Comparing Similarity Calculation Methods in Conversational CBR

Mingyang Gu, Xin Tong, and Agnar Aamodt
Department of Computer and Information Science, Norwegian University of Science and Technology, Sem Saelands vei 7-9, N-7491, Trondheim, Norway
Email: mingyang,tongxin,agnar@idi.ntnu.no

***Abstract*— Conversational Case-Based-Reasoning (CCBR) provides a mixed-initiative dialog for guiding users to construct their problem description incrementally through a question-answering sequence. Similarity calculation in CCBR, as in traditional CBR, plays an important role in the retrieval process since it decides the quality of the retrieved case. In this paper, we analyze the different characteristics of the query (new case) between CCBR and traditional CBR, and argue that the similarity calculation method that only takes the features appearing in the query into account, so called query-biased, is more suitable for CCBR. An experiment is designed and executed on 36 datasets. The results show us that on 31 datasets out of the total 36, the CCBR system using the query-biased similarity calculation method achieves more effective performance than those using case-biased and equally-biased similarity calculation methods.**

I. Introduction

The basic idea underlying case-based reasoning (CBR) [1], [2] is to reuse the solution to the previous most similar problem in helping solve the current problem. Before we can reuse any existing solutions, we have to find the most similar previous case based on the current problem description.

In traditional CBR processes, users are assumed to be able to provide a well-defined problem description, and based on such a description a CBR system can find the most appropriate previous case (base case). But this assumption is not always realistic. In some situations, users only have vague ideas about their target problems at the beginning of retrieval, and tend to describe them using surface features.

Conversational Case-Based Reasoning (CCBR) [3] provides a mixed-initiative dialog for guiding users to construct their problem description incrementally through a question-answering sequence. In CCBR, a user provides one or several explicit features as her initial query (new case). The CCBR system uses the initial query to retrieve the first set of candidate cases, and identifies a group of informative features from them to generate discriminative questions. Both the retrieved cases and identified discriminative questions are ranked and shown to the user. The user either finds the base case to terminate the retrieval process or chooses a question, which she considers relevant to her task and can answer explicitly, and provides the answer to it (CCBR systems usually also prompt the alternative answer options that correspond to the feature values available in the case base). An updated query is constructed through combining the previous query with the newly gained answer. Subsequent rounds of retrieving and question-answering will cut down the returned case set iteratively until the user finds her desired base case, or no discriminative questions are available. That is, instead of letting a user guess how to describe her target problem, CCBR discovers a sequence of discriminative questions helping extract information from the user to construct the problem description incrementally. CCBR applications have been successfully fielded, e.g., in the troubleshooting domain [4], [5] and in the products and services selection [6], [7].

In both traditional CBR and CCBR, one key research topic is to calculate the similarities between a query and stored cases to decide which case is most similar to the current problem. Normally, the similarity between a query and a stored case is measured by the accumulated similarities on all counted features. On the one hand, the similarity is influenced by different methods to calculate the similarity on each feature. For example, in syntactic methods two cases can be thought similar on one nominal feature only when they have the same value on that feature [8], while in knowledge-intensive methods, two cases with various values on one nominal feature can possibly be considered as similar through exploring general domain knowledge [9], [10]. On the other hand, the similarity is also influenced by the counted feature scope, i.e. set of the features appearing in the query, in the case, or in both of them. In this paper, from the perspective of counted feature scope, we provide a framework to classify the similarity calculation methods into three categories: case-biased (features in the stored case), query-biased (features in the query) and equally-biased (features in both the query and the stored case).

CCBR research is currently to a large extent focusing on the discriminative question selecting and ranking to minimize the cognitive load demanded on users to retrieve the base case [6], [11], for example, selecting the most informative questions to ask [6], [12], [13], [14], [15], or using feature inferencing to avoid asking users the questions which can be answered implicitly using the currently known information [12], [15]. To our knowledge, there are so far no published results on how different similarity calculation methods influence on the performance of a CCBR system.

In this paper, we analyze the differences on query characteristics between traditional CBR and CCBR, and hypothesize that the similarity calculation method only taking the query features into account is more suitable for CCBR. An experi-

0-7803-9093-8/05/$20.00 ©2005 IEEE.

ment is designed and executed in an attempt to evaluate this hypothesis.

