AI," *Foundations and Trends® in Machine Learning*, vol. 2, no. 1, pp. 1–127, Nov 2009. [Online]. Available: http://dx.doi.org/10.1561/2200000006

[77] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, pp. 281–305, Feb 2012.

[78] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, Jun 2011, pp. 315–323.

[79] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, Jan 2014.

[80] S. Louisy, G. McGrawy, and R. O. Wyckoy, "CBR Assisted Explanation of GA Results," *Computer Science*, vol. 812, pp. 855–6486y, 1992.

[81] B. Luo and H. Wu, "Approximate Optimal Control Design for Nonlinear One-Dimensional Parabolic PDE Systems Using Empirical Eigenfunctions and Neural Network," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 6, pp. 1538–1549, Dec 2012.

[82] Y. H. Kim and F. L. Lewis, "Optimal design of CMAC neural-network controller for robot manipulators," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 1, pp. 22–31, Feb 2000.

[83] B. Jiang and J. Canny, "Interactive Machine Learning via a GPU-accelerated Toolkit," in *Proceedings of the 22Nd International Conference on Intelligent User Interfaces*, ser. IUI '17. New York, NY, USA: ACM, 2017, pp. 535–546. [Online]. Available: http://doi.acm.org/10.1145/3025171.3025172

[84] C. Mayer, R. Mayer, and M. Abdo, "StreamLearner: Distributed Incremental Machine Learning on Event Streams: Grand Challenge," in *Proceedings of the 11th ACM International Conference on Distributed and Event-based Systems*, ser. DEBS '17. New York, NY, USA: ACM, Jun 2017, pp. 298–303. [Online]. Available: http://doi.acm.org/10.1145/3093742.3095103

[85] H.-H. Tsai and P.-T. Yu, "On the optimal design of fuzzy neural networks with robust learning for function approximation," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 30, no. 1, pp. 217–223, Feb 2000.

[86] T. Choi, C. Hui, S. Ng, and Y. Yu, "Color Trend Forecasting of Fashionable Products with Very Few Historical Data," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 42, no. 6, pp. 1003–1010, Nov 2012.

[87] M. Längkvist, M. Alirezaie, A. Kiselev, and A. Loutfi, "Interactive Learning with Convolutional Neural Networks for Image Labeling," in *International Joint Conference on Artificial Intelligence (IJCAI) :*, Jul 2016.

[88] G. Cauwenberghs and T. Poggio, "Incremental and decremental support vector machine learning," in *Advances in neural information processing systems*, 2001, pp. 409–415.

[89] S. Amershi, M. Cakmak, W. B. Knox, and T. Kulesza, "Power to the people: The role of humans in interactive machine learning," *AI Magazine*, vol. 35, no. 4, pp. 105–120, Dec 2014.

[90] G. A. Carpenter, S. Grossberg, N. Markuzon, J. H. Reynolds, and D. B. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps," *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 698–713, Sep 1992.

[91] M. Babbar-Sebens, S. Mukhopadhyay, V. B. Singh, and A. D. Piemonti, "A web-based software tool for participatory optimization of conservation practices in watersheds," *Environmental Modelling & Software*, vol. 69, pp. 111 – 127, Jul 2015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1364815215000912

[92] A. D. Piemonti, M. Babbar-Sebens, S. Mukhopadhyay, and A. Kleinberg, "Interactive genetic algorithm for user-centered design of distributed conservation practices in a watershed: An examination of user preferences in objective space and user behavior," *Water Resources Research*, vol. 53, no. 5, pp. 4303–4326, May 2017. [Online]. Available: https://doi.org/10.1002/2016WR019987

[93] A. Hoblitzell, M. Babbar-Sebens, and S. Mukhopadhyay, "Fuzzy and deep learning approaches for user modeling in wetland design," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2016, pp. 002 133–002 138.

[94] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning," in *OSDI*, vol. 16, Nov 2016, pp. 265–283.

[95] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *CoRR*, vol. abs/1412.6980, Dec 2014. [Online]. Available: http://arxiv.org/abs/1412.6980

[96] S. Zheng, Y. Song, T. Leung, and I. Goodfellow, "Improving the Robustness of Deep Neural Networks via Stability Training," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2016.

[97] Y. Gal and Z. Ghahramani, "Dropout as a bayesian approximation: Representing model uncertainty in deep learning," in *Proceedings of The 33rd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, M. F. Balcan and K. Q. Weinberger, Eds., vol. 48. New York, New York, USA: PMLR, 20–22 Jun 2016, pp. 1050–1059. [Online]. Available: http://proceedings.mlr.press/v48/gal16.html

[98] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[99] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, Sep 2013. [Online]. Available: https://doi.org/10.1177/0278364913495721

[100] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529 EP –, Feb 2015. [Online]. Available: https://doi.org/10.1038/nature14236

[101] M. Hessel, J. Modayil, H. Van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver, "Rainbow: Combining improvements in deep reinforcement learning," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[102] M. C. Machado, M. G. Bellemare, E. Talvitie, J. Veness, M. J. Hausknecht, and M. Bowling, "Revisiting the Arcade Learning Environment: Evaluation Protocols and Open Problems for General Agents," *CoRR*, vol. abs/1709.06009, Sep 2017. [Online]. Available: http://arxiv.org/abs/1709.06009

[103] D. J. Russo, B. V. Roy, A. Kazerouni, I. Osband, and Z. Wen, "A Tutorial on Thompson Sampling," *Foundations and Trends® in Machine Learning*, vol. 11, no. 1, pp. 1–96, 2018. [Online]. Available: http://dx.doi.org/10.1561/2200000070

[104] S. Parisi, S. Ramstedt, and J. Peters, "Goal-driven dimensionality reduction for reinforcement learning," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep 2017, pp. 4634–4639.

