Consistency Measures for Feature Selection: A Formal Definition, Relative Sensitivity Comparison and a Fast Algorithm

Kilho Shin, Danny Fernandes and Seiya Miyazaki
University of Hyogo, Kobe, Japan
{yshin,danny}@ai.u-hyogo.ac.jp
Panasonic Corporation, Osaka, Japan
miyazaki.seiya@jp.panasonic.com

Abstract

Consistency-based feature selection is an important category of feature selection research yet is defined only intuitively in the literature. First, we formally define a consistency measure, and then using this definition, evaluate 19 feature selection measures from the literature. While only 5 of these were labeled as consistency measures by their original authors, by our definition, an additional 9 measures should be classified as consistency measures. To compare these 14 consistency measures in terms of sensitivity, we introduce the concept of *quasi-linear compatibility order*, and partially determine the order among the measures. Next, we propose a new fast algorithm for consistency-based feature selection. We ran experiments using eleven large datasets to compare the performance of our algorithm against INTERACT and LCC, the only two instances of consistency-based algorithms with potential real world application. Our algorithm shows vast improvement in time efficiency, while its performance in accuracy is comparable with that of INTERACT and LCC.

1 Introduction

Designing lean, efficient and accurate feature selection algorithms is one of the central problems in machine learning theory. The literature has three major approaches for designing feature selection algorithms: the *filter*, *wrapper* and *embedded* approaches (*e.g.* [Molina *et al.*, 2002]). The wrapper and embedded approaches are rather pragmatic in nature, aimed at optimizing the output of the learning algorithms that will be used. By contrast, filter algorithms study intrinsic properties of datasets, and attempt to find optimal feature subsets that yield good learning results, regardless of choice of learning algorithms. Also, the filter approach has a practical advantage over the other two in that the algorithms are usually much faster. In this paper, our interest is in the filter approach.

Filter approach algorithms can be classified into two categories according to how they evaluate relevance of feature subsets. One group evaluates relevance for individual features, and selects features based on an evaluation rule (*e.g.* select the features with higher relevance than a threshold). In the second group, algorithms use measures to evaluate relevance of subsets of features. The advantage of the first group is higher time efficiency, since an algorithm evaluates relevance only as many times as the features appearing in a dataset.

Nevertheless, algorithms of the first group have the weakness that they may pass over relevant features when features *interact* to determine class labels ([Zhao and Liu, 2007]). For example, let F_1 and F_2 be mutually independent uniformly random binary features, that determine binary class labels C as in $\mathrm{C} = \mathrm{F}_1 \oplus \mathrm{F}_2$, where $\oplus$ denotes the binary addition. Although F_1 and F_2 together determine the class labels, considered individually, they are completely irrelevant in determining class labels.

Consistency-based feature selection was proposed as a solution to this problem (*e.g.* [Zhao and Liu, 2007]), and several consistency measures are proposed ([Liu *et al.*, 1998; Pawlak, 1991; Arauzo-Azofra *et al.*, 2008; Shin and Xu, 2009]). In consistency-based feature selection, consistency measures are used to evaluate relevance of feature subsets. A consistency measure is intuitively defined as a metric to measure the *distance* of a feature subset from the consistent state. A feature set $\{\mathrm{F}_1, \ldots, \mathrm{F}_n\}$ is said to be *consistent*, when

$$\Pr(\mathrm{C} = c \mid \mathrm{F}_1 = f_1, \ldots, \mathrm{F}_n = f_n) = 0 \text{ or } 1$$

holds for all $c, f_1, \ldots, f_n$. When a feature subset is consistent, the inconsistency value is 0, and as an inconsistent feature subset approaches the consistent state, the measure decreasingly approaches 0. To illustrate, $\{\mathrm{F}_1, \mathrm{F}_2\}$ in our previous example is measured to be 0, whereas the measure for $\{F_1\}$ and $\{F_2\}$ should be high. A consistency-based algorithm solves the aforementioned problem by selecting feature subsets with a sufficiently small inconsistency value.

Despite the advantage of the consistency-based approach in handling interaction effects, there are three important outstanding issues regarding the measure:

1. We only have an intuitive definition for consistency measures. A formal definition is lacking.
2. Although several measures are proposed as instances of consistency measures, we don't have a basis to compare them.
3. Due to the relatively heavy computational cost required to evaluate a consistency measure, consistency-based algorithms exhibit low time efficiency.

We address these issues as follows.

1. We propose the *determinacy* and *monotonicity* conditions as axioms for the definition of consistency measures. Using this definition, we identify additional feature selection measures in the literature as inconsistency measures.
2. To compare these inconsistency measures in terms of sensitivity, we introduce the concept of *quasi-linear compatibility order*.
3. We propose a new consistency-based algorithm that shows vast improvement in time-efficiency over other consistency-based algorithms, while exhibiting comparable performance in accuracy.

2 Consistency measures defined

In this section, we provide a formal definition for consistency measures, and use this definition to reclassify feature selection measures listed in the detailed survey by Molina et al. ([Molina *et al.*, 2002]). For convenience, we use the notations shown below.