The rest of this paper is organized as follows. In Section 2, we present a formal framework to classify the similarity calculation methods in CBR into three categories from the feature scope point of view; in Section 3, we focus on the query differences between CCBR and traditional CBR, and hypothesize that query-biased similarity calculation method is more suitable for CCBR; in Section 4, we design an experiment to evaluate this hypothesis, and the results are also analyzed and discussed. At the end we draw our conclusions in Section 5.

II. Similarity Calculation Framework in CBR

Generally, a case in CBR can be represented using the following three parts conceptually [2]:

Problem description: the state of the world at the time the case was happening and, if appropriate, what problem needed to be solved at that time

Solution description: the stated or derived solution to the problem specified in the problem description

Outcome: the resulting state of the world after the solution was carried out

A query (new case) only has the first part. The similarity measurement between a query and a stored case is based on the comparison of the problem description part of them. In our research, we assume that the problem description of a case takes the form of a set of $< feature, value >$ pairs. It is not necessary for both a query and a case to have the same feature set.

We further define that:

Nq: set of features appearing in a query

Nc: set of features appearing in a stored case

One concept that is closely related to similarity is distance. The greater the distance between a query and a stored case, the less the similarity between them is. The main use of the similarity measurement in CBR is to sort the retrieved cases. From that point of view, the similarity and distance measurements have an inverse relationship, and either of them may be chosen. We adopt the distance measurement in our research, as defined by the following formula:

$$distance(q,c) = \sqrt{\frac{\sum_{f \in FS} w_f dif^2(q_f, c_f)}{\sum_{f \in FS} w_f}} \tag{1}$$

where q, c, f, FS and w_f denote a query, a stored case, a particular feature, a selected feature set and the weight for the feature f respectively.

In addition, $dif(q_f, d_f)$ is a function used to compute the difference between a query and a stored case on a feature f, which is defined as following in our research:

$$dif(q_f, c_f) = \begin{cases} |q_f - c_f| & f \text{ is a numerical feature (normalized)} \\ 0 & f \text{ is a nominal feature, and } q_f = c_f \\ 1 & f \text{ is a nominal feature, and } q_f \neq c_f \\ 1 & \text{c or q has missing value on } f \end{cases} \tag{2}$$

Based on three types of value assignment methods to FS in Equation 1, we divide the similarity measurement methods in CBR into three categories.

A. Case-Biased Similarity Calculation Methods

In case-biased similarity calculation methods, $FS = N_c$, and Equation 1 is transformed as follows:

$$distance(q,c) = \sqrt{\frac{\sum_{f \in N_c} w_f dif^2(q_f, c_f)}{\sum_{f \in N_c} w_f}} \tag{3}$$

In this type of calculation method, the features appearing in the stored case are the basis for the similarity calculation (here comes the name of 'case-biased'). This type of methods are used in [3], [8]. The basic idea behind it is that the problem description of a stored case is a sufficient condition for the corresponding solution actions, so to what degree the problem description is satisfied by the query decides whether the solution in the stored case is suitable for the current problem.

B. Query-Biased Similarity Calculation Methods

In query-biased similarity calculation methods, $FS = N_q$, only the features appearing in the query are taken into account. This type of similarity calculation method focuses on the query, and the intuitive idea underlying it is that whether a stored case can be retrieved is decided by to what degree the query specified by a user is satisfied by this stored case.

C. Equally-Biased Similarity Calculation Methods

In equally-biased similarity calculation methods, $FS = N_q \cup N_c$, that is, both the features appearing in the query and those in the stored case are taken into account (the case and the query are treated equally). This type of similarity calculation methods are used in [16], [17]. The basic idea behind such type of methods is that the degree to which the query and the case are similar decides whether the solution to that case can be reused in the current target problem.

III. Using Query-Biased Similarity Calculation Methods in CCBR

CCBR considers the situation where users can not well define their queries, and alternatively provides a multi-retrieval process to help users construct their queries incrementally through a sequence of question-answering cycles. The important difference between CCBR and traditional CBR is that *the query used in CCBR is assumed incomplete, that is, the CCBR query only represents the user's currently identified features.*

TABLE I

A FRUIT RETRIEVAL EXAMPLE IN CCBR

	Taste	Color	Place	With Fur	With Core	Water Inside	Shape
Query	sweaty	red	Asia		yes		
apple	sweaty	red	Asia	yes	yes	no	round
kiwi	sweaty	brown	America				
banana	sweaty				no		

