Learning Adaptations for Case-Based Classification: A Neural Network Approach

Xiaomeng Ye, David Leake, Vahid Jalali and David Crandall

Luddy School of Informatics, Computing, and Engineering, Indiana University
Bloomington IN 47408, USA
{xiaya,leake}@indiana.edu, vjalalib@alumni.iu.edu, djcran@iu.edu

Abstract. Case-based Reasoning (CBR) solves new problems by retrieving a stored case for a similar problem and adapting its solution to fit. Acquiring case adaptation knowledge is a classic problem for CBR. A popular method for addressing it is the case difference heuristic (CDH) approach, which learns adaptations from pairs of cases based on their problem differences and solution differences. The CDH approach has often been used to generate adaptation rules, but recent CBR research on case-based regression has investigated replacing learning rules with learning CDH-based network models for adaptation. This paper presents and evaluates a neural network based CDH approach for learning adaptation models for classification, C-NN-CDH. It examines three variants, (1) training a single neural network on problem-solution differences, (2) segmenting adaptation knowledge by the classes of source cases, with a separate neural network to generate adaptations for each group, and (3) adapting from an ensemble of source cases and taking the majority vote. Experimental results demonstrate improved performance compared to previous research on statistical methods for computing CDH differences for classification. Additional results support that C-NN-CDH achieves classification performance comparable to that of multiple classic classification approaches.

Key words: Case Difference Heuristic, Classification, Ensemble Learning, Neural Network Based Adaptation

1 Introduction

Case-based Reasoning (CBR) solves new problems by retrieving a stored case with a similar problem and adapting the solution to accommodate the new problem (e.g., [18]). Case-based reasoning is appealing for properties such as enabling a natural knowledge capture process for cases in suitable domains (e.g., [19]), facilitating knowledge acquisition, and interpretability of cases to justify solutions [5].

However, obtaining the adaptation knowledge needed to adapt prior solutions is a classic challenge. In response, extensive research has explored the use of machine learning methods to acquire case adaptation knowledge for both classification (e.g., [7, 11]) and regression (e.g., [8, 10, 17, 22, 23]). An interesting recent direction is the use of neural network methods for CDH learning for regression adaptations. This paper presents and evaluates a neural network method for learning the adaptation knowledge needed for classification tasks.

The case difference heuristic (CDH) approach [9] is one of the most used methods to learn adaptation knowledge. The CDH approach takes pairs of cases from the case base and from each pair learns a rule to adapt one case to another. As a simplified example, consider applying CBR to apartment price prediction. Suppose two apartments A and B are very similar, except that A has carpeted floor while B has a wooden floor, and that the rent for B is $200 more. By comparing apartments A and B, a CDH approach might learn the rule that changing from carpet to wooden floor increases an apartment's rent by $200. An issue for CDH approaches is what generalization to learn from the case pairs: Rather than learning an absolute difference, a CDH approach might learn the percent change or another characterization of the observed difference.

Recent work by Liao, Liu, and Chao [17] learns adaptations for regression tasks with a case difference heuristic approach using a neural network to learn the difference characterization. Their approach trains a network to map problem differences to differences in output values, avoiding the need to pre-define generalization strategies. Jalali, Leake, and Forouzandehmehr [11] apply the CDH approach to classification, using a statistical method to generate case adaptation rules for classification.

To our knowledge, this study presents the first neural network based case difference heuristic approach to classification. Our approach, which we refer to as case-based **C**lassification with **N**eural **N**etwork based **CDH** (C-NN-CDH), uses neural networks to learn adaptation knowledge from pairs of cases. Experimental results on multiple data sets show that the C-NN-CDH approach outperforms the statistical approach of [11], the previous state of the art for statistical adaptation for classification.

This study investigates five variants of the C-NN-CDH approach. Some variants segment the pairs of cases based on their classes and train a separate model per segment. The segmented variants offer faster training but slightly lower accuracy. We also tested variants using an ensemble of adaptations; These provide roughly comparable performance to their non-ensemble counterpart. A variant taking a majority vote of one adaptation from each class provides additional accuracy for certain data sets. Comparisons with a sampling of standard classification approaches supports that the accuracy of the C-NN-CDH approaches is competitive with those methods. In particular, this hybrid method provides accuracy comparable to that of a network-only method dedicated to the classification task, while its use of CBR provides at least two benefits: Inertia-free lazy learning (avoiding the need for costly retraining with new data), and the ability to provide cases that can be considered when assessing a classification; similar cases can be useful for explanation [5], and less similar cases—for example, whose classifications are changed by adaptation—may be useful as nearest unlike neighbors [21] or counterfactual explanations [12].

