

CBR for Modeling Complex Systems

Rosina Weber¹, Jason M. Proctor², Ilya Waldstein³, and Andres Kriete^{4,5}

^{1,2,3} College of Information Science & Technology, Drexel University

⁴ School of Biomedical Engineering, Science and Health Systems, Drexel University

⁵ Drexel and Coriell Bioinformatics Initiative, Coriell Institute for Medical Research
{¹rw37,²jp338,⁴ak3652}@drexel.edu,³sainet@snip.net

Abstract. This paper describes how CBR can be used to compare, reuse, and adapt inductive models that represent complex systems. Complex systems are not well understood and therefore require models for their manipulation and understanding. We propose an approach to address the challenges for using CBR in this context, which relate to finding similar inductive models (solutions) to represent similar complex systems (problems). The purpose is to improve the modeling task by considering the quality of different models to represent a system based on the similarity to a system that was successfully modeled. The revised and confirmed suitability of a model can become additional evidence of similarity between two complex systems, resulting in an increased understanding of a domain. This use of CBR supports tasks (e.g., diagnosis, prediction) that inductive or mathematical models alone cannot perform. We validate our approach by modeling software systems, and illustrate its potential significance for biological systems.

1 Introduction

This paper explores the contribution of the CBR methodology for the modeling task, particularly when the system (e.g. biological, organizational, computational) to be modeled is complex i.e., not well understood or not easily accessible. We envision using CBR to recommend a model to a previously unknown system based on its similarity to previously recorded systems and their adopted models. *Modeling* is the task concerned with creating a description of a system with the purpose of understanding or predicting its functioning and/or effects. When data is available, models can be created with inductive methods. When theory is available, models can be created with mathematical methods. When neither is available, we propose reusing models through CBR. The role we propose for CBR in the modeling task is one of an aggregator or manager of data and knowledge pertinent to the case-based recommendation of models – not as an alternative to inductive or mathematical models.

This paper's intended contribution is to propose an approach to assess similarity between complex systems and between inductive models, and to demonstrate that a suitable inductive model can be recommended to represent a system on the basis of the system's similarity to other systems. As a result, CBR can be used as the underlying methodology for reasoning with complex cases, whose problems are complex systems and whose solutions are their models. This use of CBR will allow

the performance of tasks such as prediction and diagnosis; but even more importantly, it will leverage the understanding of the systems it will model.

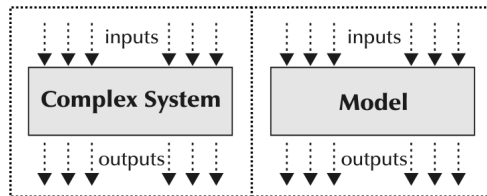


Fig. 1. Models represent complex systems

The CBR methodology can combine knowledge from different sources into one aggregated reasoning task to propose a solution. When domain knowledge is incorporated into the design of a case base that recommends models to represent complex systems, it represents a potential advantage over selecting models without domain knowledge.

Determining the quality of a model also requires domain knowledge. Therefore, when a model is proposed and its suitability is revised and confirmed, the case base learns a new case and measures of quality of how a model represents a system can be verified (i.e. confirmed or rejected). Besides the individual power of one more case to improve a future recommendation, this new case becomes a new (properly positioned) piece of the puzzle, allowing a better understanding of the domain, and potentially advancing the field. The ability to assess similarity between complex systems and between inductive models is critical to allow the combination of domain knowledge with the manipulation and understanding of complex systems.

Recommending models for systems that are not well understood or not easily accessible encompasses uncertainty. This uncertainty is associated with the suitability of a model to describe a system. After each confirmed solution, the enhanced understanding of the domain problem is expected to result in reduced levels of uncertainty.

The significance of the approach described in this paper is illustrated by its applicability in modeling software programs and biological systems. Both in software engineering and in bioinformatics, modeling methods used are mathematical or inductive, but neither can be leveraged into a system-wide understanding of the role of models and their interplay with the environment. The conception of a computational approach that benefits from the power of inductive modeling and also takes advantage of analogical reasoning has the potential to drastically improve the performance of tasks such as prediction and diagnosis, and even enhance the understanding of these systems.

1.1 Two Target Problems: Software Programs and Biological Systems

We describe the problems of modeling software programs and biological systems because they are sufficiently similar so that one can serve as a proof of concept for the other. We have already implemented the first one, modeling software programs, and we use it to understand research challenges and test strategies to address them. The second problem seems more significant because of its potential benefits to human health.