TABLE II

DISTANCE MEASUREMENTS WITH FOUR FEATURES IN THE EXAMPLE

	apple	kiwi	banana
Query-biased	0	$\frac{3}{4}$	$\frac{3}{4}$
Case-biased	$\frac{3}{7}$	$\frac{2}{3}$	$\frac{1}{2}$
Equally-biased	$\frac{3}{7}$	$\frac{3}{4}$	$\frac{3}{4}$

TABLE III

DISTANCE MEASUREMENTS WITH TWO FEATURES IN THE EXAMPLE

	apple	kiwi	banana
Query-biased	0	$\frac{1}{2}$	$\frac{1}{2}$
Case-biased	$\frac{5}{7}$	$\frac{2}{3}$	$\frac{1}{2}$
Equally-biased	$\frac{5}{7}$	$\frac{2}{3}$	$\frac{2}{3}$

The features without specified values in a CCBR query do not necessarily mean that they have "missing-value" as in traditional CBR. They may have values, but we have not assigned the value for them in current CCBR stage. The case-biased and equally-biased similarity calculation methods assume that the query is fully specified and all the features that appear in the case but not in the query are considered to have the "missing value". So the difference measurement on each such feature is assigned the same value, e.g. 1. If we proceed further in CCBR, and specify values on more features, the really calculated distance measurement will benefit the base case since it has the higher potential to get less difference measurements on these newly specified features than other cases. The distance between the partially specified query and a stored case is heavily influenced by the number of the features appearing in the case but not yet specified in the query. The query-biased similarity calculation method can avoid the influence of these features, and rank the case that most satisfies the currently partially specified query with the highest priority.

For example, as illustrated by Table I, the potential fully specified query for searching a desired fruit has four features. The distance measurements using three different similarity calculation methods are shown in II, in which we assume that each feature has the same weight ($\frac{1}{7}$). We can see that the most similar case with the fully specified query is apple, no matter which similarity calculation method is adopted.

At one stage of the conversation in CCBR a user may only specify two features, "Taste=sweaty, Color=red". The distance measurements between the query and each fruit are shown in Table III. If query-biased method is adopted, the most similar case to the query has already been the base case, i.e. apple.

The most similar case will be kiwi or banana if equally-biased method is used, and banana if case-biased method is adopted. With the conversation going on, the system further prompts two questions "where does the fruit come from?" and "does the fruit have a core inside?", and the user answers these questions with "Place=Asia, With Core=yes". Until this stage, the base case, i.e. apple, can be ranked with the highest priority (as shown in Table II). But the user may be angry at her 'tricky searching assistant', "since only apple satisfies the first two specified features, why do dot you show me at that time and still bother me to answer extra two questions?", and her satisfaction level will be reduced.

We can see from the above example that the query-biased similarity calculation method can avoid the influence of the features that appear in the case but not in the partially specified query in CCBR. Since the base case also has a higher potential to have higher similarity on partial set of query features than other cases, it is reasonable to believe that a CCBR system that uses the query-biased similarity calculation method can show users the base case on earlier conversation stage than those using case-biased or equally-biased methods. So our hypothesis tested in this paper is that using the query-biased similarity calculation method, a CCBR system can improve its performance, that is, using less conversation sessions to find the base case than using equally-biased or case-biased methods.

IV. EXPERIMENT DESIGN AND RESULTS ANALYSIS

Our experiment is designed with the objective to compare the conversation lengths of CCBR systems using different similarity calculation methods. The best way to do that is with human subjects. Unfortunately, we can not get sufficient subjects to run the experiment. Therefore, we use a variant of the leave-one-out cross validation (LOOCV) method to simulate the human-computer conversation process, the similar methods to which have been successfully used by the CCBR community [12]. The designed evaluation process is carried out on 36 datasets, and the results provide a significant support to our hypothesis.