2 Background

2.1 Learning Case Adaptation Knowledge

Adaptation is arguably the most difficult process in CBR. Much CBR research has applied machine learning to acquire adaptation knowledge of different forms. Some approaches apply case-based reasoning to adaptation. For example, Leake et al. [15] present a method in which a case base of pairs of cases is populated from past successful adaptations, and Craw et al. [4] assemble pairs of stored cases, retrieve the pair most similar to the pair of a retrieved case and the query, and adapt the retrieved case toward the current query.

The case difference heuristic (CDH) approach [9] is a widely used knowledge-light method for learning case adaptation rules from knowledge contained in the case base. For each pair of cases, the CDH approach generates an adaptation rule capturing the transformation needed to adapt the solution of one case into the solution of the other. Specifically, it attributes the difference in the cases' solution descriptions to the difference in their problem descriptions. When deciding applicability of the generated rules in the future, the rule is considered to apply if the difference between the retrieved case's problem and the new problem is similar to the difference from which the rule is generated. Thus the similarity to the original difference becomes the antecedent for the rule. When triggered, the rule adapts the retrieved case's solution according to the previous solution difference.

Applying the case difference heuristic depends on several design questions. One concerns how to calculate problem differences; Another concerns how to select the case pairs for training (e.g., from pairs of neighboring cases or from random pairs); Another concerns how to translate a raw solution difference into the change to be effected by the rule, for example, as an additive, multiplicative or other change. There have been many variations of CDH (e.g. [4, 10, 17, 23]). This paper proposes a general approach in which difference and changes are learned by a neural network. It remains agnostic on such design decisions.

Augmenting CDH with Network Methods Adaptation rule generation using the CDH approach is often shaped by pre-defined criteria for generating rules from differences. In contrast, machine-learning-based approaches [3, 4, 17, 23, 29] provide increased flexibility. Liao, Liu, and Chao [17], Policastro, Carvalho, and Delbem [23], and Zhang et al. [29] propose a network approach in which the network generates adaptations from a problem and retrieved solution passed to the network. Part of the appeal of network-based approaches is that network learning facilitates the generation of more complex transformation functions. Outside of CBR but in similar spirit, Wetzel et al. [24] use a siamese network to predict the target value differences given two data points, and predict a target value using an ensemble of training data points. Leake, Ye, and Crandall [16] follow the CDH approach of Craw, Wiratunga, and Rowe [4], by considering both problem difference and adaptation context. Following Liao, Liu, and Chao [17], Leake, Ye and Crandall propose a neural network based case difference heuristic approach, NN-CDH, as a general technique for regression tasks.

Applying the CDH Method to Classification The traditional CDH approach has been successfully applied in solving regression tasks. However, less research has addressed classification tasks or dealing with nominal attributes in generating adaptation rules. Early approaches that adopt CDH for classification or dealing with nominal attributes relied on exact matching and binarization (e.g., [2, 7]) which can only express the relationship between nominal values using one bit of information (i.e. 0 and 1). More recently, CDH was enhanced with the Value Difference Metric (VDM) [11]. VDM is a probabilistic method to measure similarity that makes it possible to compare nominal values in a one dimensional numeric space. Another contribution of that work was to use an ensemble of adaptations for classification (EAC) to retrieve multiple source cases, generate needed adaptation rules, adapt from all retrieved source cases, and produce a final solution by the majority vote of all adapted solutions.

To illustrate the importance of expressive power in comparing nominal values one can consider an example classification task where the goal is to decide whether a given fruit is apple or not based on the color of the fruit. In this case, binarizing color results in capturing the difference between the colors red and yellow as the same as that between the colors red and blue. However, using a more expressive method like VDM enables recognizing the relative proximity of colors red and yellow compared to that of colors red and blue. Recent advances in deep neural networks have made it possible to take the expressive power in comparing nominal values a step further by expressing them in multi-dimensional space as embedding vectors. To the best of our knowledge this has not been exploited previously for CDH learning.

2.2 Class-to-class Methodology

Class-to-class (C2C) methodology is a difference-based approach that classifies a query based on instances from multiple classes [25–27]. A C2C model first learns the similarity and difference patterns between pairs of classes. Given two cases, the trained C2C model can determine whether their similarity and difference conform to learned patterns. If they do, the C2C model can provide evidence for their belonging to the corresponding classes.

Traditional case-based methods explain their conclusion by the most similar case retrieved and its adaptation, while C2C methods can explain with both supportive and contrastive evidence. The contrastive evidence from the C2C methodology is one type of counterfactual explanation (cf. Keane and Smith [12] and Kenny and Keane [13]). For example, for an applicant rejected of a loan, the supportive explanation is that another applicant with similar status is also rejected, and the contrastive explanation is that if this applicant had better credit history, (s)he would have been similar to an accepted applicant.