Both software programs and biological systems process inputs to produce outputs. In software programs, *inputs* and *outputs* are the terms used to describe the values entered and results from programs' computations. An individual's biological system

receives inputs from the environment (e.g. nutrition) and produces as a result health outcomes. In fact, a reasonable explanation for the functioning of cells and genes is that they follow programs to produce an outcome like processed data. Both problems are complex systems that require system modeling. In both problems, the essence of the modeling task is to represent input-output analysis (Fig. 1). Given the suitability of using artificial neural networks to model input-output analysis, they are chosen to model both software (e.g. [18]) and biological systems (e.g. [14]).

The problem we focus on in the software engineering domain is to model software programs with the purpose of generating test cases for software testing. This may be useful because it is easier to manipulate a model than it is to manipulate a real and complete program and because the entire program's details may not always be available. The model can be built inductively by the analysis of randomly generated inputs and the corresponding resulting outputs [18].

In biology, we focus on modeling an individual's biological system with the purpose of predicting health outcomes based upon dietary inputs. This can improve medical understanding, helping individuals predict and achieve desired health outcomes. The model is necessary because it is impossible to submit each individual to different inputs to study what the outputs would be. The model can only be built by comparing and combining models generated with data from other individuals and partial data from the target individual. This is where a computing platform requires analogical reasoning for the modeling task: *to help find a quality model to represent an individual, it is necessary to assess genetic similarities to make use of biological assumptions* (e.g. twins may have similar susceptibility to environment). Consequently, a reasoning platform to model biological systems has to be able to manipulate inductive models and assess similarity between them.

There are uncertainties in both systems. We may know the programming language and we may be able to infer how a program might have been written; but even if we have the code, it is not clear how to use it to define good test cases for its testing. In human biological systems, we may know the genetic constitution of an individual and may have the expression of genes from blood cells and some other accessible organs, but there is always uncertainty with respect to the remaining cells as long as the human individual is alive. Gene expression varies with age [21][23], so even if we know the current expression of genes in some cells, there is uncertainty as to what the expression will be in the future.

Improvements in software testing methods can be significant. The cost of poor software testing is estimated to reach up to 60 million US dollars annually [11]. In the domain of biology, the recent availability of the human genome and knowledge of pathways has created a demand for computing solutions to understand the behavior of molecular processes. This is an area with potential high payoffs in human health but where data is still expensive or impossible to obtain. It is therefore necessary that these computing solutions are able to leverage existing data to support, manipulate, complement, and explain phenotypical and medical facts. These same computing infrastructures can recommend testing methods to support high quality software.

In Section 2 we describe a case-based platform applied to model software programs. Section 3 proposes our approach to overcome the main obstacle to apply CBR: *what makes a system similar to another such that we can reuse their models?* This approach is validated in Section 4. It is then used as the basis to design a second

platform, which we discuss in Section 5 to model biological systems. Section 6 discusses related work and Section 7 presents concluding remarks and future work.

2 CBR for Modeling Software Programs

The application we describe here models software programs with the purpose of generating test cases. The current application integrates a case-based framework [22] into a system (CI-Tool) [4] that uses computational intelligence methods (e.g., inductive [11][18]) to generate test cases. Although CBR itself does not perform the modeling task, it creates cases to improve the overall quality of the system [22].

We limit our presentation of the software program modeling system to one inductive method: artificial neural networks (ANN). The elements we discuss when representing cases with ANN are also present when using other inductive modeling methods. Fig. 2 depicts the modules of the software program modeling system. A data mart retains data and functions, managing the communications between the modules. A software program is the input to the system; the ANN module creates an inductive model for it. The individual case base is dedicated to storing cases where problems describe features of software programs, solutions describe elements of an ANN, and the outcome describes the accuracy of the ANN as a model of the software program.

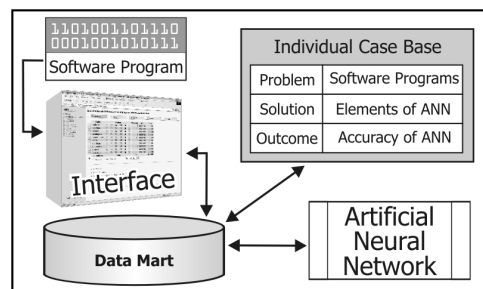


Fig. 2. Software Program Modeling system

Elements of an ANN include its configuration parameters (e.g. learning rate, training dataset, pruning accuracy). We assess model quality based on the evaluation of the ANN training. Our model's accuracy is obtained through the average error rate between the expected outputs and actual output with the final weight matrix.

There are two problems when using inductive models like ANN as part of case representation; they both stem from the presence of random functions in the computation of ANN. One problem is that one parameter configuration can produce more than one accuracy score. In order to ensure we use values that are sufficiently representative, we train each ANN ten times, using the same training data, and use the average accuracy.

The second problem is that multiple parameter configurations can produce similar accuracies, making it impossible to guarantee one parameter configuration to be optimal. To address this problem, we adopted the notion of a configuration of *good quality*. Starting from the default parameter configuration defined in the CI-Tool [4][18], we evolve configurations with a genetic algorithm [16]. We refer to the configuration resulting from this process as being of *good quality*.