Symbol	Definition
$\boldsymbol{\Omega}$	The countable population of features.
$\mathfrak{P}_0(\boldsymbol{\Omega})$	The set of all of the finite subsets of $\boldsymbol{\Omega}$.
$\boldsymbol{X}$	An element of $\mathfrak{P}_0(\boldsymbol{\Omega})$.
C	A random variable to represent a class label.
$\mathcal{P}(\boldsymbol{\Omega}, \mathrm{C})$	The set of probability distributions over $\boldsymbol{\Omega} \cup \{C\}$.
$\mathcal{E}$	A finite dataset.
$P(\mathcal{E})$	The empirical probability distribution induced from $\mathcal{E}$.
$\boldsymbol{a}[\boldsymbol{X}]$	The vector of values of $\boldsymbol{a} \in \mathcal{E}$ with respect to $\boldsymbol{X}$.
$\boldsymbol{a}[\mathrm{C}]$	The class label of $\boldsymbol{a} \in \mathcal{E}$.
$\mathcal{E}_{\boldsymbol{X}=\boldsymbol{x}}$	$= \{\boldsymbol{a} \in \mathcal{E} \mid \boldsymbol{a}[\boldsymbol{X}] = \boldsymbol{x}\}$
$\mathcal{E}_{\boldsymbol{X}=\boldsymbol{x},\mathrm{C}=\xi}$	$= \{\boldsymbol{a} \mid \boldsymbol{a}[\boldsymbol{X}] = \boldsymbol{x}, \boldsymbol{a}[\mathrm{C}] = \xi\}$

2.1 Definition

The concept of a consistency measure was introduced to evaluate the distance of a given feature set from the *consistency state*. Using the notations introduced above, consistency of feature sets is formally defined as follows.

Consistency state. For $P \in \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C})$, $\boldsymbol{X} \subseteq \boldsymbol{\Omega}$ is said to be *consistent with respect to* P, when

$$P(\mathrm{C} = \xi \mid \boldsymbol{X} = \boldsymbol{x}) = 0, 1$$

holds for any value vector $\boldsymbol{x}$ to $\boldsymbol{X}$ and class label ξ.

A consistency measure determines a non-negative value $\mu(P, \boldsymbol{X})$ given a probability distribution $P \in \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C})$ and a feature subset $\boldsymbol{X} \in \mathfrak{P}_0(\boldsymbol{\Omega})$, and hence,

$$\mu : \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C}) \times \mathfrak{P}_0(\boldsymbol{\Omega}) \longrightarrow [0, \infty).$$

Here, we focus on two crucial properties of the distance. One, is that the distance between two points is 0, if, and only if, the points are identical. The other is that, the closer two points are to each other, the smaller is the distance value. The *determinacy* condition introduced in this section corresponds to this first property of the distance, while the *monotonicity* condition corresponds to the second property.

The determinacy condition can be formulated in a straightforward manner as follows.

Determinacy. $\mu(P, \boldsymbol{X}) = 0$ holds, if, and only if, $\boldsymbol{X}$ is consistent with respect to P.

By contrast, it is not straightforward to provide the similar expression for the second property of the distance in the case of consistency measures. This is because it is impossible to reasonably determine which of feature subsets $\boldsymbol{X}$ or $\boldsymbol{Y}$ is closer to the consistency state. Assume that P is the empirical probability distribution derived from a finite dataset $\mathcal{E}$. Then, the *inconsistency rate* measure of $\boldsymbol{X}$ ([Liu *et al.*, 1998]), is defined as follows.

$$\mu_{\mathsf{icr}}(P, \boldsymbol{X}) = \sum_{\boldsymbol{x}} \left(P(\boldsymbol{X} = \boldsymbol{x}) - \max_{\xi} P(\boldsymbol{X} = \boldsymbol{x}, \mathrm{C} = \xi) \right)$$

If $\mu_{\mathsf{icr}}(P, \boldsymbol{X}) = 0.5$ and $\mu_{\mathsf{icr}}(P, \boldsymbol{Y}) = 0.01$, $\boldsymbol{X}$ and $\boldsymbol{Y}$ become consistent when we remove 50% and 1% examples in $\mathcal{E}$ appropriately. This interpretation reasonably lead us to conclude $\boldsymbol{Y}$ is closer to the consistency state than $\boldsymbol{X}$. Yet, without contradiction, we can also assume that $\boldsymbol{X}$ becomes consistent if we add one more feature to $\boldsymbol{X}$, while $\boldsymbol{Y}$ requires many features to be added to become consistent. This would result in a contradiction. Thus, in order to determine which feature subset in $\boldsymbol{X}$ and $\boldsymbol{Y}$ is closer, we require an interpretation model, which can vary according to specific algorithms. As an exception when two feature subsets are in an inclusion relation, we don't need such a model . Thus, the *monotonicity* condition defined below should hold, regardless of the interpretation model.

Monotonicity. If $\boldsymbol{X} \supseteq \boldsymbol{Y}$, $\mu(P, \boldsymbol{X}) \leq \mu(P, \boldsymbol{Y})$ holds for an arbitrary probability distribution $P \in \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C})$.

Although the monotonicity condition was known in the literature, the literature emphasizes the importance of the condition for algorithm efficiency ([Pawlak, 1991; Liu *et al.*, 1998; Arauzo-Azofra *et al.*, 2008; Shin and Xu, 2009]). By contrast, in this paper, we claim that the conditions of determinacy and monotonicity should be the axioms that define a consistency measure.

An example of a consistency measure according to our definition is the *binary measure* defined below.

$$\mu_{\mathsf{bin}}(P, \boldsymbol{X}) = \begin{cases} 0, & \text{if } \boldsymbol{X} \text{ is consistent with respect to } P, \\ 1, & \text{otherwise.} \end{cases}$$

2.2 Reclassifying feature selection measures

For our analysis, we took advantage of the excellent survey by Molina et al., that lists six types of measures: *Dependence*, *Information/Uncertainty*, *Error probability*, *Divergence*, *Interclass distance* and *Consistency*. Ben-Bassat shows that the first four categories overlap ([Ben-Bassat, 1982]).

We ignored the types *Dependence* and *Information/Uncertainty* from the scope of our analysis, since these evaluate the relation between individual features and class labels. As described earlier, consistency measures evaluate relevance of feature subsets instead of individual features. We investigated 19 measures, and found that 14 were consistency measures. Surprisingly, only 5 of these 14 were described as such in the Molina survey.