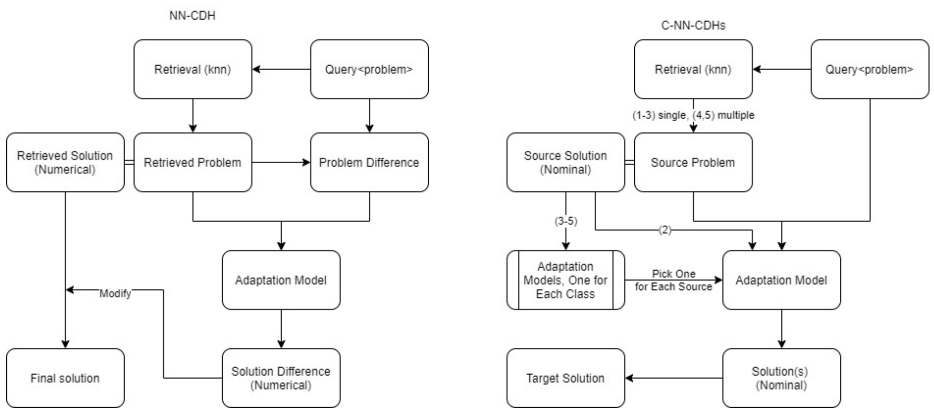

Fig. 1. Workflows of NN-CDH (left) and C-NN-CDHs (right). Variants (1-5) of C-NN-CDHs involve different procedures and arrows are marked with the corresponding numbers to reflect that.

3 An NN-CDH Approach for Classification

NN-CDH and other CDH methods learn from pairs of cases. One of the pairs is treated as the source case (with its source problem and solution) and the other as the target (with its target problem and solution), where the source is to be adapted toward the target. For simplicity, we will refer to these as a case pair. A CDH method learns an adaptation rule to adapt the solution of the source case to provide a solution for the target case. For an NN-CDH approach, the CBR system first retrieves a source case similar to the query (target case), and calculates the problem difference between the source problem and the target problem. The problem difference is then passed to a neural network, which is previously trained on problem and solution differences of training case pairs. The neural network predicts the solution difference between the source solution and the target solution. Finally, the CBR system applies the predicted solution difference to the source solution, and uses the adapted result as the final prediction. This process is shown in Figure 1.

Calculating the difference between problem or solution values requires a difference function, which is especially difficult to define for nominal values.

Our method replaces the traditional CDH difference calculation by the implicit calculation of a machine learning technique (e.g. neural network), potentially taking into account not only the difference, but the context of the source case itself. We name this general approach of handling pairs of cases as the case difference heuristic approach for classification ("C-CDH").

As a baseline testbed system, we implemented a C-CDH system that stores case pairs treated as adaptation rules. The case pairs are grouped based on source solution. We refer to this as variant (0) and describe it in Algorithm 1. The system performs classification by retrieving the most similar source case, and retrieving the case pair, which is selected to share the same source solution

Algorithm 1 C-CDH, variant (0)

```
 1: for each $i$ in all classes do
 2:     case_pairs[$i$] ← {}
 3: for each $source$ and $target$ in $pairs$ do
 4:     case_pairs[$sol(source)$].append([$prob(source), prob(target)$:$sol(target)$])
 5: procedure Testing($query$)
 6:     $retrieved$ ← 1-nn($CB, query$)
 7:     $r$ ←1-nn(case_pairs[$sol(retrieved)$], [$prob(retrieved), prob(query)$])
 8:     return $r$
```

Algorithm 2 C-NN-CDH, variant (1-2)

```
 1: $adapt\_NN$ ← new classification neural network
 2: case_pairs ← {}
 3: for each $source$ and $target$ in $pairs$ do
 4:     if using variant (1) then
 5:         case_pairs.append([$prob(source), prob(target)$:$sol(target)$])
 6:     else if using variant (2) then
 7:         case_pairs.append([$prob(source), prob(target), sol(source)$:$sol(target)$])
 8: $adapt\_NN$.fit(case_pairs)
 9: procedure Testing($query$)
10:     $retrieved$ ← 1-nn($CB, query$)
11:     $r$ ← $adapt\_NN$.predict($prob(retrieved), prob(query), sol(retrieved)$)
12:     return $r$
```

and has the most similar source problem and target problem (cf. [20]). The target solution of the retrieved case pair is used as the final classification.

Our implementation of C-CDH uses a classification neural network to learn and predict the target solution based on information from the source problem and the target problem.