The purpose of the CBR module is to recommend the reuse of an ANN for modeling previously unknown software programs. Here is where the challenges become apparent. In order to reason, the case-based reasoner has to be able to assess the similarity between different software programs and between different ANN models.

3 Strategy to Overcome Challenges for Using CBR

The CBR system for modeling software programs has revealed the most important challenges for using a CBR platform to support such a modeling task. First, because the system we want to model is not well understood, how can we determine what makes one system similar to another? Second, how can we assess similarity between inductive models such as ANN? In other words, if we do not understand the cases well, *how can we find similar solutions for similar problems?*

To allow the case-based platform to recommend a model to describe a new (previously unknown) problem (i.e. system), it may seem that we need to first identify similar problems. However, if we find similarity between problems, we would not know which similar elements are relevant for making two problems amenable to be solved with the same solution. Therefore, we need to first identify similar solutions. Once similar solutions are identified, that is, solutions that would require minimal adaptation to be reused by another problem, we can assume that there may be similarity between problems that have been solved with similar solutions. The main challenge we focus on is to learn what makes one problem similar to another, such that the solutions can be reused.

The first step of our approach is to cluster existing problem-solution pairs (cases) based on features of the *solutions*. Note that for these problems the same exact solution may not appear in more than one case. Once the cases are organized in clusters, the second step is to identify the subset of *problem* features that support these clusters. If we find these features, then it means that they can be used to guide similarity between problems whose solutions can be reused. The final step is to use these features to define the similarity measure across cases. We base our strategy on three assumptions.

Our *first assumption* is that similar solutions can be recognized by individual similarities between elements (i.e. features) of a detailed representation for the solution. Therefore, if the solution is an inductive model, two solutions are similar if the majority of their representational elements are similar; and they are dissimilar otherwise. In an ANN, for example, some of these elements are the values assigned to parameters in its training (e.g. number of epochs). Note that these elements do not assess how well the inductive method models the system. These elements are exclusively the ones that will be used in the reuse step of the CBR cycle.

Because it is important to define similarity and dissimilarity of solutions; in the absence of domain knowledge, we chose an unsupervised learning method to group solutions based on the values of their elements: clustering. Clustering is a well-known method to organize data elements in groups based on attribute values that describe the elements. It produces a set of clusters that group elements that are similar to each other within the same cluster and dissimilar to elements in different clusters.

Our *second assumption* is that, for the task in question, similar problems are the ones that share similar solutions. The first step produces clusters of cases based on the similarity of the solutions. However, to employ the CBR paradigm, we need to determine what makes *problems* similar so that we can reuse their solutions. Hence, the second step is to identify the subset of problem features that support these clusters.

We propose to perform the second step with discriminant analysis (DA). DA is a statistical method that defines boundaries that separate the data into categories to analyze the predictive value of a set of independent variables. Stepwise DA is a variation that initially considers all of the independent variables (our problem features), but removes those that do not make a significant contribution to the overall predictive ability, including those that are closely correlated with other variables. In short, it determines the predictive features from the problem descriptions and eliminates non-predictive features.

Our *third assumption* is that we can use the discriminant functions produced by the DA to assess the similarity between a new, unseen problem and the previously recorded problems. The discriminant functions describe the location of features with respect to each cluster. The rationale of using these functions for similarity is to assess how similarly localized features of a target case are to the features of each of the previously recorded candidate cases. There is always one fewer function than the number of clusters, so if there are 5 clusters, each tuple will be a vector of 4 discriminant function values. The selection step indicates the most similar case by finding the closest tuple using Euclidean distance. This step yields the best matching case, whose solution (i.e. model) we can reuse to describe the target problem (i.e. system). It is important to note that reusing inductive models is not trivial. In fact, we do not reuse the exact model, but the strategy (i.e. parameter configuration) adopted in the most similar case.

4 Validation

In this section, we evaluate the hypothesis that our approach to similarity assessment can support the recommendation of a model to a previously unknown problem with an accuracy that is as high as the accuracy of the models recorded in the case base. That is, it should, on average, produce accuracy that is not significantly lower. We use two metrics for the comparison: $Accuracy_{\text{ORIG}}$ is the average original accuracy of the models recorded in the case base; $Accuracy_{\text{CBR}}$ is the average accuracy obtained with the parameter values recommended with our CBR approach.