[105] S. Bitzer, M. Howard, and S. Vijayakumar, "Using dimensionality reduction to exploit constraints in reinforcement learning," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2010, pp. 3219–3225.

[106] I. Bar-Gad, G. Morris, and H. Bergman, "Information processing, dimensionality reduction and reinforcement learning in the basal ganglia," *Progress in Neurobiology*, vol. 71, no. 6, pp. 439–473, Dec 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0301008203001928

[107] T. Rusch, C. W. Korn, and J. Gläscher, "A Two-Way Street between Attention and Learning," *Neuron*, vol. 93, no. 2, pp. 256–258, Jan 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0896627317300065

[108] H. Wu and X. Liu, "Double Thompson Sampling for Dueling Bandits," in *Advances in Neural Information Processing Systems 29*, D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, Eds. Curran Associates, Inc., 2016, pp. 649–657. [Online]. Available: http://papers.nips.cc/paper/6157-double-thompson-sampling-for-dueling-bandits.pdf

[109] O. Granmo, "Solving two-armed Bernoulli bandit problems using a Bayesian learning automaton," *International Journal of Intelligent Computing and Cybernetics*, vol. 3, no. 2, pp. 207–234, Jun 2010. [Online]. Available: https://doi.org/10.1108/17563781011049179

APPENDICES

A. DATABASE DESIGN

Tables

Table A.1.: CBM Tables

CBM Tables
‘igmi2db‘.‘cbm‘
‘igmi2db‘.‘cbmInitial‘
‘igmi2db‘.‘cbmInitialF0‘
‘igmi2db‘.‘cbmInitialF1‘
‘igmi2db‘.‘cbmInitialF2‘
‘igmi2db‘.‘cbmInitialF3‘
‘igmi2db‘.‘cbmInitialF4‘
‘igmi2db‘.‘computerclusterinfo‘

Table A.2.: Fitness Function Tables

Fitness Function Tables
‘igmi2db‘.‘f0‘
‘igmi2db‘.‘f1‘
‘igmi2db‘.‘f2‘
‘igmi2db‘.‘f3‘
‘igmi2db‘.‘f4‘
‘igmi2db‘.‘f5‘
‘igmi2db‘.‘f6‘

Table A.3.: SDM Tables

SDM Tables
‘igmi2db‘.‘sdm‘
‘igmi2db‘.‘sdmData‘
‘igmi2db‘.‘sdmModellingData‘
‘igmi2db‘.‘sdmModellingDataRandomdata‘
‘igmi2db‘.‘sdmModellingDataRandomdatacumnew‘

Table A.4.: User Tables

User Tables
‘igmi2db‘.‘users‘
‘igmi2db‘.‘users_activity‘
‘igmi2db‘.‘usersFeedback‘
‘igmi2db‘.‘usersFeedbackTiming‘
‘igmi2db‘.‘usersFeedbackTimingNew‘
‘igmi2db‘.‘userstartexperimentsession‘
‘igmi2db‘.‘userstats‘
‘igmi2db‘.‘userstatsFeedbackTiming‘
‘igmi2db‘.‘userstatsFeedbackTimingNew‘
‘igmi2db‘.‘userwatershedmapping‘
‘igmi2db‘.‘userloginsession‘

Table A.5.: WRESTORE and IGAMI2 Tables

WRESTORE and IGAMI2 Tables
‘igmi2db‘.‘hdmarchiveNondominated‘
‘igmi2db‘.‘igami2EventMaster‘
‘igmi2db‘.‘igami2EventServer‘
‘igmi2db‘.‘watershed‘
‘igmi2db‘.‘wrestorefriends‘
‘igmi2db‘.‘wrestoregrouprequests‘
‘igmi2db‘.‘wrestoregroups‘
‘igmi2db‘.‘wrestorelogs‘

Table A.6.: Action Tables

Action Tables
‘igmi2db‘.‘abortSearch‘
‘igmi2db‘.‘abortUserThread‘
‘igmi2db‘.‘abortsearch‘
‘igmi2db‘.‘newUser‘
‘igmi2db‘.‘newfeedlikes‘
‘igmi2db‘.‘newscomments‘
‘igmi2db‘.‘newsfeed‘
‘igmi2db‘.‘newuser‘
‘igmi2db‘.‘takefeedback‘
‘igmi2db‘.‘takefeedbackModified‘
‘igmi2db‘.‘takefeedbacknew‘
‘igmi2db‘.‘takefeedbackupdate‘
‘igmi2db‘.‘takefeedbackwarmup‘