This is the basic version of classification with a neural network based case difference heuristic (C-NN-CDH) approach and will be referred to as variant (1). As a direct extension, we built variant (2) in which the adaptation neural network also takes in the source solution as input. Variants (1) and (2) are described in Algorithm 2.

Variants (3-5) add grouping of case pairs based on their source solutions. The target solutions in a group are not restricted. With grouping, an adaptation neural network can be trained on a specific group of case pairs to learn the adaptation knowledge where the source solution is determined. In other words, each **specialized adaptation neural network** learns how to adapt cases of a specific solution toward all solutions (including the source solution). By segmenting the pairs of cases based on source solutions, we naturally incorporate the source solution as an important input, as it determines which specialized adaptation neural network to use. The training is also easier as one group of case pairs is more homogeneous and the knowledge to learn is more specific. Variants

Algorithm 3 C-NN-CDH, variant (3-5)

```
1: for each $case$ in $CB$ do
2:     $CB\_by\_class[sol(case)]$.append($case$)
3: for each $i$ in all classes do
4:     case_pairs[$i$] $\leftarrow$ {}
5: for each $source$ and $target$ in $pairs$ do
6:     case_pairs[$sol(source)$].append([$prob(source), prob(target)$:$sol(target)$])
7: for each $i$ in all classes do
8:     $adapt\_NN[i]$ $\leftarrow$ new classification neural network
9:     $adapt\_NN[i]$.fit(case_pairs)
10: procedure Testing($query$)
11:     if using variant (3) then
12:         $retrieved \leftarrow$ 1-nn($CB, query$)
13:         $r \leftarrow adapt\_NN[sol(retrieved)]$.predict($prob(retrieved), prob(query)$)
14:     else if using variant (4) then
15:         for each $retrieved$ in k-nn($CB, query$) do
16:             $rs$.append( $adapt\_NN[sol(retrieved)]$.predict($prob(retrieved), prob(query)$))
17:         $r \leftarrow$ majority_vote($rs$)
18:     else if using variant (5) then
19:         for each $i$ in all classes do
20:             $retrieved \leftarrow$ 1-nn($CB\_by\_class[i], query$)
21:             $rs$.append( $adapt\_NN[i]$.predict($prob(retrieved), prob(query)$))
22:         $r \leftarrow$ majority_vote($rs$)
23:     return $r$
```

(3-5) share the same training procedure for their specialized adaptation neural networks but differ in their testing procedures.

Variant (3) predicts the target solution by retrieving the most similar source case and using one specialized adaptation neural network. Inspired by the ensemble of adaptations for classification approach (EAC) [11], variant (4)—named EAC-NN-CDH—retrieves k multiple similar source cases (we used $k = 3$), adapts all source cases using corresponding specialized adaptation neural networks, and selects a classification by majority vote. Ties are broken arbitrarily.

Variant (5) is inspired by class-to-class (C2C) methodology and named C2C-NN-CDH. In this study, C2C-NN-CDH retrieves one most similar source case from each class, adapts all source cases using their corresponding networks, and uses the majority vote to decide the final classification. The voting process is similar to the all-versus-all approach in multiclass classification [1]. Variants (3-5) are described in Algorithm 3.

Variants (1-5) are illustrated in Figure 1. All variants are summarized below:

(0) Non-network C-CDH.
(1) C-NN-CDH with one adaptation neural network that considers source problem and target problem.
(2) is based on (1), but also considers source solution.
(3) uses multiple specialized adaptation neural networks.

(4) is based on (3), but uses an ensemble of adapted solutions from multiple cases.
(5) is based on (4), but uses an ensemble of adapted solutions from multiple cases of all classes.

4 Evaluation

We evaluated all variants (0-5) on two groups of data sets. The first group of data sets follows those used in Jalali, Leake, and Forouzandehmehr [11], to enable comparison with the previous state of the art on statistical CDH classification. Experiments on this group allow comparison with the ensemble approach EAC and EAC-retrieval, an ablated EAC removing the adaptation component. The second group of data sets is a subset of data sets in the comparative evaluation of classification algorithms by Zhang et al. [28]. Experiments on this group allow comparison with algorithms evaluated in that paper, including: Extreme Learning Machine (ELM), Sparse Representation based Classification (SRC), Deep Learning (DL), Support Vector Machine (SVM), Random Forests (RF), AdaBoost (AB), C4.5, Naive Bayes classifier (NB), K Nearest Neighbours classifier (KNN) and Logistic Regression (LR). We compare the results of runs of our systems with the reported results from Jalali, Leake, and Forouzandehmehr [11] and Zhang et al. [28]. We note that the data preprocessing steps are not described in detail in the two papers. This may result in minor variations.