4.1 Dataset

This study uses twenty-one (21) software programs that constitute our cases. We have identified 23 features to describe these problems. Note that domain knowledge does not indicate what makes two programs similar for the purposes of recommending an inductive model for them. Thus, we include all the features we could determine and expect the approach to indicate the relevant features. Table 1 shows 4 out of the 23 problem features used in the study and values for these features in three software programs T01, T02, and T03. The solutions for these 21 cases were obtained with the system described in Section 2; the quality of the solutions is substantiated by the method in [16]. The solution features consist of elements of ANN such as configuration parameter values and the dataset used for the training. Cases also have an outcome, which indicates the resulting accuracy of the ANN training.

Table 1. Subset of features for three software programs

Features		Testbed	T01	T02	T03
Problem Features	No. of Input Variables		3	5	4
	No. of Program Variables		0	1	0
	Highest Max of Input Variables Range		1000	10	25000
	No. of Conditionals		1	0	0
Solution Features	Training Accuracy		91	91	94
	Pruning Accuracy		93	91	90
	Learning Rate		0.48	0.54	0.47
Outcome Feature	Accuracy		93.4%	90.3%	86.8%

4.2 Methodology

Our methodology is to employ leave-one-out cross validation (LOOCV) across the 21 software programs. At each iteration, the 20 remaining cases are clustered based on linearly normalized solution features, using hierarchical clustering with squared Euclidean distance as the similarity metric. Then, we perform stepwise DA on the problem features to obtain the set of coefficients that describe each cluster. We apply these coefficients to the feature values of all 21 cases (20 known cases and 1 target) to compute tuples to assess their similarity and obtain the closest case. The values used to configure the ANN in the closest case are reused to train a new ANN for the target problem. We use these parameters to train ten ANNs to compute the Accuracy_{CBR}. Note that we use the same training data for these ten runs that we use for the ten runs to compute Accuracy_{ORIG}.

4.3 Results

The results support our hypothesis that our approach can recommend a model that is as accurate as the models originally recorded in the case base 71.4% (15 out of 21) of the time. Table 2 shows the distribution of accuracy comparisons using ANOVA between our two metrics Accuracy_{CBR} and Accuracy_{ORIG} for the 21 cases. We define significance at $p < 0.05$, and present the averages (Avg) and standard deviations (SD) of the p values for each category in Table 2.

Table 2. Summary of accuracy comparisons

Performance of Accuracy _{CBR}	No. of Cases	% to 21	Avg p	SD p
Accuracy _{CBR} is significantly higher	2 Cases	9.5	0.001	0.001
Accuracy _{CBR} produces no significant difference	13 Cases	61.9	0.329	0.224
Accuracy _{CBR} is significantly lower	6 Cases	28.6	0.002	0.002

4.4 Discussion

The consistency of the results indicates that our approach can support the use of CBR to recommend inductive models to represent complex systems. Indirectly, they indicate our assumptions (Section 3) were sound. In this discussion we attempt explore the validity of our approach and investigate ways of improving these results.

The DA identified 13 problem features that contributed to the similarity calculation in every iteration of the LOOCV, and 6 more that were used in fewer than half. Of the 171 possible pairs of these features (each of the 19 features compared symmetrically with the 18 others), only 23 pairs showed any significant correlation at $p < 0.05$, and these were primarily between the infrequently used features. This shows that the features are independent, so the discriminant functions are reliable for the overall validation.

Our choice for the clustering analysis revealed satisfactory results. We confirmed it by observing that the data in our dataset had natural groupings, making it amenable to clustering. The evidence is that for 18 (85.7%) of the LOOCV iterations, the clusters obtained with 20 cases were identical (except for the presence of the target case). In the 3 (14.3%) iterations when they varied, only one or two cases changed clusters. There were five stable clusters identified, and these were confirmed using other distance metrics as well as k -means tests for 3 through 7 clusters.

Our choice for the discriminant analysis to capture the relative importance of the features and use it for similarity assessment also proved to be satisfactory. DA is a way to represent the organization of features in the discriminant space in respect to the clusters. Thus, our approach implies that not only the clusters but the relative position with respect to a cluster are relevant for similarity assessment. For 20 (out of the 21) iterations, the reused case (the closest according to our approach) was originally a member of a different cluster than the target case for the iteration. This indicates that clusters could not have been used as outcome classes and that the success of the approach also depends upon the relative position of each case in relation to the cluster.

In order to further investigate the use of clusters, we applied gradient descent (GD) and extracted the relative importance of the features for the entire case base, using the clusters to measure classification accuracy. The results generated a case base that produced an accuracy with LOOCV of 14.3%, i.e. only 3 times the most similar case received the correct classification. In addition, only 3 times (for different cases) the results with the GD weights coincided with the closest case recommended by our approach.

We wanted to employ an algorithm that could reveal a subset of features (and possibly their relative weights) that could be used in the entire case base for similarity assessment. Such results would contribute new knowledge to the domain. In our dataset, for example, we would be able to indicate how to compare two software systems to reuse software testing approaches. However, at least with our dataset, there was no subset of features that could justify the reuse of models with the same accuracy as our proposed approach.