Table A.7.: Miscellaneous Tables

Miscellaneous Tables
‘igmi2db‘.‘fusiontableTest_1‘
‘igmi2db‘.‘hdmarchiveChildren‘
‘igmi2db‘.‘kendallstats‘
‘igmi2db‘.‘kendallstatsUserData‘
‘igmi2db‘.‘newuserParamters‘
‘igmi2db‘.‘sessionInfo‘
‘igmi2db‘.‘shareddesignmapping‘
‘igmi2db‘.‘biasindvdata‘

Stored Procedures

Table A.8.: INSERT Stored Procedures

INSERT Stored Procedures
‘igmi2db‘.‘InsertComment‘
‘igmi2db‘.‘InsertFriendRequest‘
‘igmi2db‘.‘InsertGroupRequest‘
‘igmi2db‘.‘InsertNewGroup‘
‘igmi2db‘.‘InsertNewsFeed‘
‘igmi2db‘.‘InsertRatings‘
‘igmi2db‘.‘InsertSharedDesign‘
‘igmi2db‘.‘InsertUserDetails_Updated‘
‘igmi2db‘.‘InsertUserWatershedMapping‘

Table A.9.: GET Stored Procedures

GET Stored Procedures
‘igmi2db‘.‘GetAllGroups‘
‘igmi2db‘.‘GetAllUserNames‘
‘igmi2db‘.‘GetFriendList‘
‘igmi2db‘.‘GetFriendListForDropdown‘
‘igmi2db‘.‘GetFriendRequests‘
‘igmi2db‘.‘GetGroupJoiningRequests‘
‘igmi2db‘.‘GetGroupList‘
‘igmi2db‘.‘GetGroupUserDetails‘
‘igmi2db‘.‘GetUserGroupList‘

Table A.10.: DELETE Stored Procedures

DELETE Stored Procedures
‘igmi2db‘.‘DeleteComment‘
‘igmi2db‘.‘DeleteFriendRequest‘
‘igmi2db‘.‘DeleteGroupDetails‘
‘igmi2db‘.‘DeleteGroupRequest‘
‘igmi2db‘.‘DeleteUserWatershedMapping‘

Table A.11.: SELECT, SEARCH, and UPDATE Stored Procedures

SELECT, SEARCH, and UPDATE Stored Procedures
‘igmi2db‘.‘SearchFriends‘
‘igmi2db‘.‘SearchGroups‘
‘igmi2db‘.‘selectAllDataUser‘
‘igmi2db‘.‘selectAllDataUserComplete‘
‘igmi2db‘.‘UpdateFriendRequest‘
‘igmi2db‘.‘UpdateGroupDetails‘
‘igmi2db‘.‘UpdateUserDetails_Updated‘

Table A.12.: Miscellaneous Stored Procedures

Miscellaneous Stored Procedures
‘igmi2db‘.‘AcceptGroupRequest‘
‘igmi2db‘.‘CheckDuplicateGroupName‘
‘igmi2db‘.‘CheckDuplicateUserName‘
‘igmi2db‘.‘DislikeNews‘
‘igmi2db‘.‘LikeNews‘
‘igmi2db‘.‘movedata‘
‘igmi2db‘.‘purgeUserData‘
‘igmi2db‘.‘RejectGroupRequest‘
‘igmi2db‘.‘warmup‘
‘igmi2db‘.‘whileLoopProc‘