All data sets are for classification tasks, taken from UCI repository [6]. All nominal values are converted to one-hot encoding and all numeric values are standardized by removing the mean and scaling to unit variance. For most data sets, five 10-fold cross validations are carried out, where 10% of the total cases are used for testing and 90% are used for training (only two 10-fold cross validations are run for two larger data sets with excessive training time). The average accuracies and balanced accuracies (with their standard deviations) of all runs for each data set are recorded. Standard deviations are omitted in reports below as almost all are less than 0.05. Balanced accuracy in general is comparable to accuracy but is not shown for reasons of space.

4.1 Assembling Case Pairs

As discussed in Section 2.1, the collection of case pairs is a design problem for CDH. Given a training data set of n cases and m classes, our test implementations learn adaptation knowledge from three kinds of pairs from the case base:

– Neighboring pairs: Each case is paired with its nearest neighbor using 1-NN (k-nearest neighbor with $k = 1$). There are n neighboring pairs.
– Random Pairs: Each case is paired with 10 random cases. There are $10n$ random pairs.
– Class-to-class Pairs (C2C Pairs): Each case is paired with its nearest neighbor in every other class. There are $n(m - 1)$ pairs.

Each type of pairs provides one specialized form of adaptation knowledge to the adaptation model: The neighboring pairs provide minor adaptations to cover small problem differences; The random pairs provide random and bigger adaptations; The C2C pairs provide adaptations needed to change one case into other classes. The number of each kind of pairs is a design parameter that could be fine tuned. Pairs might also be selected according to other criteria such as generality or applicability, but this is beyond the scope of this study.

4.2 Implementation Details

For the C-CDHs, the retrieval component retrieves a single case. As baselines for performance without adaptation, we also implemented nearest neighbor algorithms 1-NN and 3-NN, with 3-NN averaging the classifications of the three most similar cases.

The adaptation neural network is a feedforward network with 2 hidden layers (128 and 64 nodes with ReLU activation functions) and an output layer with softmax activation function. The loss function is categorical cross entropy and the model is optimized using Adam [14]. For comparison, a neural network classifier is implemented with the same configuration. Note that the adaptation neural network is a component in the CBR system (C-NN-CDH) and it produces the final classification based on a retrieved case and the query, while the neural network classifier directly produces the final classification based solely on the query. All networks are trained until their parameters converge.

For 3-NN and 1-NN, all training cases are used as the case base. For the neural network classifier, 10% of the training cases are separated out for training validation. For all the C-CDHs, pairs are assembled using methods described in Section 4.1. For non-network variant (0), all case pairs are stored for future search. For the adaptation neural networks in variants (1-5), 95% of the case pairs are used for training and 5% for validation.

4.3 C-NN-CDH vs. EAC

Table 1 compares the accuracy of C-NN-CDH with EAC. The accuracy of the best performing system and the best performing C-CDH for every data set is highlighted. We observe:

- The non-network C-CDH often makes the retrieval result of 1-NN worse. Because all variants use the same retrieval, this can be ascribed to variant (0) actually impairing performance. We hypothesize that this is due to using untuned retrieval for case pairs.
- The C-NN-CDHs have slightly different performance, the best of which is on par with that of the neural network classifier.
- The C-NN-CDHs consistently improve the retrieval result of 1-NN. The C-NN-CDHs also outperform EAC in many experiments. Note that the EAC-retrieval, by using a probability-guided metric, is often better than 1-NN. This means that C-NN-CDHs build on worse retrieval than EAC but end

	EAC		Baseline Systems			C-CDHs					
	EAC-retrieval	EAC	NNet	3-NN	1-NN	(0)	(1)	(2)	(3)	(4)	(5)
Credit	0.8189	0.8436	**0.8633**	0.8333	0.7978	0.7004	0.8205	0.8153	0.8229	0.8253	***0.8453***
Balance	0.7474	0.8402	0.9737	0.7964	0.7769	0.6561	**0.9792**	0.9785	0.9763	0.9737	0.976
Car	0.935	0.9605	0.9974	0.8313	0.781	0.6822	**0.9993**	0.9991	0.989	0.9936	0.9946

Table 1. Accuracies of Systems Compared with EAC [11]