Another potential source of improvement is the refinement of the reuse step. In this study, we did not contemplate the second closest case as a candidate for reuse, and a brief analysis showed the second closest case would have improved our results.

The similarity between solutions could have also been explored by using extracted rules as an explicit representation of the ANN. On examination, there was no correlation between rules and the resulting accuracy. The rules are the ANN's restatement of the problem, but they do not necessarily reflect their quality.

Finally, we believe that the two iterations where the recommended model produced accuracy significantly higher than the one previously recorded indicate that our

approach can also be used to improve the quality of model recommendation. That is, not only CBR may be indicated as an alternative when data is not available, but it may also be used to find highly suitable models. This supports our ultimate goal for our case-based platform to increase our understanding of modeling complex systems.

5 Case-Based Platform for Modeling Biological Systems

The diagnosis-prediction task can realize the Nutrigenomics and e-diagnosis dreams. Nutrigenomics is the field that interfaces nutritional environments with genetic and cellular processes [9]. E-diagnosis [10][26] is concerned with bringing quantitative biological information into the problem of medical diagnosis. The goal of the prediction task is to successfully determine which model accurately describes a human individual so that health outcomes, such as the diet-regulated influence of genes on chronic diseases [9][15], can be predicted based on this individual's genetic constitution and diet. Thus, diet and other forms of intervention can be designed to specifically meet each individual's genetic needs and to personalize recommendations to guarantee health outcomes. Imagine a simple exam at the time of birth to establish environmental and nutritional boundaries a child should stay in order to guarantee a long and healthy life.

Our genetic constitution interacts with the environment to either predispose or protect us from disease. The interplay of these two factors is most obvious if one compares cancer incidences in different countries [1]. Only specific genetic diseases show a clear and strong genetic background due to genetic mutations. Otherwise, the interplay between DNA and environment can be ranked according to the amount of genetic influence; e.g. the following conditions are sorted from most genetic to most environmental influence: psoriasis, depression, schizophrenia, diabetes, asthma, cardiac condition, cancer, and multiple sclerosis [7]. Environmental factors can be divided up into two components: 1) nutrition, treatments like drugs, air quality, and presence of toxins; and 2) lifestyle, like activities that impact metabolism, amount of sleep, stress, etc.

Recently, for the first time, a relationship between stress and the impact on the genetic constitution itself was reported [8]. However, it is important to notice that even normal aging has an effect on the genetic constitution and gene expression [21][23]. Changes are not consistent between individuals and may vary most at mid-lifespan [12], giving rise to a difference between chronological and biological age. Genetic changes, on the other hand, impact biological organization, e.g., immune system, respiratory system, mental abilities, bone structure. As a consequence, these changes, on whatever level of organization they occur, determine the interaction between the individual and the environment and shift with age.

The problem of modeling biological systems in order to support tasks such as diagnosis and prediction must consider genetic information, and it must be able to capture both how genes are influenced by the environment and how changes in gene expression impact an individual's health. In practice, given a partial description of a target individual, the goal is to fit a model (which could be created from a combination of models) that can accurately predict the individual's health from the environment to which the individual is exposed.

Table 3. Cases in case-based modeling platform

case	individual K			individual M			individual N			
problem	De-facto age	35			58			35		
	nutrition	chemicals x,y,z			chemicals p,q,r,s,t,u,v,x,y,z			chemicals x,y,z		
	genotype	TGGGGACACCTCGCCTGC			TGGGGACACCTCTCCTGCAC			TCAGGACACCTCGCCTGCAC		
	gene expression	AA80AB20			AA80AB80			AA80AB20		
	health	BP 120x180	BMI 40		BP 120x150	BMI 26		BP 120x180	BMI 40	
outcome solution	models	1	2	n	1	2	n	1	2	n
	accuracy	.6	.85	.15	.5	.65	.15	.3	.3	.5

Legend: BP=blood pressure; BMI=body mass index

For example, we would like to predict the health of individual K. The available information for individual K is a description of his interaction with his environment, with detailed proportions of nutrients, chemicals (e.g. drugs), and toxins; his genetic constitution through his DNA; his tissue, gene product expression profiles and blood clinical chemistry; and a description of his health through biomarkers and medical evaluation. The black area in Table 3 represents a case for individual K, who is 35 years old and obese.

Obtaining an accurate model for K's biological system will allow us to diagnose the causes for his obesity by determining the relationship between environmental and lifestyle parameters on one end and molecular constitution and physiological capabilities on the other end. It will be possible to prescribe a personalized strategy based on the predictive ability of such a model. The confirmation of the model's suitability will increase the overall understanding of biological systems.