Error probability. The error probability defined bellow is also known as *Bayesian risk*.

$$\mu_{\mathsf{ep}}(P, \boldsymbol{X}) = 1 - \mathbf{E}(\max_{y} P\left(\mathrm{C} = y \mid \boldsymbol{X} = \boldsymbol{x}\right))$$

Bayesian risk is equivalent to the inconsistency rate measure ([Liu *et al.*, 1998]) when applied to the empirical probability distributions derived from finite datasets, and Liu et al. proved that the inconsistency rate satisfies the conditions of determinacy and monotonicity.

Divergence. When class labels are binary, a divergence (distance) between two probability distributions of $p_{\boldsymbol{X}}(\boldsymbol{x}) = P(\boldsymbol{X} = \boldsymbol{x} \mid C = 0)$ and $q_{\boldsymbol{X}}(\boldsymbol{x}) = P(\boldsymbol{X} = \boldsymbol{x} \mid C = 1)$ can be used as a measure for feature selection. The divergences of Chernoff, Bhattacharyya, Kullback-Leibler[1], Kolmogorov, Matusita and Patrick-Fisher are commonly derived from the formula

$$J_{\boldsymbol{X}} = \sum_{\boldsymbol{x}} f(p_{\boldsymbol{X}}(\boldsymbol{x}), q_{\boldsymbol{X}}(\boldsymbol{x})). \tag{1}$$

The definitions for f are given in Table 1.

Table 1: Instances of divergence

	$f(x, y) =$	divergence $=$
Chernoff	$x^s y^{1-s}$	$-\ln J$
Bhattacharyya	$\sqrt{xy}$	$-\ln J$
Kullback-Leibler	$(x-y)\ln\frac{x}{y}$	J
Kolmogorov	$\mid x - y \mid$	J
Matusita	$(\sqrt{x}-\sqrt{y})^2$	$\sqrt{J}$
Patrick-Fisher	$(x-y)^2$	$\sqrt{J}$

$J_{\boldsymbol{X}}$ can be converted to measures for feature selection as described in Table 2, and the resulting measures satisfy the determinacy and monotonicity conditions for the cases of Chernoff, Bhattacharyya, Kolmogorov and Matusita. In examining the monotonicity condition, Lemma 1 plays a crucial role.

Lemma 1. *If $f(x+y, z+u) \leq f(x,z) + f(y,u)$ holds for arbitrary $x \geq 0, y \geq 0, z \geq 0$ and $u \geq 0$, $J_{\boldsymbol{X}} \leq J_{\boldsymbol{Y}}$ holds for $\boldsymbol{X} \subseteqq \boldsymbol{Y}$. Also, if $f(x+y, z+u) \geq f(x,z) + f(y,u)$ holds for arbitrary $x \geq 0, y \geq 0, z \geq 0$ and $u \geq 0$, $J_{\boldsymbol{X}} \geq J_{\boldsymbol{Y}}$ holds for $\boldsymbol{X} \subseteqq \boldsymbol{Y}$.*

Interclass distance. Interclass distance determines a distance between two classes as an average of the distances between all of the examples that belong to the classes. To illustrate, for two examples $\boldsymbol{a}, \boldsymbol{b}$ in a dataset $\mathcal{E}$, we let $d_X(\boldsymbol{a}, \boldsymbol{b}) = \sum_{X \in \boldsymbol{X}} \left|\boldsymbol{a}[X] - \boldsymbol{b}[X]\right|$, which is the L1-norm of the vector $\boldsymbol{a}[\boldsymbol{X}] - \boldsymbol{b}[\boldsymbol{X}]$ in the space $\mathbb{R}^{\boldsymbol{X}}$. Then, a distance between classes ξ and η is defined by

$$D_{\boldsymbol{X}}(\xi, \eta) = \frac{1}{\mid \mathcal{E}_{\mathrm{C}=\xi} \mid\mid \mathcal{E}_{\mathrm{C}=\eta} \mid} \sum_{\boldsymbol{a} \in \mathcal{E}_{C=\xi}} \sum_{\boldsymbol{b} \in \mathcal{E}_{C=\eta}} d_{\boldsymbol{X}}(\boldsymbol{a}, \boldsymbol{b}).$$

[1]Since there is no essential difference between the class labels 0 and 1, we examine a symmetric version of Kullback-Leibler divergence.

Table 2: Determinacy and monotonicity for divergences

Divergence	Symbol	n	nn	D	M
Chernoff	$\mu_{\mathsf{nCh}}/\mu_{\mathsf{nnCh}}$	J	J	Y	Y
Bhattacharyya	$\mu_{\mathsf{nBh}}/\mu_{\mathsf{nnBh}}$	J	J	Y	Y
Kullback-Leibler	$\mu_{\mathsf{nKL}}/\mu_{\mathsf{nnKL}}$	e^{-J}	e^{-J}	N	Y
Kolmogorov	$\mu_{\mathsf{nKol}}/\mu_{\mathsf{nnKol}}$	$1-\frac{J}{2}$	$1-J$	Y	Y
Matusita	$\mu_{\mathsf{nMa}}/\mu_{\mathsf{nnMa}}$	$1-\frac{J}{2}$	$1-J$	Y	Y
Patrick-Fisher	$\mu_{\mathsf{nPF}}/\mu_{\mathsf{nnPF}}$	$1-\frac{J}{A}$	$1-\frac{J}{B}$	Y	N