	Baseline Systems			C-CDHs					
	NNet	3-NN	1-NN	(0)	(1)	(2)	(3)	(4)	(5)
Neural Network	Yes				Yes	Yes	Yes	Yes	Yes
Segmented Training							Yes	Yes	Yes
Ensemble		Yes						Yes	Yes
Class-to-class									Yes
Yeast	**0.5983**	0.5010	0.5317	0.4230	0.5218	0.5176	0.4648	0.4923	***0.5833***
Seeds	0.9400	0.9152	0.9285	0.8523	**0.9533**	**0.9533**	0.9361	0.9514	0.9485
Pima	**0.7536**	0.7320	0.7057	0.6408	0.6940	0.6955	0.6802	0.6908	***0.7216***
Page-blocks	**0.9685**	0.9665	0.9646	0.9473	0.9595	0.9566	0.9489	0.9515	***0.9675***
Contraceptive	**0.5439**	0.4103	0.4346	0.3891	0.4744	0.4778	0.4870	0.4859	***0.5139***
White Wine	0.5763	0.5425	**0.6569**	0.4693	0.6353	0.6344	0.5504	0.6035	***0.6354***
Balance	0.9737	0.7964	0.7769	0.6561	**0.9792**	0.9785	0.9763	0.9737	0.9760
Car	0.9974	0.8313	0.7810	0.6822	**0.9993**	0.9991	0.9890	0.9936	0.9946

Table 2. Variant Characteristics and Variant Accuracies Compared with Classifiers in [28]

with better results, demonstrating the value of their learned adaptation capability.

4.4 C-NN-CDH vs. Other Classification Algorithms

We compare variants of C-CDH with 11 state-of-the-art classification algorithms (referred as other algorithms) that are not necessarily related to CBR [28]. Data sets are chosen to be compatible with the our baseline and proposed systems. In other words, they require no additional preprocessing and are not complicated data such as images or structured sequences.

In Table 2, for each data set, the accuracies of the baseline systems and C-CDHs are listed. In Table 3, the best and the worst of the other algorithms and their corresponding accuracies are listed for comparison. Last, the best performing C-NN-CDH is chosen and its projected rank is shown in Table 3—i.e., the rank if it were ranked among the other algorithms. We observe:

– The average rank of our best C-NN-CDHs is 3.4. In Zhang et al. [28], SVM has an average rank of 3.5 and is the 3rd best among the 11 classifiers in

	Best		Best C-CDH			Worst	
	Name	Accuracy	Name	Accuracy	Rank	Name	Accuracy
Yeast	ELM	0.6487	(5)	0.5833	7	DL	0.3311
Seeds	KNN	0.9524	(1,2)	0.9533	1	DL	0.2381
Pima	AB	0.8312	(5)	0.7216	6	DL/SRC	0.5974
Page-blocks	SVM	0.9444	(5)	0.9675	1	ELM/DL	0.8704
Contraceptive	GBDT	0.5541	(5)	0.5139	7	AB/DL	0.4122
White Wine	AB/DL	0.5694	(5)	0.6354	1	NB	0.3959
Balance	SRC	1.0000	(1)	0.9792	2	DL	0.4603
Car	GBDT	1.0000	(1)	0.9993	2	AB/DL	0.6705

Table 3. Rank of Best C-CDHs among Classifiers in [28]

terms of average rank (However we do not test C-NN-CDHs on all the data sets as in Zhang et al. [28]).

– C-NN-CDHs do not always improve the final result compared to the simple retrieval of 1-NN. For example, when testing on white wine quality, all C-NN-CDHs perform worse than 1-NN. We hypothesize that this is due to the high number of classes in this data set and the subjective nature of wine quality. When the relation between problem and solution is highly volatile, nearest neighbor is already a good guess while any adaptation might alter the prediction for worse.

4.5 C-NN-CDH vs. Baseline Neural Network

Because standard deviation is not reported in the works being compared to C-NN-CDH [11, 28], we are not able to calculate a P-value stating the significance of the difference between C-NN-CDHs and their methods. However, we are able to do so for the difference between the best performing C-NN-CDH and the baseline neural network in Table 4. The P-value is the probability of obtaining the observed difference between the samples if the null hypothesis were true. The null hypothesis states that the two distributions of results are the same. The calculation is based on the assumption that the distributions are normal. As Table 4 shows, the neural network wins in 3 data sets, C-NN-CDH wins in 2, and there are no significant differences between the two in the remaining half of the data sets.

It is expected that Table 4 does not show a clear advantage of C-NN-CDH over neural network in terms of accuracy, because the two use the same architecture and are naturally of similar power. However, C-NN-CDH is a component generally applicable to CBR classification systems, which can offer benefits such as lazy learning and explainability, in contrast to a neural network.