A case-based platform for modeling biological systems has one crucial distinction from the software program system described in Section 2. Software programs can be easily modeled because it is possible to randomly generate inputs for training the inductive method as many times as necessary for a reasonably accurate model. With living biological systems, it is not typically feasible to submit the required amounts of inputs to observe changes in outputs. Sometimes it is possible to do it partially, or in varying scales, and targeting different systems (human, animal, or cellular). Modeling human systems is especially problematic because there are health risks, limited number of human subjects, uncertainty in intervention commitment, and it requires a long term for observance of outcomes. It is easier to conduct experiments with animals (e.g. mice), but tailoring results for humans is bounded by the different biological structure of the different species. It is possible to use human cells, but studies with a subset of cells lose the interaction with the rest of the body. Therefore, instead of one model to describe the biological system of an individual, we propose a case-based platform that will incorporate a series of models (Table 3), obtained from different sources (e.g. partial genetic data from the individual, other individuals,

animal cells), to represent potential ways of describing an individual's biological system. This adds complexity to the CBR cycle, as a series of solutions are needed when acquiring cases and the reuse step has to contemplate the suitability of potential models before a solution can be proposed.

In order to determine a model to describe individual K, we first have to assess the similarity between K and other cases in the case base. Those candidate cases from the case base are described with a series of models and their corresponding estimated qualities to describe each biological system (M and N in Table 3). Let us now suppose that individual M resulted with a high similarity score when compared to K, whereas individual N obtained low score. The reuse step would examine the nature of the models in order to assess their potential viability. Thus, not only the similarity between individuals would be used for the basis of reuse, but also the suitability of the models based on how the models were obtained. We use the similarity between inputs used to obtain the model and the individual's inputs as indicators of the expected accuracy of the model.

For individual M, let us assume Model₁ was obtained from a study with mice, like the one in [17]. This study used as inputs a portion of knockout mice (i.e. mice that had some genes *turned off*) and obtained as outputs different responses to polyunsaturated fatty acids. The small accuracy estimated for Model₁ stems from the fact that the source data was obtained from mice. Let us assume that Model₂ was built using human cells, such as studies described in [2]. This study has associated the lack of some specific chemicals, let us call them chemicals *p*, *q*, *r*, *s*, with an output of DNA damage. The higher accuracy for Model₂ originates from the fact that the model used human cells and that chemicals targeted by the study were also present in M's nutrition.

The reuse step would examine Model₁ and would balance the fact that the model was built for mice and K is human. Additionally, it would assess if there is domain knowledge to correlate low gene expression in K (AA20) with knockout genes in the mice population, which would cause to increase the accuracy of Model₁ for K to .6. For Model₂, K and M are genetically similar and therefore Model₂ would be potentially a good model for K. However, the chemicals used as inputs in Model₂ match the chemicals that are absent in K's nutrition, suggesting that if Model₂ is a good fit then the DNA damage might be present in K, increasing the accuracy for the model in K to .85. This is further corroborated by K's surface features, which include a BMI of 40 – severe obesity. K's solution is shown in the gray area in Table 3.

The high accuracy of Model₂ to describe K can be used to support the diagnosis that K has DNA damage due to the lack of chemicals *p*, *q*, *r*, *s* in his nutrition, and the DNA damage could be responsible for K's inability to process fatty acids, making him obese. The reuse step in this case allows us to better understand how to fit models to humans and a revision step (e.g. confirming a recommendation) after observing a patient along the years can potentially improve the understanding of such biological system.

This is how case-based reasoning can contribute and improve results compared to inductive or mathematical models alone. The contribution of CBR for this task is that it combines knowledge from different sources into one reasoning that enables a solution otherwise not feasible. This is where the ability to assess similarity between partially described systems and inductive models pays off: *we can use domain*

knowledge supporting similarity between different problems in order to assess the quality of a model to represent one of the problems (i.e. systems). To implement such a platform in practice the approach introduced in Section 3 is required, because it allows us to manipulate and assess similarity between systems and models that are not well understood.

6 Related Work

The CBR platform described in Section 5 reflects an ongoing trend to unite different computer science approaches to biomedicine [24]. Biological data abounds. Projects have been started to establish databases to organize such data. For example, the UK Biobank is a long-term project to start at the beginning of 2006 to gather information on the health and lifestyle of 500,000 volunteers [25]. CBR can become an essential methodology to analyze this data.

The problem of retrieving similar cases when the target problem is incomplete due to missing feature values was investigated in [6]. This work differs from ours in that their problems are sufficiently understood to design a similarity measure. Our problems are not well understood to design a similarity measure using conventional methods.

The most extensive analysis of neuro-CBR integrations [13] proposes a hierarchy for their description. When interpreting our use of ANN as an integration of the ANN technique into the CBR methodology, it could be categorized as chain-processing. CBR is the main processor and the ANN is responsible for a preprocessing (ANN models are trained for case acquisition) and a post-processing step (new ANNs are trained for reuse).