Experiment Design

The LOOCV proceeds with a series of conversations, each conversation starting with selecting a case from the case base as the target case and the remaining cases forming the case base to be searched. The initial query is constructed through selecting the predefined number of features from that target

TABLE IV

PSEUDO CODE OF THE EVALUATION PROCESS.

```
Procedure evaluation(CaseBase)
    SuccessOnEqually,SuccessOnQuery,SuccessOnCase=false
    TestCases,SessionsOnEqually,SessionsOnCase,SessionsOnQuery=0
    GlobalWeights=weighting(CaseBase)
    for each case X ∈ CaseBase
        X_n=weighted1NNOnEqually(X,CaseBase-X)
        if Solution(X_n) = Solution(X) then
            TestCases=TestCases+1
            X_q=featureSelection(InitialFeatureNumber)
            do while not (SuccessOnEqually and SuccessOnCase
                and SuccessOnQuery) and X_q ≠ null
                if not SuccessOnEqually then
                    ReturnedCasesOnEqually=
                      weightedkNNOnEqually(X_q, CaseBase-X)
                    SessionsOnEqually=SessionsOnEqually+1
                    if X_n ∈ ReturnedCasesOnEqually then
                        SuccessOnEqually=true
                    End If
                End If
                if not SuccessOnCase then
                    ReturnedCasesOnCase=
                      weightedkNNOnCase(X_q, CaseBase-X)
                    SessionsOnCase=SessionsOnCase+1
                    if X_n ∈ ReturnedCasesOnCase then
                        SuccessOnCase=true
                    End If
                End If
                if not SuccessOnQuery then
                    ReturnedCasesOnQuery=
                      weightedkNNOnQuery(X_q, CaseBase-X)
                    SessionsOnQuery=SessionsOnQuery+1
                    if X_n ∈ ReturnedCasesOnQuery then
                        SuccessOnQuery=true
                    End If
                End If
                X_q=X_q+featureSelection(1)
            End Loop
        End IF
    End Loop
Return SessionsOnEqually/TestCases, SessionsOnCase/TestCases, SessionsOnQuery/TestCases
```

case. Based on this initial query, a retrieval process is carried out and the first k most similar cases are returned. If the base case is included in the returned case set, which means users find their desired case, the conversation process is finished successfully. Otherwise, a new feature is selected from the target case and added into the query to simulate a question-answering session between a human subject and a computer, and the updated query is used to start a new round of retrieval. The selecting, adding, and retrieving cycle continues until the base case appears in the returned case set, or no features are left to be selected.

There are three questions we further need to address in the experiment design: the retrieval algorithm, the feature selection strategy, and the base case determination.

Retrieval Algorithm: A weighted k-NN algorithm is introduced in our experiment to complete the case retrieval task, in which the first k most similar cases are returned. The number k is used to simulate the number of cases that will be shown to users on each conversation session in CCBR. In our experiment setting, we set k to 3. We use a feature weighting method, similar to EACH [18], to get a set of global weights, each corresponding to one feature appearing in the case base.

In EACH, given a test case from the case base, its most similar case is selected from the remaining stored cases using a weighted 1-NN algorithm. If the most similar case suggests the same solution as the test case, the weight of each matched feature is increased by a fixed positive amount, while weights for mismatched features are decreased by the same amount.

Three variants of this basic algorithm are constructed based on the three different similarity calculation methods introduced in Section 2.

Feature Selection Strategy: The feature selection strategy is used to decide which feature should be selected from a set of candidate features, and added into the current query to simulate a question-answering process. In our experiment, a weight-biased random selection strategy is designed. For example, there are three features, A, B, and C in the candidate feature set with the weight values, 0.1, 0.2, and 0.05, respectively (learned from the feature weighting process). According to the weight-biased random feature selection strategy, feature A, B, C will be selected with the possibilities $\frac{2}{7}$, $\frac{4}{7}$, and $\frac{1}{7}$, respectively. Such a feature selection strategy simulates a question-answering process: a CCBR system ranks the more informative questions (transformed from features) with higher priority, and a user prefers to select the most relevant or important feature to answer first.

Base Case Determination: For each case in the case base, its base case is defined as the one returned by a weighted 1-NN algorithm using equally-biased similarity calculation (here the query is fully specified and complete) and with the same solution-feature value. Therefore, not all the cases in the case base can act as a target case to simulate a conversation. The cases that can not find its corresponding base case from the remaining cases (its nearest neighbor has a different solution-feature value) will be dropped out from the leave-one-out cross validation.

Since we choose the base case as the retrieval result of the 1-NN algorithm using the equally-biased similarity method, the simulated conversation process using the equally-biased method will terminate with the base case appearing among the returned case set within all the candidate features are added into the query. It is not guarantied that the conversation process using the case-biased or the query-biased method can terminate with the base case found in the returned case set.