4.6 Evaluation Summary

From experiments on both groups of data sets, we answer the following questions:

	NNet Better <———————>C-NN-CDH Better								
	Contra.	Pima	Credit	Yeast	Page-b.	Seeds	Balance	Car	White W.
NNet	0.5439	0.7536	0.8633	0.5983	0.9685	0.9400	0.9737	0.9974	0.5763
C-NN-CDH	0.5139	0.7216	0.8453	0.5833	0.9675	0.9533	0.9792	0.9993	0.6354
P-value	0.0004	0.0006	0.0525	0.0687	0.6761	0.1407	0.1373	0.0174	<0.0001

Table 4. The Significance of the Difference between Best Performing C-NN-CDH and Baseline Neural Network

- **Can the neural network effectively learn adaptation knowledge?**
  Yes. One or more C-NN-CDHs can always provide performance comparable or even surpass that of the neural network classifier. In principle the adaptation neural networks might learn to discard the source problem and solely use the target problem to predict the target solution, effectively performing as a neural network classifier. However, our experiments reveal that this is not the case, because (1) the weights associated with the source problem are non-zero, and (2) as shown in Table 4, C-NN-CDHs perform significantly differently from the neural network in multiple experiments. C-NN-CDHs are indeed learning adaptation knowledge.
  This demonstrates that if a neural network is powerful enough to tackle the classification task directly, it may also be powerful enough to learn the adaptation knowledge or the relation between pairs of cases in the task domain.
- **Is the source solution an important attribute to consider in adaptation?**
  Not necessarily. A surprising result is that variant (1) actually performs almost identically to variant (2), which also considers the source solution in adaptation. We speculate that this is because the source solution is heavily coupled with the source problem, and therefore does not provide additional information useful in adaptation.
- **Does segmenting pairs of cases by source case solution lead to better performance?**
  This depends. In terms of accuracy, variants (1,2) actually perform better than variants (3,4) in most data sets. We speculate that this is because a single adaptation neural network in (1,2) is well trained with all pairs of cases, while a specialized adaptation neural network in (3,4) is trained with a segmented group of examples. In terms of efficiency, the training time needed for variants (1,2) are several times higher than that for variants (3,4). This is expected as variants (3,4) train on segmented training examples, and therefore converge faster.
- **Does an ensemble of adaptations improve accuracy?**
  Not really, for these data sets. EAC-NN-CDH (variant (4)) performs about the same as its counterpart variant (3) without ensemble. [11] showed that EAC adaptation is a better adaptation method than applying a single adaptation rule, while we do not observe a significant benefit of EAC-NN-CDH over C-NN-CDH. We attribute this to the generalization power of C-NN-

CDH, which produces predictions stable enough that an ensemble version does not appreciably alter its prediction.

- **Is a class-to-class methodology useful for adaptation?**
  Yes. C2C-NN-CDH (variant (5)) performs differently from and, in many scenarios, better than the other C-NN-CDHs. C2C-NN-CDH reaches its prediction by collecting evidence from diverse source cases from all classes, which can form a global support especially when there are multiple classes. Moreover, C2C methodology offers the possibility of explanation with contrastive evidences.

5 Conclusion

The flexibility of a case-based reasoning system to solve novel problems depends on its ability to adapt prior solutions to new circumstances. The generation of knowledge for adapting cases is a classic challenge for case-based reasoning. The case difference heuristic approach is a knowledge-light method for learning adaptation knowledge. Neural network based CDH has been successfully applied to case-based regression but not previously to classification.

This paper presents a method with multiple variants for extending network CDH for classification tasks with three contributions beyond the prior methods. First, variants (3-5) group the pairs of cases used for learning by the solutions of the source problems they adapt, generating per-category adaptation knowledge. Second, they apply one or multiple neural networks to learn the adaptation knowledge for classification. Third, variant (5) utilizes cross-class adaptation to reach a conclusion from cases of diverse classes.

In summary, the C-NN-CDH approach achieves better performance than EAC, the state of the art in statistical CDH adaptation, and is on par with standard classification methods from outside of case-based reasoning. It is generally applicable to CBR framework and thus preserve other benefits of CBR including lazy learning and explainability.

References

1. Aly, M.: Survey on multiclass classification methods. Tech. rep., Caltech (2005)
2. Badra, F., Cordier, A., Lieber, J.: Opportunistic adaptation knowledge discovery. In: Case-Based Reasoning Research and Development, ICCBR 2009. pp. 60–74. Springer, Berlin (2009)
3. Corchado, J., Lees, B.: Adaptation of Cases for Case Based Forecasting with Neural Network Support, pp. 293–319. Springer London, London (2001)
4. Craw, S., Wiratunga, N., Rowe, R.: Learning adaptation knowledge to improve case-based reasoning. Artificial Intelligence 170, 1175–1192 (2006)
5. Cunningham, P., Doyle, D., Loughrey, J.: An evaluation of the usefulness of case-based explanation. In: Case-Based Reasoning Research and Development: Proceedings of the Fifth International Conference on Case-Based Reasoning, ICCBR-03. pp. 122–130. Springer-Verlag, Berlin (2003)
6. Dua, D., Graff, C.: UCI machine learning repository (2017)