B. SOFTWARE DESIGN

IGAMI2

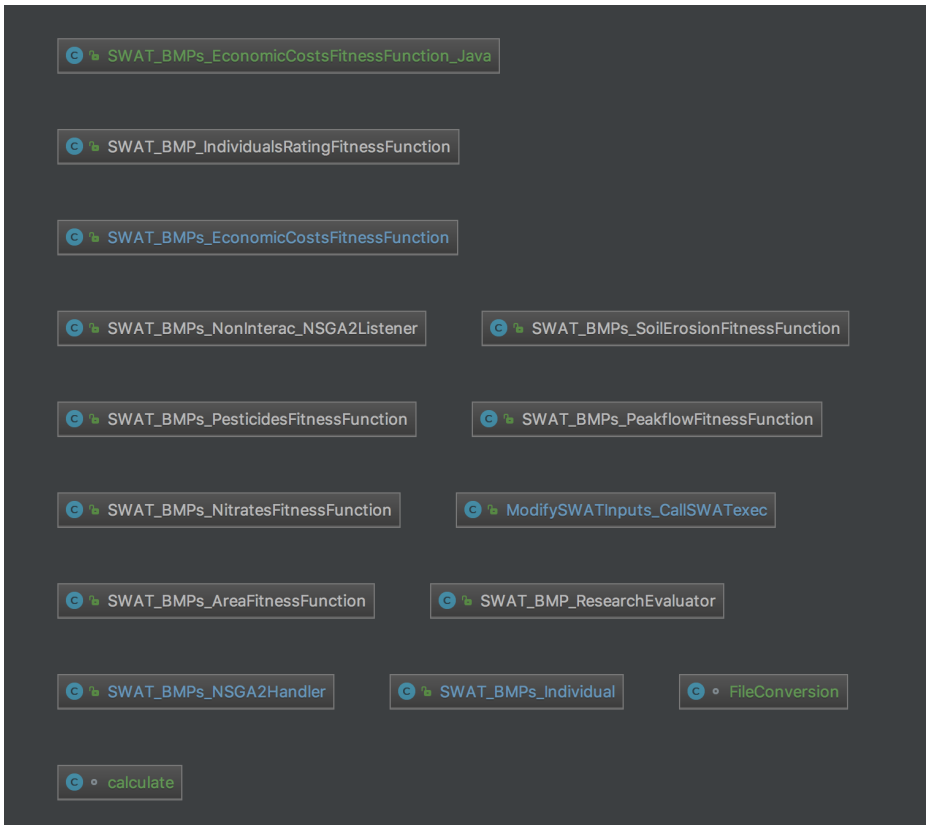

Figure B.1.: IMAGI2

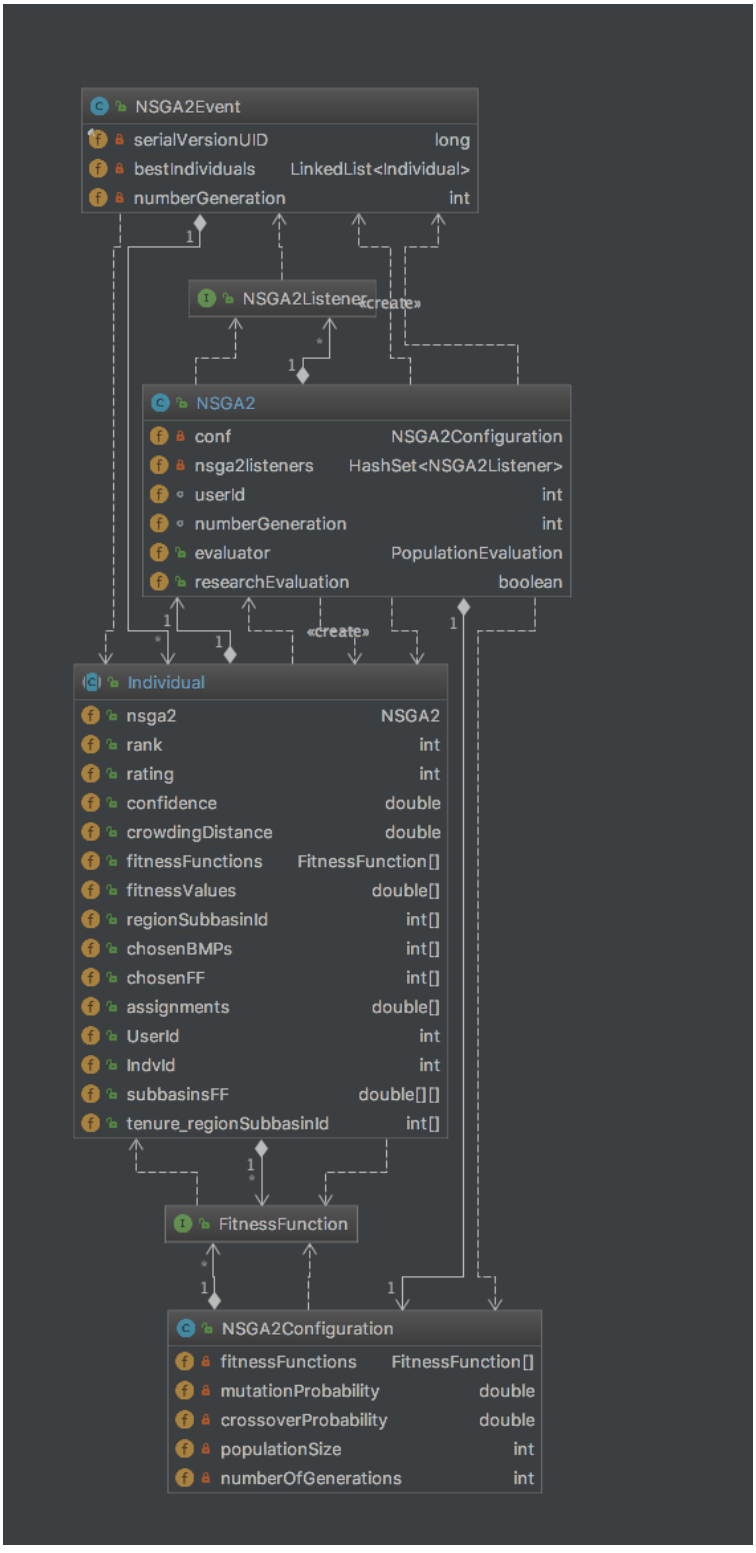

Figure B.2.: NSGA2

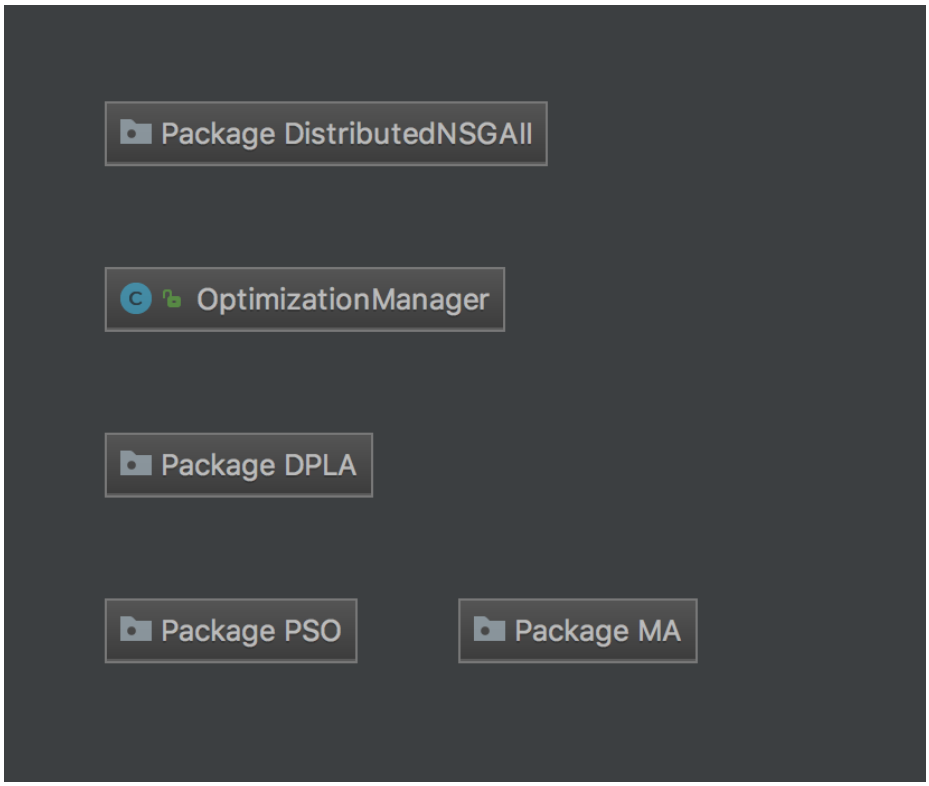

Figure B.3.: Select Packages