In our experiment, we assign the biggest conversation length (the number of conversation sessions when all candidate features are added into the query) to the unsuccessful conversations. That is, the base case selection mechanism benefits the CCBR system using the equally-biased similarity calculation method. However, the experiment results show us that even with such biased base case selection strategy, the average conversation length using the query-biased similarity method is shorter than that using other two methods, and the average conversation length using the case-biased method is almost the same with that using equally-biased method, as illustrated in

Table V.

The pseudo code for the experiment process is listed in Table IV.

Experiment Environment and Dataset

We implement our evaluation algorithm inside the Weka framework [19], and test it using all the 36 datasets provided by Weka project, originally from the UCI repository [20].

All the numeric features in these datasets are normalized using the corresponding filter provided in Weka3.4.3 according to the requirement of the similarity calculation algorithm. The statistical information about the test datasets is illustrated in the left part of Table V, in which the first 6 columns denote respectively: the name of each dataset (Dataset), the number of the cases (Total cases), the total number of the features excluding the solution feature (Features), the number of the numeric features and nominal features (Numeric/Nominal), the percentage of the missing data (Missing Data) calculated using equation: $\frac{number\ of\ the\ missing\ values}{Total\ cases*Features}$, and the number of solutions (Solutions).

Experiment Results

The experiment results are listed in the right part of Table V, in which the columns: Test cases, Equally Biased, Case Biased, and Query Biased, denote tested cases (corresponding to TestCases in Table IV), the average conversation lengths using the corresponding similarity calculation method for each dataset.

To show the comparison results more clearly, we add three columns into Table V, Case-Query, Equally-Query, and Case-Equally, to illustrate the differences of the average conversation lengths between each pair of similarity calculation methods. For example, the Case-Query column contains the results of subtracting the average conversation length using the query-biased method from that using the case-biased method on each dataset. And the last row gives the average values of corresponding columns.

Out of 36 datasets, there are 31 datasets in which the query-biased similarity calculation method uses less conversation sessions to find the base case than other two methods (average using 3.66, 3.43 less conversation sessions respectively). That gives us a straightforward evidence that the CCBR system using query-biased method is more effective than those using equally-biased and case-biased methods.

The conversation lengths between CCBR systems using the case-biased method and the equally-biased method do not have clear difference since there are no difference at all on 22 datasets out of 36, and the average difference over 36 datasets is only 0.23. Even if the results show us that the equally-biased method is a little more effective than the case-biased one, but considering that the base case determination strategy benefits the equally-biased similarity calculation method, the experiment results can not provide strong evidence to say that there is performance difference between these two methods.

Further more, we carry out the statistical hypothesis test to evaluate our predefined hypothesis in Section 3. The whole hypothesis is divided into three sub hypotheses to test:

H1: the CCBR system using the query-biased similarity calculation method can achieve more effective performance than that using the equally-biased method, that is, using less conversation sessions to find the base case.

H2: the CCBR system using the query-biased similarity calculation method can use less conversation sessions to find the base case than that using the case-biased method.

H3: there exists performance difference between the CCBR system using the case-biased similarity calculation method and that using the equally-biased method, that is, these two methods use different number of conversation sessions to find the base case.

We choose the values appearing in the column: Equally-Query, Case-Query, and Case-Equally respectively in Table V, as the parameter values to execute the significance test. The test results (reported in Table VI) show us that the first two sub hypotheses are accepted, and the last one is refused given the significance level of 0.01.

Conclusions

In this paper, we provide a framework to classify the similarity calculation methods used in CBR from the perspective of counted feature scope. And based on the special characteristic of the CCBR query, partially specified and incomplete, we hypothesize that CCBR system using the query-biased similarity calculation method can achieve higher performance than those using case-biased or equally-biased methods. The experiment provides a significant support to our hypothesis. While the conversation process in the experiment is simulated by a leave-one-out cross validation process, an experiment executed on human subjects will provide more evidence to evaluate our hypothesis.

References

[1] A. Aamodt and E. Plaza, "Case-based reasoning: Foundational issue, methodological variations, and system approaches," *AI Communications*, vol. 7, no. 1, pp. 39–59, 1994.

[2] J. Kolodner, *Case-based reasoning*. Morgan Kaufmann Publishers Inc., 1993.