The symbols "n" and "nn" indicate "normalized" and "non-normalized", respectively. For a normalized divergence, $p_{\boldsymbol{X}}(\boldsymbol{x})$ and $q_{\boldsymbol{X}}(\boldsymbol{x})$ are determined by $p_{\boldsymbol{X}}(\boldsymbol{x}) = P(\boldsymbol{X} = \boldsymbol{x} \mid C = 0)$ and $q_{\boldsymbol{X}}(\boldsymbol{x}) = P(\boldsymbol{X} = \boldsymbol{x} \mid C = 1)$, whereas they are determined by $p_{\boldsymbol{X}}(\boldsymbol{x}) = P(\boldsymbol{X} = \boldsymbol{x}, C = 0)$ and $q_{\boldsymbol{X}}(\boldsymbol{x}) = P(\boldsymbol{X} = \boldsymbol{x}, C = 1)$ for a non-normalized divergence. Also, the symbols "D" and "M" mean "Determinacy" and "Monotonicity", and the values A and B used in the row of Patrick-Fisher are defined as follows.

$$A = \sum_{\boldsymbol{x}} P(\boldsymbol{X} = \boldsymbol{x} \mid C = 0)^2 + \sum_{\boldsymbol{x}} P(\boldsymbol{X} = \boldsymbol{x} \mid C = 1)^2$$

$$B = \sum_{\boldsymbol{x}} P(\boldsymbol{X} = \boldsymbol{x}, C = 0)^2 + \sum_{\boldsymbol{x}} P(\boldsymbol{X} = \boldsymbol{x}, C = 1)^2$$

On one hand, by applying a positive decreasing function to $\sum_{\xi \neq \eta} D_{\boldsymbol{X}}(\xi, \eta)$ for example, we can derive a measure from the interclass distance so that it satisfies the monotonicity condition, since, if $\boldsymbol{X} \subseteqq \boldsymbol{Y}$, $d_{\boldsymbol{X}}(\boldsymbol{a}, \boldsymbol{b}) \leq d_{\boldsymbol{Y}}(\boldsymbol{a}, \boldsymbol{b})$ holds. On the other hand, even for such a measure, the determinacy condition does not hold. When $\boldsymbol{X}$ is consistent, any pair $\boldsymbol{a} \in \mathcal{E}_{\mathrm{C}=\xi}$ and $\boldsymbol{b} \in \mathcal{E}_{\mathrm{C}=\eta}$ satisfies $\boldsymbol{a}[\boldsymbol{X}] \neq \boldsymbol{b}[\boldsymbol{X}]$, and hence, $d_{\boldsymbol{X}}(\boldsymbol{a}, \boldsymbol{b}) \neq 0$. Nevertheless, this does not mean that $D_{\boldsymbol{X}}(\xi, \eta) > D_{\boldsymbol{Y}}(\xi, \eta)$ would always hold for consistent $\boldsymbol{X}$ and inconsistent $\boldsymbol{Y}$.

Other consistency measures. In [Molina *et al.*, 2002], the inconsistency rate μ_{icr} ([Liu *et al.*, 1998]) is discussed. Other consistency measures in the literature are: rough set consistency measure μ_{rs} ([Pawlak, 1991]), inconsistent example pair measure μ_{icp} ([Arauzo-Azofra *et al.*, 2008]), conditional entropy μ_{ce} and complementary symmetric uncertainty μ_{csu} ([Shin and Xu, 2009]).

3 Comparing consistency measures

A method to compare consistency measures would provide a basis to select a particular measure for a specific task. For example, suppose we wish to select a measure with higher predictive accuracy. How can we go about doing this? Comparing measures is difficult, since one feature selection algorithm cannot always outperform others for all classification algorithms and all datasets. One approach that would provide an approximate answer would be: test the feature selection algorithm in combination with a wide variety of classification algorithms and datasets – but this is time consuming and very impractical. To the best of our knowledge, the literature does not guide us on how we could compare consistency measures.

To fill in this gap in the literature, we introduce the terms *sensitivity* and *quasi-linear compatibility order* to serve as a theoretical basis to compare measures. Now, a measure with a threshold, classifies its evaluation targets into two groups,

those with values smaller than the threshold and, the rest. We are interested in the former group. A measure that can identify any group of targets that another measure identifies is termed to have a higher sensitivity, and can therefore be considered to be better. We do the comparison in terms of the relative sensitivity of the measures. It is reasonable for example, to expect that measures with a higher sensitivity, when implemented in feature selection algorithms, will result in better accuracy performance for a majority of classification algorithms and a majority of datasets.

3.1 Quasi-linear compatibility order

Consider the simple case when two consistency measures μ_1 and μ_2 satisfy $\mu_2 = \alpha\mu_1$. In this case, we can intuitively conclude that they are equivalent. This intuition is justified by verifying $\forall(\delta > 0)[\Gamma(\delta : \mu_1) = \Gamma(\alpha\delta : \mu_2)]$, where

$$\Gamma(\delta : \mu) = \{(P, \boldsymbol{X}) \in \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C}) \times \mathfrak{P}_0(\boldsymbol{\Omega}) \mid \mu(P, \boldsymbol{X}) \leq \delta\}$$

represents the set of $(P, \boldsymbol{X})$ that are identified by a measure μ with a threshold δ. Although this reasoning gives us some direction, it is still too restrictive in two senses. First, it doesn't give us any idea about which of two measures is more or less sensitive. Second, since a consistency-based algorithm deals with situations where feature subsets are close to the consistency state, we are only interested in comparing measures in a neighborhood of $\delta = 0$.

Therefore, we extend the aforementioned idea of comparison, and introduce *quasi-linear compatibility order* as follows.

Definition 1. *Let μ_1 and μ_2 be consistency measures. We say that μ_1 is* no less sensitive than *μ_2 with respect to quasi-linear compatibility, if, and only if,*

$$\exists(\alpha > 0)\exists(\delta_0 > 0)\forall(\delta \leq \delta_0)[\Gamma(\delta : \mu_2) \subseteqq \Gamma(\alpha\delta : \mu_1)]$$

holds. Moreover, we denote the relation by $\mu_1 \succeq \mu_2$.