7. D'Aquin, M., Badra, F., Lafrogne, S., Lieber, J., Napoli, A., Szathmary, L.: Case base mining for adaptation knowledge acquisition. In: Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-07). pp. 750–755. Morgan Kaufmann, San Mateo (2007)
8. Fuchs, B., Lieber, J., Mille, A., Napoli, A.: Differential adaptation: An operational approach to adaptation for solving numerical problems with CBR. Knowledge-Based Systems (2014), in press
9. Hanney, K., Keane, M.: Learning adaptation rules from a case-base. In: Proceedings of the Third European Workshop on Case-Based Reasoning. pp. 179–192. Springer, Berlin (1996)
10. Jalali, V., Leake, D.: Enhancing case-based regression with automatically-generated ensembles of adaptations. Journal of Intelligent Information Systems pp. 1–22 (2015)
11. Jalali, V., Leake, D., Forouzandehmehr, N.: Learning and applying case adaptation rules for classification: An ensemble approach. In: Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17. pp. 4874–4878 (2017)
12. Keane, M.T., Smyth, B.: Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable ai (xai) (2020)
13. Kenny, E.M., Keane, M.T.: On generating plausible counterfactual and semi-factual explanations for deep learning (2020)
14. Kingma, D., Ba, J.: Adam: A method for stochastic optimization (2017)
15. Leake, D., Kinley, A., Wilson, D.: Acquiring case adaptation knowledge: A hybrid approach. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence. pp. 684–689. AAAI Press, Menlo Park, CA (1996)
16. Leake, D., Ye, X., Crandall, D.: Supporting case-based reasoning with neural networks: An illustration for case adaptation (2021)
17. Liao, C., Liu, A., Chao, Y.: A machine learning approach to case adaptation. In: 2018 IEEE First International Conference on Artificial Intelligence and Knowledge Engineering (AIKE). pp. 106–109 (2018)
18. López de Mántaras, R., McSherry, D., Bridge, D., Leake, D., Smyth, B., Craw, S., Faltings, B., Maher, M., Cox, M., Forbus, K., Keane, M., Aamodt, A., Watson, I.: Retrieval, reuse, revision, and retention in CBR. Knowledge Engineering Review 20(3) (2005)
19. Mark, W., Simoudis, E., Hinkle, D.: Case-based reasoning: Expectations and results. In: Leake, D. (ed.) Case-Based Reasoning: Experiences, Lessons, and Future Directions, pp. 269–294. AAAI Press, Menlo Park, CA (1996)
20. McSherry, D.: An adaptation heuristic for case-based estimation. In: Proceedings of the Fourth European Workshop on Advances in Case-Based Reasoning. pp. 184–195. EWCBR '98, Springer-Verlag, London, UK, UK (1998)
21. Nugent, C., Doyle, D., Cunningham, P.: Gaining insight through case-based explanation. J. Intell. Inf. Syst. 32, 267–295 (06 2009)
22. Patterson, D., Rooney, N., Galushka, M.: A regression based adaptation strategy for case-based reasoning. p. 87–92. American Association for Artificial Intelligence, USA (2002)
23. Policastro, C., Carvalho, A., Delbem, A.: Automatic knowledge learning and case adaptation with a hybrid committee approach. Journal of Applied Logic 4(1), 26–38 (2006)
24. Wetzel, S.J., Ryczko, K., Melko, R.G., Tamblyn, I.: Twin neural network regression (2020)

25. Ye, X.: The enemy of my enemy is my friend: Class-to-class weighting in k-nearest neighbors algorithm. In: Proceedings of the Thirty-First International Florida Artificial Intelligence Research Society Conference, FLAIRS 2018. pp. 389–394 (2018)
26. Ye, X.: C2C trace retrieval: Fast classification using class-to-class weighting. In: Proceedings of the Thirty-Second International Florida Artificial Intelligence Research Society Conference, FLAIRS 2019. pp. 353–358 (2019)
27. Ye, X., Leake, D., Huibregtse, W., Dalkilic, M.: Applying class-to-class siamese networks to explain classifications with supportive and contrastive cases. In: International Conference on Case-Based Reasoning. pp. 245–260. Springer (2020)
28. Zhang, C., Liu, C., Zhang, X., Almpanidis, G.: An up-to-date comparison of state-of-the-art classification algorithms. Expert Systems with Applications 82 (04 2017)
29. Zhang, F., Ha, M., Wang, X., Li, X.: Case adaptation using estimators of neural network. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No.04EX826). vol. 4, pp. 2162–2166 vol.4 (Aug 2004)