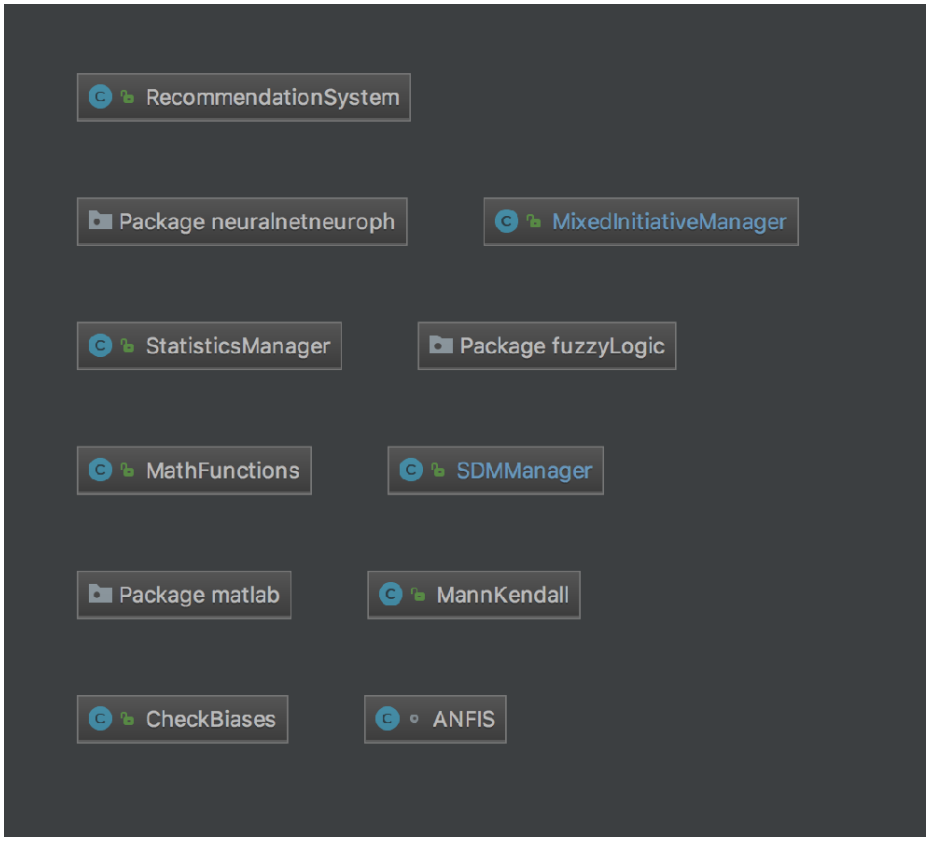

Figure B.4.: RecommendationSystem

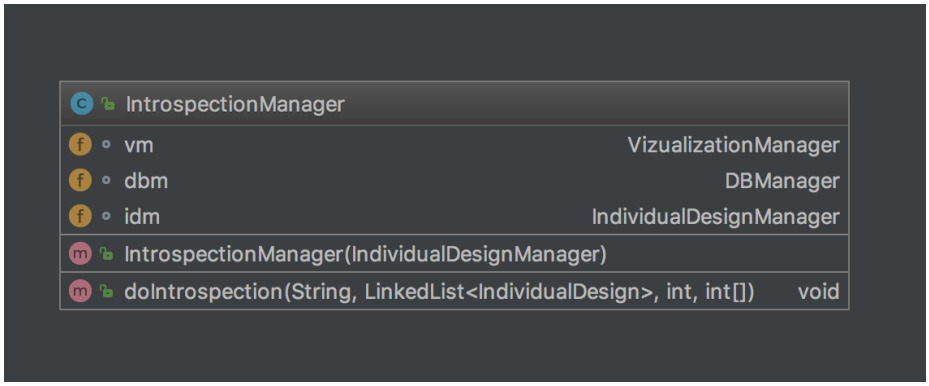

Figure B.5.: IntrospectionManager

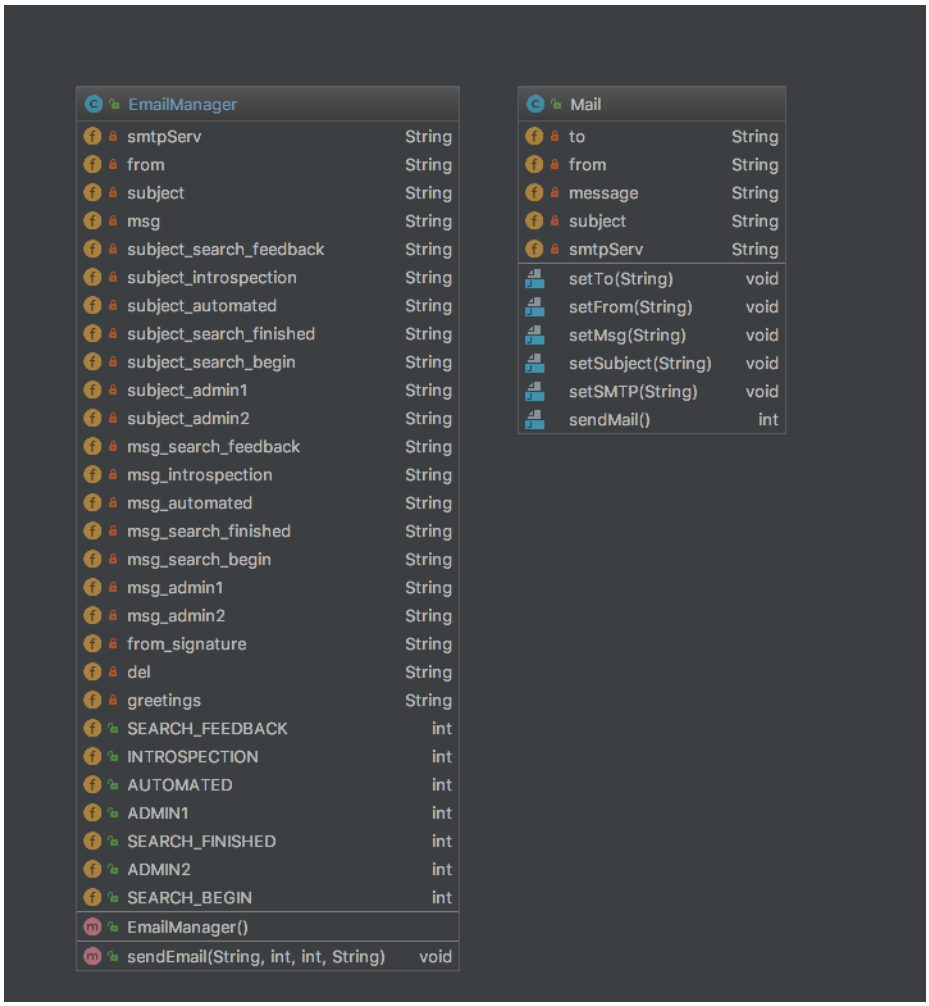