This definition implies all of $(P, \boldsymbol{X})$ identified by a less sensitive measure with a sufficiently small threshold δ should be also identified by a more sensitive measure with a threshold linearly converted from δ.

Definition 2. *When both of $\mu_1 \preceq \mu_2$ and $\mu_1 \succeq \mu_2$ simultaneously hold, we denote the relation by $\mu_1 \sim \mu_2$.*

The relation $\sim$ is an equivalence relation, and $\succeq$ becomes a *partial* order over the quotient space by the relation $\sim$. We call this order *quasi-linear compatibility order*. When $\mu_1 \preceq \mu_2$ and $\mu_1 \not\sim \mu_2$, we denote this by $\mu_1 \prec \mu_2$.

Theorem 1 provides us with a useful method to compare given inconsistency measures.

Theorem 1. *Assume μ_1 and μ_2 are both consistency measures and 0 is not an isolated point of the range of μ_1*[2]*. When we define $S(x)$ and $I(x)$ by*

$$S(x) = \sup\{\mu_2(P, \boldsymbol{X}) \mid (P, \boldsymbol{X}) \in \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C}) \times \mathfrak{P}_0(\boldsymbol{\Omega}), \mu_1(P, \boldsymbol{X}) = x\} \text{ and}$$

$$I(x) = \inf\{\mu_2(P, \boldsymbol{X}) \mid (P, \boldsymbol{X}) \in \mathcal{P}(\boldsymbol{\Omega}, \mathrm{C}) \times \mathfrak{P}_0(\boldsymbol{\Omega}), \mu_1(P, \boldsymbol{X}) = x\},$$

we have the following properties.

1. *$\mu_1 \preceq \mu_2$ holds, if, and only if, $\limsup_{x\to 0} \frac{S(x)}{x} < \infty$ holds.*
2. *$\mu_1 \succeq \mu_2$ holds, if, and only if, the following two relations hold simultaneously.*

$$\limsup_{y\to 0}\{\mu_1(P, \boldsymbol{X}) \mid \mu_2(P, \boldsymbol{X}) \leq y\} = 0 \tag{2}$$

$$\liminf_{x\to 0} \frac{I(x)}{x} > 0 \tag{3}$$

3.2 Comparing sensitivity of consistency measures

We investigated the order in sensitivity among the consistency measures identified in 2.2. For example, $I_{\mu_{\mathsf{ce}}}(\mu_{\mathsf{icr}}) \geq -\log(1 - \mu_{\mathsf{icr}})$ and $S_{\mu_{\mathsf{ce}}}(\mu_{\mathsf{icr}}) = -(1 - \mu_{\mathsf{icr}})\log(1 - \mu_{\mathsf{icr}}) - \mu_{\mathsf{icr}} \log \mu_{\mathsf{icr}}$ hold, and hence, $\mu_{\mathsf{icr}} \succ \mu_{\mathsf{ce}}$ follows from

$$\liminf_{\mu_{\mathsf{icr}}\to 0} \frac{I_{\mu_{\mathsf{ce}}}(\mu_{\mathsf{icr}})}{\mu_{\mathsf{icr}}} \geq 1 \quad \text{and} \quad \limsup_{\mu_{\mathsf{icr}}\to 0} \frac{S_{\mu_{\mathsf{ce}}}(\mu_{\mathsf{icr}})}{\mu_{\mathsf{icr}}} = \infty.$$

Similarly, we have the order in sensitivity of consistency measures depicted below.

$$\begin{array}{c} \mu_{\mathsf{icp}} \\ \curlyvee \\ \mu_{\mathsf{icr}} \sim \mu_{\mathsf{ep}} \sim \mu_{\mathsf{nnKol}} \\ \curlyvee \\ \{\mu_{\mathsf{nCh}}, \mu_{\mathsf{nnCh}}, \mu_{\mathsf{nBh}}, \mu_{\mathsf{nnBh}}, \mu_{\mathsf{nKol}}, \mu_{\mathsf{nMa}}, \mu_{\mathsf{nnMa}}, \mu_{\mathsf{rs}}, \mu_{\mathsf{ce}}, \mu_{\mathsf{csu}}\} \\ \curlyvee \\ \mu_{\mathsf{bin}} \end{array}$$

4 Our Algorithm – CWC

Since the number of feature subsets is an exponential function of the size of the entire feature set, selection of a search organization has a great impact on the time efficiency of the resulting algorithms. Here, [Zhao and Liu, 2007] made a significant contribution by showing that their algorithm with a linear search organization, INTERACT, can exhibit sufficiently good performance in accuracy. INTERACT evaluates only as many feature subsets as the involved features. Nevertheless, INTERACT failed to show sufficient improvement in time-efficiency. In fact, the authors of [Zhao and Liu, 2007] used only relatively small datasets for experiments with INTERACT.

This low-efficiency of INTERACT is because the inconsistency rate μ_{icr}, which is implemented in INTERACT, is a continuous measure, that is, it can take an arbitrary value in a certain interval, and hence, evaluation of the measure is computationally costly.

Our new algorithm, *CWC* (*Combination of Weakest Components*), attempts to combine the binary measure μ_{bin} and a linear search organization. CWC is a greedy backward elimination algorithm. We can expect some improvement in time-efficiency with CWC, since evaluation of μ_{bin} is much faster than any other continuous consistency measures.

The algorithm of CWC is described in Fig. 1. In the algorithm, the denoisation of $\mathcal{E}$ are based on the following rules.

Rule 1. Eliminate all minor examples.