One aspect of our approach resembles the philosophy of the work discussed in [19], where authors propose an integration of ANN and memory-based learning. They argue that memory-based learning allows them to reuse the memory of the training instances used to train the ANN – what is never done with ANN, because inputs are discarded after the network is trained. The similarity is that we use ANN training data to help determine the suitability of potential models to represent a complex system (Section 5).

Notable CBR systems limit the biological information in their problem descriptions to the use of biomarkers. For example, blood pressure and blood clinical chemistry are used in ALEXIA [5]; and chemical compounds are represented in [3] to predict carcinogenic activity. These applications neither model individuals' biological systems nor reason at the genetic level.

7 Concluding Remarks and Future Work

We proposed an approach to assess similarity between complex systems and between inductive models. We have demonstrated that an inductive model can be recommended to represent a system on the basis of the system's similarity to other systems. This illustrates how CBR can contribute to the modeling task when systems to be modeled are not well understood. Our approach represents an important step

towards a learning platform that benefits from the combination of CBR and inductive modeling. Such a platform has the potential to enable unprecedented understanding of complex phenomena.

Biological data is usually partial and incomplete; different studies are pieces in a complex puzzle that humans are not capable of understanding. A case-based platform for biological systems would aggregate partial data into a lazy learning paradigm, where each new iteration would help increase the understanding of biological systems.

7.1 Future Work

This paper demonstrated the suitability of CBR for solutions that consisted of neural networks. We plan to test our approach using other inductive methods, i.e. info-fuzzy networks [11] and also with mathematical models.

We implemented the reuse step in our approach without considering the potential usefulness of the second closest case. A brief examination revealed this alternative may be useful to improve the accuracy of the 6 cases where the accuracy from the CBR recommendation in the LOOCV was lower than previously recorded data for that case. Our approach revealed that 13 features were consistently included in the DA functions, whereas the remaining 10 were consistently excluded. We plan to use these features in an attempt to help assess adaptation needs to reuse a solution of better quality, similar to adaptation-guided retrieval [20].

We also plan to test different variations of algorithms like the backward strategy removing one problem feature at a time and then confirming the clustering until one set of features for the entire dataset supports the clustering. This will probably require the elimination of some outliers, and a bigger dataset.

Finally, we plan to explore rule sets generated by each run of the ANN in order to learn more about what features make some rule sets more successful than others. This could help us predict the quality of the model without having to apply it.

Acknowledgements

The authors would like to thank Dr. M. Last, Dr. A. Kandel, and T. Barr for their continuous support in different stages of our work. Thanks R. J. Upadhyay for his help in developing testbeds. Dr. R. Weber and J. M. Proctor are supported in part by the National Institute for Systems Test and Productivity at USF under the USA Space and Naval Warfare Systems Command grant no. N00039-02-C-3244, for 2130 032 L0, 2002.

References

- [1] Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., Walter, P.: *Molecular Biology of the Cell*. 4th edn. Garland Publishing, New York (2002)
- [2] Ames, B.N.: DNA Damage from Micronutrient Deficiencies is Likely to Be a Major Cause of Cancer. *Mutat Res.* 475 1-2 (2001) 7-20