Figure B.6.: EmailManager

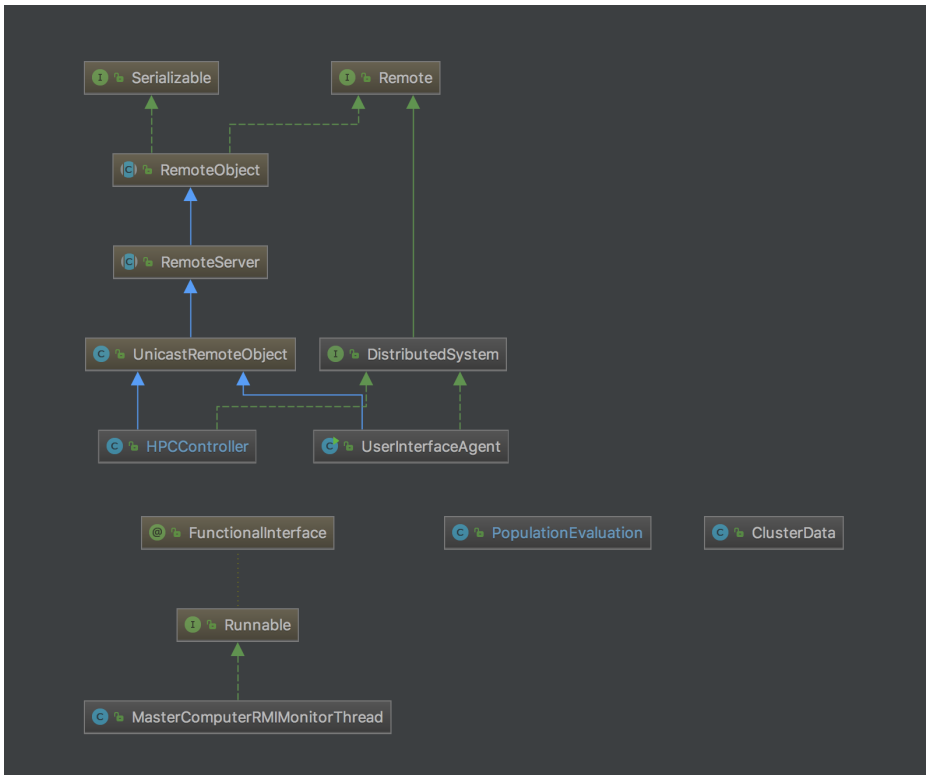

Figure B.7.: Interfaces

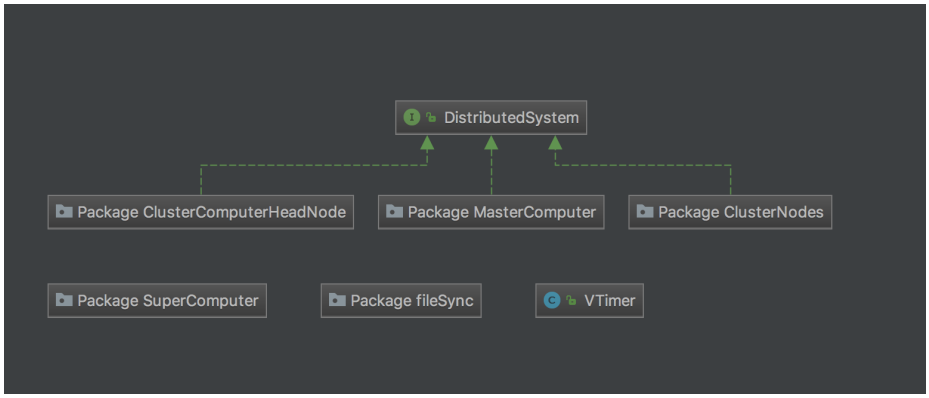

Figure B.8.: DistributedSystem

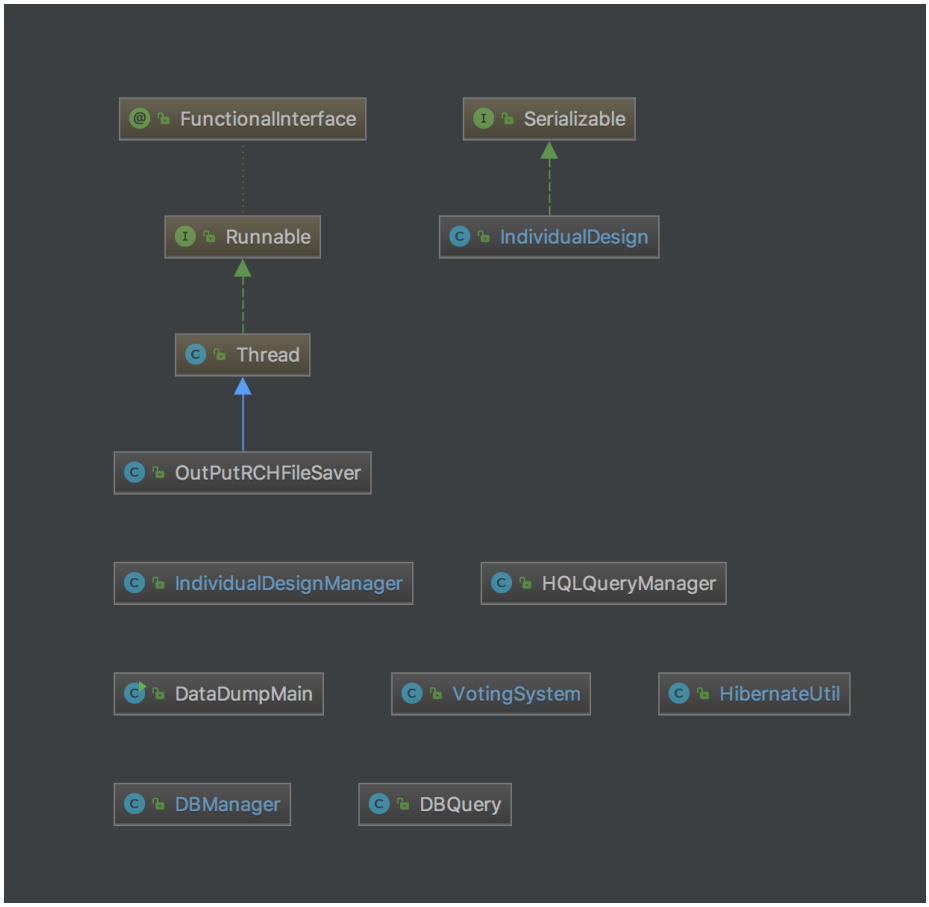

Figure B.9.: Additional Interfaces

VITA

VITA

Andrew Hoblitzell

Summary

Enthusiastic and experienced technology worker seeking the opportunity to apply problem-solving skills in a challenging and engaging IT environment.