Rule 2. Eliminate all inconsistent examples.

Rule 3. Eliminate all examples with occurrence 1.

[2] Among the consistency measures in 2.2, all but the binary measure μ_{bin} meet this condition.

Algorithm: CWC
INPUT: A feature set $\mathbf{\Omega}$, an example set $\mathcal{E}$,
OUTPUT: A minimal subset $\boldsymbol{X} \subseteq \mathbf{\Omega}$ such that $\mu_{\mathsf{bin}}(P(\mathcal{E}), \boldsymbol{X}) = 0$.
STEPS:
Let $\boldsymbol{X} = \mathbf{\Omega}$.
If $\mu_{\mathsf{bin}}(P(\mathcal{E}), \boldsymbol{X}) = 1$, remove presumable noise from $\mathcal{E}$.
For each $X \in \boldsymbol{X}$ from the first to the end.
If $\mu_{\mathsf{bin}}(P(\mathcal{E}), \boldsymbol{X} \setminus \{X\}) = 0$, let $\boldsymbol{X} = \boldsymbol{X} \setminus \{X\}$.
End For.

Figure 1: The algorithm of CWC

Two examples $\boldsymbol{a}$ and $\boldsymbol{b}$ are called *inconsistent*, when they have the same feature values ($\boldsymbol{a}[\mathbf{\Omega}] = \boldsymbol{b}[\mathbf{\Omega}]$), but different class labels ($\boldsymbol{a}[\mathrm{C}] \neq \boldsymbol{b}[\mathrm{C}]$). For inconsistent $\boldsymbol{a}$ and $\boldsymbol{b}$, if the number of cases of $\boldsymbol{a}$ is effectively smaller than that of $\boldsymbol{b}$, $\boldsymbol{a}$ is called *minor*. In this paper, we test two different denoising algorithms, *aggressive* and *moderate*. The aggressive algorithm executes all of Rule 1, 2 and 3 in that order, whereas the moderate algorithm does not employ Rule 3.

Also, when examining each individual feature, CWC does not assume any particular order of features. However, the linear search organization tends to discard features tested earlier with higher probability than those tested later ([Zhao and Liu, 2007]). Hence, the resulting accuracy will be usually better, when features are aligned in some appropriate order (*e.g.* the increment order of symmetric uncertainty).

As shown in 3.2, μ_{bin} is the least sensitive measure. Hence, it is reasonable to expect that the combination of the weakest measure and the weakest search organization would significantly sacrifice performance in accuracy in exchange for improvement in time efficiency. But as we see in Section 4.1, CWC shows vast improvement in time efficiency and performance in accuracy comparable with LCC ([Shin and Xu, 2009]). LCC provides improvement in accuracy performance by addressing a theoretical flaw in INTERACT.

Since μ_{icr} is more sensitive than μ_{bin}, LCC may possibly show better accuracy when an optimal threshold is given. However, it is impractical to look for an optimal threshold on a trial-and-error basis, since LCC is as slow as INTERACT.

4.1 Experimental results

Settings. We compared CWC with other feature selection algorithms with respect to runtime, accuracy and number of features selected. The main goal of CWC is to improve runtime efficiency compared to other consistency-based algorithms. Accuracy and number of features selected are two important measures in the literature to evaluate feature selection algorithms. In addition, since CWC includes a denoising process, we investigate its reluctance to noise in datasets.

We selected INTERACT and LCC as the consistency-based algorithms for comparison. Other known consistency-based algorithms cannot be used for our experiments because of their impractical low time efficiency. In addition, we compared CWC with widely known non-consistency-based algorithms, FCBF ([Yu and Liu, 2003]), ReliefF ([Kononenko, 1994]) and CFS ([Hall, 2000]).

The datasets for the experiments on runtime, accuracy and feature numbers are taken from NIPS 2003 Feature Selection Challenge and WCCI 2006 Performance Prediction Challenge. Thus we have large-scale datasets of various types (*e.g.* dense vs. sparse) (Table 4). For the experiment on noise tolerance, we first generated synthetic datasets that include the unique minimum feature subset that is consistent, and then added noise by flipping class labels.

Consistency-based algorithms handle only discrete features. So we discretized datasets with continuous features before running CWC, LCC and INTERACT, on them. For the discretization, we employ a very simple algorithm that divides the range of features equally into five class labels. In addition, we align features in the order of symmetric uncertainty before CWC, LCC and INTERACT run on the datasets.

In the experiment with CWC on runtime, accuracy and number of features, we show only the results with respect to the moderate denoising, since the moderate denoising turns out to pass over less relevant features than the other, when noise exists in the datasets. For noise tolerance performance, we compared aggressive versus moderate denoising.

To make the comparison fair, we first developed a platform for consistency-based feature selection in C++, and then implemented CWC, INTERACT and LCC on the platform. For the legacy feature selection algorithms, we used an implementation on top of Weka ([Witten and Frank, 2005]).

Runtime. For legacy algorithms (Table 5), we linearly converted the experiment observations by a factor of 5.1 assuming the algorithms run directly on top of the operating systems. We did this because our C++ implementation of INTERACT is 5.1 times faster than its Weka implementation.

CWC performed better than all but FCBF. In particular, its superiority over INTERACT and LCC is remarkable. The total runtime of CWC (646.8 sec) is only 2.2% that of INTERACT and LCC (29254.4 sec).

Accuracy. Table 6 shows the values of AUC of ROC curve when we apply Naive Bayes, J48 and libSVM classifying algorithms to the results of the feature selection algorithms tested. For each combination of feature selection algorithm, dataset and classifying algorithm, Table 6 exhibits the average of the five AUC values obtained through a five-fold cross-validation. The row titled "all" shows the corresponding values for the raw datasets without performing feature selection first. We used the default parameters chosen by Weka for these classifying algorithms.