- [3] Armengol, E., Plaza, E.: Relational Case-based Reasoning for Carcinogenic Activity Prediction. *Artificial Intelligence Review*, 20, 1 - 2 (2003) 121 - 141
- [4] Barr, T.: Architectural Overview of the Computational Intelligence Testing Tool. In: *Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering*. IEEE Computer Society, Los Alamitos (2004) 269- 270
- [5] Bichindaritz, I.: Memoire: Case Based Reasoning Meets the Semantic Web in Biology and Medicine. In: Gonzalez Calero, P.A., Funk, P. (eds.): *Case-Based Reasoning Research and Development*. LNAI, Vol. 3155. Springer, Berlin Heidelberg New York (2004) 47-61
- [6] Bogaerts, S., Leake, D. B.: Facilitating CBR for Incompletely-Described Cases: Distance Metrics for Partial Problem Descriptions. In: Gonzalez Calero, P.A., Funk, P. (eds.): *Case-Based Reasoning Research and Development*. LNAI, Vol. 3155. Springer, Berlin (2004) 62-76
- [7] Chakravati, A., Little, P.: Nature, nurture and human disease. *Nature*. 421 (2003) 412-414
- [8] Epel, E.S., Blackburn, E.H., Lin, J., Dhabhar, F.S., Adler, N.E., Morrow, J.D., Cawthon R.M.: Accelerated Telomere Shortening in Response to Life Stress. *Proc. Natl. Acad. Sci.* 101 49 (2004) 17312-5
- [9] Kaput, J., Rodriguez, R.L.: Nutritional Genomics: the Next Frontier in the Postgenomic Era. *Physiol. Genomics* 16 (2004) 166-177
- [10] Kriete, A., Boyce, K.: Automated tissue analysis – a bioinformatics perspective. *Methods Inf. Medicine* 1 (2005) 32-37
- [11] Last, M., Friedman, M., Kandel, A.: The Data Mining Approach to Automated Software Testing. In: *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM Press, New York (2003) 388-396
- [12] Lu, T., Pan, Y., Kao, S.Y., Li, C., Kohane, I., Chan, J., Yankner, B.A.: Gene Regulation and DNA Damage in the Ageing Human Brain. *Nature* 429, 6994 (2004) 883-91
- [13] Malek, M.: Hybrid Approaches for Integrating Neural Networks and Case-Based Reasoning: From Loosely Coupled to Tightly Coupled Models. In: Pal, S.K., Dillon. T.S., Yeung, D.S. (eds.): *Soft Computing in Case Based Reasoning*. Springer Verlag, London (2001) 73-94
- [14] McFarlane, A.C., Yehuda, R., Clark, C.R.: Biologic Models of Traumatic Memories and Post-Traumatic Stress Disorder. The role of neural networks. *Psychiatr Clin North Am.* 25, 2 (2002) 253-70
- [15] Park, E.I., Paisley, E.A., Mangian, H.J., Swartz, D.A., Wu, M., O'Morchoe, P.J., Behr, S.R., Visek, W.J., Kaput, J.: Lipid Level and Type Alter Stearoyl CoA Desaturase mRNA Abundance Differently in Mice with Distinct Susceptibilities to Diet-Influenced Diseases. *J Nutr.* 127, 4 (1997) 566-73
- [16] Proctor, J. M., Weber, R.: Systematically Evolving Configuration Parameters for Computational Intelligence Methods. Submitted to the First International Conference on Pattern Recognition and Machine Intelligence (PReMI'05) (2005)
- [17] Ren, B., Thelen, A.P., Peters, J. M., Gonzalez, F.J., Jump, D.B.: Polyunsaturated Fatty Acid Suppression of Hepatic Fatty Acid Synthase and S14 Gene Expression Does not Require Peroxisome Proliferator-Activated Receptor- α . *J. Biol. Chem.* 272 (1997) 26827–26832
- [18] Saraph, P., Last, M., Kandel, A.: Test Set Generation and Reduction with Artificial Neural Networks. In: Last, M., Kandel, A., Bunke, H. (eds.): *Artificial Intelligence Methods in Software Testing*. World Scientific (2004) 101-132

- [19] Shin, C. K., Park, S. C.: Towards Integration of Memory Based Learning and Neural Networks. In: Pal, S.K., Dillon. T.S., Yeung, D.S. (eds.): *Soft Computing in Case Based Reasoning*. Springer Verlag, London (2001) 95-114
- [20] Smyth, B., Keane, M.T.: Experiments on Adaptation-Guided Retrieval in Case-Based Design. In: Veloso, M., Aamodt, A. (eds.): *Proceedings of the 1st International Conference on Case-Based Reasoning*. LNAI, Vol. 1010, Springer, Berlin (1995) 313-324
- [21] Thomas, R.P., Guigneaux, M., Wood, T., Evers, B.M.: Age-Associated Changes in Gene Expression Patterns in the Liver. *J Gastrointest Surg.* 6 3 (2002) 445-53
- [22] Weber, R., Wu, D.: Knowledge Management for Computational Intelligence Systems. In: *Proceedings of the Eighth IEEE International Symposium on High Assurance Systems Engineering*. IEEE Computer Society, Los Alamitos (2004) 116-125
- [23] Welle, S., Brooks, A.I., Delehanty, J.M., Needler, N., Thornton, C.A.: Gene Expression Profile of Aging in Human Muscle. *Physiol. Genomics* 14, 2 (2003) 149-59
- [24] Wiemer, J., Schubert, F., Granzow, M., Ragg, T., Fieres, J., Mattes, J., Eils, R.: Informatics United: Exemplary Studies Combining Medical Informatics, Neuroinformatics and Bioinformatics. *Methods Inf. Med.* 42, 2 (2003) 126-33
- [25] Wright, A., Carothers, A.D., Campbell, H.: Gene-environment interactions – the Biobank UK study. *Pharmacogenomics J.* 2 (2002) 75-82
- [26] Zhao, L.P., Gilbert, S., Defty, C.: E-Diagnosis Using GeneChip Technologies *Proceedings of the Fourth International Conference on Advances in Infrastructure for e-Business, e-Education, e-Science, e-Medicine on the Internet*. CD-ROM. (2002)