[3] D. W. Aha, L. A. Breslow, and H. Munoz-Avila, "Conversational case-based reasoning," *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, vol. 14, no. 1, p. 9, 2001.

[4] K. M. Gupta, "Knowledge-based system for troubleshooting complex equipment," *International Journal of Information and Computing Science*, vol. 1, no. 1, pp. 29–41, 1998.

[5] P. Cunningham and B. Smyth, "A comparison of model-based and incremental case-based approaches to electronic fault diagnosis," in *Case-Based Reasoning Workshop*, Seattle, USA, 1994.

[6] H. Shimazu, "Expertclerk: A conversational case-based reasoning tool for developing salesclerk agents in e-commerce webshops," *Artificial Intelligence Review*, vol. 18, no. 3-4, pp. 223 – 244, 2002.

[7] M. Gu, A. Aamodt, and X. Tong, "Component retrieval using conversational case-based reasoning," in *International Conference on Intelligent Information Systems*, Z. Shi, Ed., Beijing, China, 2004.

[8] D. W. Aha, "Case-based learning algorithms," in *the DARPA Case-Based Reasoning Workshop*, Morgan Kaufmann, 1991.

[9] A. Aamodt, "Explanation-driven case-based reasoning," *Topics in Case-based reasoning*, pp. 274–288, 1994.

[10] A. Aamodt, "Knowledge-intensive case-based reasoning in creek," in *7th European Conference on Case-Based Reasoning*, P. Funk and P. A. G. Calero, Eds., vol. 1 - 15. Madrid, Spain: Spinger, 2004.

TABLE V

DATASET DESCRIPTION AND EXPERIMENT RESULTS

Dataset	Total cases	Features	Numeric/ Nominal	Missing data	Solut -ions	Test cases	Equally Biased	Case Biased	Query Biased	Case- Query	Equally -Query	Case- Equally
Anneal	898	38	6 / 32	64.98%	5	891	20.76	20.76	20.55	0.21	0.21	0
Anneal Original	898	38	9 / 29	73.13%	6	824	8.36	8.37	7.26	1.10	1.09	0.01
Audiology	226	69	0 / 69	2.03%	24	180	14.01	13.58	17.43	**-3.85**	**-3.42**	-0.43
Autos	205	25	15 / 10	1.15%	7	150	7.13	7.12	2.63	4.49	4.49	-0.01
Balance Scale	625	4	4 / 0	0	3	478	3.58	3.58	3.75	**-0.17**	**-0.17**	0
Breast Cancer	286	9	0 / 9	0.35%	2	190	5.99	6.00	5.88	0.12	0.11	0.01
Breast-W	699	9	9 / 0	0.25%	2	665	8.70	8.70	4.82	3.88	3.88	0
Credit Approval	690	15	6 / 9	0.65%	2	549	11.96	11.98	8.86	3.11	3.10	0.01
Credit German	1000	20	7 / 13	0	2	682	13.11	13.11	11.32	1.79	1.79	0
Diabetes	768	8	8 / 0	0	2	554	7.87	7.87	5.15	2.72	2.72	0
Glass	214	9	9 / 0	0	7	150	8.37	8.37	3.41	4.96	4.96	0
Heart Statlog	270	13	13 / 0	0	2	204	9.51	9.51	6.74	2.77	2.77	0
Heart h	294	13	6 / 7	20.46%	5	220	7.43	7.44	4.92	2.52	2.51	0.01
Heart c	303	13	6 / 7	0.18%	5	233	7.60	7.60	6.04	1.56	1.56	0
Hepatitis	155	19	6 / 13	2.43%	2	125	10.14	10.74	9.06	1.69	1.08	0.61
Horse Colic	368	22	7 / 15	23.80%	2	290	11.77	13.29	9.53	3.76	2.24	1.52
Horse Colic Original	368	27	7 / 20	19.39%	2	258	13.26	13.41	10.61	2.80	2.65	0.15
Hypothyroid	3772	29	7 / 22	5.54%	4	3335	15.22	15.22	14.74	0.48	0.48	0
Ionosphere	351	34	34 / 0	0	2	312	24.95	24.95	7.65	17.29	17.29	0
Iris	150	4	4 / 0	0	3	143	3.75	3.75	2.14	1.61	1.61	0
Kr-vs-kp	3196	36	0 / 36	0	2	3146	21.86	21.86	20.63	1.23	1.23	0
Labor	57	16	8 / 8	35.75%	2	52	5.88	6.60	3.06	3.54	2.83	0.71
Letter	20000	16	16 / 0	0	8	17380	14.09	14.09	9.26	4.84	4.84	0
Lymph	148	18	3 / 15	0	4	123	9.76	9.76	9.82	**-0.07**	**-0.07**	0
Mushroom	8124	22	0 / 22	1.39%	2	8124	16.07	16.07	13.09	2.97	2.97	0
Primary Tumor	339	17	0 / 17	3.90%	8	125	5.94	5.90	7.22	**-1.31**	**-1.28**	-0.03
Segment	2310	19	19 / 0	0	7	2217	17.55	17.55	7.45	10.10	10.10	0
Sick	3772	29	7 / 22	5.54%	2	3421	14.66	14.67	14.75	**-0.08**	**-0.09**	0.01
Sonar	208	60	60 / 0	0	2	182	39.68	39.68	12.87	26.81	26.81	0
Soybean	683	35	0 / 35	9.78%	19	623	9.44	13.98	6.56	7.42	2.88	4.54
Splice	3190	61	0 / 61	0	3	2877	29.24	29.24	23.59	5.65	5.65	0
Vehicle	846	18	18 / 0	0	4	602	15.93	15.93	6.71	9.22	9.22	0
Vote	435	16	0 / 16	5.63%	2	412	10.15	11.31	8.95	2.36	1.20	1.16
Vowel	990	13	10 / 3	0	11	983	8.58	8.58	3.57	5.01	5.01	0
Wave Form 5000	5000	40	40 / 0	0	3	3157	12.91	12.91	11.98	0.93	0.93	0
Zoo	101	17	1 / 16	0	7	98	7.98	7.98	7.74	0.23	0.23	0
AVERAGE							12.59	12.82	9.16	3.66	3.43	0.23