To determine the threshold δ for INTERACT and LCC, we selected $\delta = 0.01$, which is the positive threshold that showed the best accuracy on average, based on multiple trials. Table 3 summarizes Table 6. The second column of Table 3 shows the number of times a feature selection algorithm performed the best in Table 6. The other columns show averages of AUC/ROC values. It is significant that CWC holds first place in all columns of Table 3.

Number of features selected. The ratios of the features that the algorithms selected to the total number of features are sufficiently small (Table 7). The difference in the ratios is not significant except in the case of INTERACT. However, the very low percentage (0.11%) for INTERACT is at the cost of its low performance in accuracy.

Table 4: Attributes of the datasets used

	ada	madelon	arcene	sylva	gina	ad	dexter	hiva	nova	gisette	dorothea
Number of Features	48	500	10000	216	970	1558	20000	1617	16969	5000	100000
Number of Examples	4147	2000	100	13086	3153	3279	300	3845	1754	6000	800

Table 5: Runtime in seconds using PC with Intel Core Duo 3.16 GHz and 4 GB memory

	ada	madelon	arcene	sylva	gina	ad	dexter	hiva	nova	gisette	dorothea
cwc	*0.031*	*0.062*	*0.063*	*0.75*	*0.484*	9.70	155.7	3.89	402.3	40.6	*33.2*
interact/lcc	0.217	2.64	36.6	5.45	17.2	29.3	606.3	50.5	2579.8	753.6	25172.8‡
fcbf	0.194†	0.582†	0.582†	1.95†	1.94†	5.24†	*2.33*†	3.11†	*15.5*†	*25.2*†	–∗
cfs	17.5†	48.6†	3.30†	11.9†	221.4†	372.3†	35.5†	932.4†	–∗	3977.6†	–∗
relief	0.388†	0.777†	421.5†	1.06†	13.6†	*4.08*†	1591.8†	*3.10*†	502.6†	351.3†	–∗

† The displayed values are estimated assuming the algorithms run directly on top of operating systems.
‡ Since INTERACT failed to process Dorothea when features are ordered, we run it without ordering features.
∗ The algorithms could not complete the job probably because of lack of memory.

Table 6: AUC of ROC curve ($\delta = 0.01$ for INTERACT and LCC)

NaiveBayes

	ada	madelon	arcene	sylva	gina	ad	dexter	hiva	nova	gisette	dorothea	Ave.
cwc	*0.891*	*0.684*	0.94	0.998	*0.912*	0.941	0.976	*0.796*	*0.938*	0.964	*0.972*	*0.900*
lcc	*0.891*	0.674	0.920	0.998	0.900	*0.943*	0.971	0.795	0.933	0.950	–	0.898
interact	0.883	0.681	0.920	0.989	0.853	0.883	0.860	0.491	0.760	0.901	–	0.829
fcbf	0.888	0.680	0.994	0.992	0.906	0.921	0.977	0.741	0.916	0.971	–	0.897
cfs	0.878	0.675	*0.996*	0.998	0.910	0.942	*0.992*	0.741	–	*0.969*	–	*0.900*
relief	0.869	0.679	0.889	*0.999*	0.887	0.921	0.927	0.624	0.715	0.939	–	0.859
all	0.892	0.628	0.707	0.997	–	0.943	0.910	0.725	0.930	0.943	0.841	0.843

J48

	ada	madelon	arcene	sylva	gina	ad	dexter	hiva	nova	gisette	dorothea	Ave.
cwc	0.862	0.777	*0.934*	0.991	0.888	0.910	0.875	*0.669*	*0.827*	*0.964*	*0.701*	*0.874*
lcc	0.857	0.774	0.872	0.990	0.887	0.906	0.849	0.595	0.813	0.949	–	0.850
interact	0.819	*0.798*	0.885	0.969	0.837	0.848	0.805	0.491	0.754	0.900	–	0.817
fcbf	*0.870*	0.636	0.927	0.937	0.882	0.891	0.896	0.556	0.775	0.946	–	0.838
cfs	*0.870*	0.797	0.841	*0.992*	*0.901*	*0.923*	*0.923*	0.540	–	0.949	–	0.860
relief	0.821	0.800	0.794	*0.992*	0.859	0.907	0.869	0.491	0.634	0.947	–	0.831
all	0.852	0.701	0.709	0.958	0.855	0.915	0.783	0.588	0.824	0.924	0.683	0.809

LibSVM

	ada	madelon	arcene	sylva	gina	ad	dexter	hiva	nova	gisette	dorothea	Ave.
cwc	*0.756*	0.820	0.892	0.982	0.907	0.859	0.887	0.500	*0.786*	0.962	*0.786*	*0.841*
lcc	0.746	*0.828*	0.894	0.957	0.873	*0.889*	0.883	*0.500*	*0.786*	0.941	–	0.830
interact	0.723	0.806	0.924	0.923	0.812	0.847	0.833	0.500	0.765	0.878	–	0.805
fcbf	0.734	0.618	0.908	0.902	0.852	0.776	0.883	*0.522*	0.760	0.939	–	0.793
cfs	0.730	0.786	*0.926*	0.968	*0.910*	0.847	*0.913*	0.503	–	*0.969*	–	0.839
relief	0.731	0.816	0.878	*0.984*	0.708	0.851	0.827	0.500	0.596	0.964	–	0.807
all	0.506	0.500	0.500	0.500	0.500	0.755	0.500	0.500	0.500	0.500	0.775	0.529

The values displayed in italic font represent the best performing feature selection algorithms in each column.