Core Strengths

- Big Data/Full Hadoop Stack/NoSQL
- Machine Learning/Data Mining
- Java/C#
- Data Science/Intelligent Systems

Specific Technology Skills

Languages: Java*, Python*, C#*, Ruby*, C# ASP.NET*, .Net Framework*, SQL*, XML/XSD*, Visual Basic*, HTML, JavaScript, PERL, Python, R, Scala/Spark

Software: IntelliJ*, Eclipse*, Microsoft Visual Studio*, Oracle 11G*, Microsoft SQL Server, Apache Hadoop, Apache Kafka/Pig/Hive/HBase/Phoenix, Apache Spark, Cascading, Dreamweaver, Microsoft Office* (Word, Excel, PowerPoint, Access), Microsoft Windows*, Redhat Linux*, Solaris Unix* (*- well acquainted)

Work Experience

Senior Member of Technical Staff, Salesforce January 2016 - Present

Working on Salesforce Einstein to:

- Manage integrations with Google, IBM, and external partners for enhanced prediction capacity.
- Design and implement scalable, reliable, and intelligent software for e-mail marketing predictions
- Working on the Salesforce Marketing Cloud's Big Data team using a Hadoop stack, HBase, Kafka, etc. to build Marketing Cloud platform services.
- Design and implement scalable, reliable, and efficient software that performs search and analytics over multi-billion record distributed datasets

  Overall responsibilities:
- Develop and follow team coding best practices and participate in peer code review
- Ensure quality software which meets integration and unit test coverage metrics.
- Create and maintain development tasks in JIRA for tracking work
- Coordinate and participate in release and sprint planning meetings and daily stand-ups
- Perform peer code reviews to uncover potential bugs and ensure code conforms to development standards
- Collaborate with other scrum teams to ensure alignment
- Troubleshoot escalations and provide root cause analysis when necessary

Developer Senior Sensitive, Anthem NGS March 2012 - December 2015

Working on the US Government's National Fraud Prevention Program which is responsible for saving tens of millions of dollars per year. The main focuses I have been

responsible for is ETL and the creation of 'big data' analytical software. Primarily responsible for:

- Writing source code (with Pig, Hive, Cascading, etc.), adapting existing components, compiling, linking and testing the developed components as units creating test stubs and test data as necessary
- Maintaining healthcare business domain knowledge necessary for job responsibilities, as well as a background in predictive modeling Fixing defects identified during unit testing and re-executing unit tests to verify the expected behavior
- Ensuring design meets performance, usability, reliability and scalability requirements in addition to the functional requirements
- Performing analysis and conceptual design on moderate to complex system components

Software Engineer, Availity 2010 - 2012

- Involved in backend integration and in object-oriented component design and development for real-time claim adjudication projects Used .C/#, ASP.NET, .NET Framework, XML/XSD, Oracle, and MSMQ in ETL process for claims and patient statement generation and integrated with other internal projects
- Interacted with business analysts, client account managers, quality assurance workers, and project managers through HelpDesk, Project Tracking System, and other technologies to improve medical practice administrative workflows

University Fellow, Purdue 2008 - Ongoing

JTE Technologies, Atlanta, GA

- Worked for the Business Intelligence Team using C# ASP.NET and the Business Objects Enterprise XI .NET Software Development Kit
  - Wrote use cases
  - Developed administrative web forms
  - Performed simple unit and regression testing

  - Composed documentation
  - Utilized Microsoft SharePoint for collaboration purposed

Computer Program Specialist, IU Cancer Center Summer 2007

- Extracted publicly available bioinformatics data from an online PDB database
- Wrote GCC/Unix programs to transform the data into a suitable format
- Perform relevant computations from a common lab framework for usage in a dry lab setting

Education

Purdue University, West Lafayette, IN

University Fellow

Doctorate of Science in Computer Science (December 2019)

Indiana University-Purdue University Indianapolis, Indianapolis, IN

University Fellow, Thesis: Transitive Biomedical Literature Mining

Master of Science in Computer Science (August 2010)

Memberships

Data Science Indy (Co-organizer) Indy Big Data (Member)

Academic Work

Hoblitzell, Andrew, Meghna Babbar-Sebens, and Snehasis Mukhopadhyay. " Multi-Criteria, Interactive Optimization for Design of Watershed Plans." INFORMS Annual Meeting, 2016. Nashville, TN. 2016 DAS Best Practice Finalist Award

Hoblitzell, Andrew, Meghna Babbar-Sebens, and Snehasis Mukhopadhyay. "Fuzzy and deep learning approaches for user modeling in wetland design." Systems, Man, and Cybernetics (SMC), 2016 IEEE International Conference on. IEEE, 2016.

Omkar Tilak, Andrew Hoblitzell, Snehasis Mukhopadhyay, Qian You, Shiaofen Fang, Yuni Xia, Joseph Bidwell, Multi-Level Text Mining for Bone Biology, Concurrency and Computation: Practice and Experience. June 2011. DOI: 10.1002/cpe.1788

Andrew Hoblitzell, Snehasis Mukhopadhyay, Qian You, Shiaofen Fang, Yuni Xia, and Joseph Bidwell. 2010. Text mining for bone biology. In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing (HPDC '10). ACM, New York, NY, USA, 522-530. DOI=10.1145/1851476.1851552 `http://doi.acm.org/10.1145/1851476.1851552` (nearly 40% acceptance rate)

Hoblitzell, A. (2010). Biomedical literature mining with transitive closure and maximum network flow. Retrieved August 10, 2015.