TABLE VI

HYPOTHESIS TEST RESULT

Alternative hypothesis	Null hypothesis	Tailed type	Degree of freedom	Significance level	Critical value	t-value	Result
$H1 : V_{Equally-Query} > 0$	$V_{Equally-Query} = 0$	one-tailed	35	0.01	2.44	3.81	refuse null-hypothesis
$H2 : V_{Case-Query} > 0$	$V_{Case-Query} = 0$	one-tailed	35	0.01	2.44	4.05	refuse null-hypothesis
$H3 : V_{Case-Equally} \neq 0$	$V_{Case-Equally} = 0$	two-tailed	35	0.01	2.727	1.68	can not refuse null-hypothesis

[11] S. Schmitt, "simvar: A similarity-influenced question selection criterion for e-sales dialogs," *Artificial Intelligence Review*, vol. 18, no. 3-4, pp. 195–221, 2002.

[12] D. W. Aha, T. Maney, and L. Breslow, "Supporting dialogue inferencing in conversational case-based reasoning," in *European Workshop on Case-Based Reasoning*, Dublin, Ireland, 1998, pp. 262–273.

[13] P. Cunningham, R. Bergmann, S. Schmitt, R. Traphoner, S. Breen, and B. Smyth, "Websell: Intelligent sales assistants for the world wide web," *KI - Kunstliche Intelligenz*, vol. 1, pp. 28–31, 2001.

[14] M. H. Goker and C. A. Thompson, "Personalized conversational case-based recommendation," in *the 5th European Workshop on Case-Based Reasoning(EWCBR 2000)*, Trento, Italy, 2000.

[15] M. Gu and A. Aamodt, "A knowledge-intensive method for conversational cbr," in *International Conference on Case-Based Reasoning*, Chicago Illinois, 2005.

[16] M. M. Richter and S. Web, "Similarity, uncertainty and case-based reasoning in padtex," Kaiserslautern, Tech. Rep., 1993.

[17] Q. Yang and J. Wu, "Enhancing the effectiveness of interactive case-based reasoning with clustering and decision forests," *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, vol. 12, pp. 49 – 64, 2001.

[18] S. Salzberg, "A nearest hyperrectangle learning method," *Mach. Learn.*, vol. 6, no. 3, pp. 251–276, 1991.

[19] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools with Java implementations*. San Francisco: Morgan Kaufmann, 2000.

[20] C. Blake and C. Merz, "Uci repository of machine learning databases [http://www.ics.uci.edu/ mlearn/mlrepository.html]," 1998.