Table 7: Number of features selected

	ada	madelon	arcene	sylva	gina	ad	dexter	hiva	nova	gisette	dorothea	Ave.	% of total†
cwc	31	14	5	19	23	35	31	48	87	22	28	25.3	0.57
lcc	31	13	6	2	25	4	20	9	55	21	–	14.6	0.33
interact	5	10	4	2	6	3	9	0	55	21	–	4.8	0.11
fcbf	9	5	17	11	16	50	62	12	47	33	–	23.9	0.54
cfs	9	8	45	13	55	26	58	13	–	80	–	34.1	0.77
relief	10	8	53	13	16	50	49	13	47	79	–	32.3	0.73

† The percentage of number of features selected to the total number of features.

Table 3: Rank of algorithms in accuracy

Algorithm	# of Top	AUC (Av.)	NB	J48	SVM
cwc	*14*	*0.872*	*0.900*	*0.874*	*0.841*
cfs	12	0.866	*0.900*	0.860	0.839
lcc	5	0.859	0.898	0.849	0.830
fcbf	2	0.842	0.897	0.838	0.793
relief	3	0.832	0.859	0.831	0.807
interact	1	0.817	0.829	0.817	0.805

Table 8: Noise tolerance

Noise Rate	0%		5%		10%	
	Prec.	Rec.	Prec.	Rec.	Prec.	Rec.
cwc (aggressive)	1.00	0.90	1.00	0.89	1.00	0.88
cwc (moderate)	1.00	1.00	0.39	0.98	0.35	0.96
lcc ($\delta = 1.0$)	1.00	0.80	0.39	0.98	0.35	0.95
interact ($\delta = 1.0$)	1.00	0.50	1.00	0.29	0.79	0.33

Noise tolerance. We ran an experiment to investigate how the aggressive and moderate denoising algorithms would result when noise exists in datasets. For this experiment, we generated 10 different synthetic datasets, each of which included 10 relevant features, 81 irrelevant features and 500 examples, and then added 5% and 10% noise. Since the 10 relevant features form the unique minimum consistent feature subset, we can evaluate precision and recall directly from the outputs of the feature selection algorithms.

Table 8 shows the averages of precision and recall scores over the 10 datasets tested. The data for LCC and INTERACT are just informative. The precision of CWC with the aggressive denoising algorithm is not affected by noise at all. In contrast, in terms of recall, CWC with the moderate denoising algorithm performs better. We can explain these results as follows. The aggressive algorithm eliminates more *noise*, and therefore, its precision scores are better. Also, since it eliminates more *effective* examples that support the right answer, its recall scores suffer.

We think that, if the numbers of features selected are small enough, reluctance to pass over relevant features is more important for the classification purpose. Therefore, in the experiments for accuracy, we presented only the results for the moderate denoising. In fact, we observed that the results for the moderate denoising were mostly better than those for the other, and the numbers of features selected were always small enough (Table 7).

5 Discussion

At first glance, the experimental results seem to contradict the property shown in 3.2 namely, that the binary measure is least sensitive. This contradiction is only apparent since it is likely that LCC and INTERACT outperform CWC when appropriate thresholds are chosen. For example, we can prove that, if the entire feature set of a given dataset is consistent from the beginning, the output of CWC is identical to those of LCC and INTERACT when the threshold is zero. This implies that LCC and INTERACT may possibly outperform CWC with positive thresholds in such a case.

Note CWC performs good enough in terms of accuracy, and it is much faster. For LCC or INTERACT, we should question whether the resulting improvement in accuracy, (which is likely to be relatively small) is worth the investment in time required to find appropriate thresholds on a trial-and-error basis. This time investment can be huge, since even on a single run LCC and INTERACT are slow.

Taking advantage of the high efficiency of CWC, we could improve performance in accuracy further. For example, we can iteratively run CWC, changing inputs (*e.g.* order of features) under an appropriate strategy.

References

[Arauzo-Azofra *et al.*, 2008] A. Arauzo-Azofra, J. M. Benitez, and J. L Castro. Consistency measures for feature selection. *Journal of Intelligent Information Systems*, 30(3):273–292, November 2008.

[Ben-Bassat, 1982] M. Ben-Bassat. Use of distance measures, information measures and error bounds in feature evaluation. *Handbook of Statistics, North-Holland Publishing Company*, 2:773 – 791, 1982.

[Hall, 2000] M. A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Proceedings of International Conference of machine Learning*, 2000.

[Kononenko, 1994] I. Kononenko. Estimating attributes: Analysis and extension of RELIEF. In *Proceedings of European Conference on Machine Learning*, 1994.

[Liu *et al.*, 1998] H. Liu, H. Motoda, and M. Dash. A monotonic measure for optimal feature selection. In *Proceedings of European Conference on Machine Learning*, 1998.

[Molina *et al.*, 2002] L. Molina, L. Belanche, and A. Nebot. Feature selection algorithms: A survey and experimental evaluation. In *Proceedings of IEEE International Conference on Data Mining*, pages 306–313, 2002.

[Pawlak, 1991] Z. Pawlak. *Rough Sets, Theoretical aspects of reasoning about data*. Kluwer Academic Publishers, 1991.

[Shin and Xu, 2009] K. Shin and X.M. Xu. Consistency-based feature selection. In *13th International Conferece on Knowledge-Based and Intelligent Information & Engineering System*, 2009.

[Witten and Frank, 2005] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Mogan Kaufmann Publishers, 2005.

[Yu and Liu, 2003] L. Yu and H. Liu. Feature selection for high-dimensional data: a fast correlation-based filter solution. In *Proceedings of International Conference of Machine Learning*, 2003.

[Zhao and Liu, 2007] Z. Zhao and H. Liu. Searching for interacting features. In *Proceedings of International Joint Conference on Artificial Intelligence*, pages 1156 – 1161, 2007.