ILA-3: An Inductive Learning Algorithm with a New Feature Selection Approach

SALEH M. ABU-SOUD†1 and SUFYAN ALMAJALI††
Department of Software Engineering
Princess Sumaya University for Technology
P.O. Box (1438) Amman 11941
JORDAN
{†abu-soud@psut.edu.jo, ††s.almajali@psut.edu.jo}

Abstract: - ILA is an inductive learning algorithm proved itself as a powerful algorithm in inductive learning community. It generates the less number of simplest rules compared with other similar algorithms with 100% accuracy; i.e. the generated rules cover all examples in the dataset. But, it becomes inefficient for large datasets. In this paper, a new generation of ILA, called ILA-3, has been developed and tailored with a new feature selection algorithm. This approach takes into consideration the way ILA works and excludes the most irrelevant features from the dataset under consideration while ILA is running, which yields to smaller datasets. Experiments show that significant efficiency improvements (that reached 30% on average) have been gained with ILA-3 over the original ILA with keeping accuracy with acceptable levels.

Key-words: ILA, ILA-3, Feature selection, Inductive learning, Irrelevant features.

1 Introduction

Machine learning algorithms automatically extract knowledge from example datasets, to be used for future predictions. The input datasets are usually composed of a pattern of examples of the concept under consideration. Each example is described by a vector of attributes or features along with a class attribute that denotes the category of the class.
Inductive learning algorithms are usually characterized by several factors as: 1) number of generated rules, 2) simplicity of generated rules, 3) induction power of the generated rules, i.e. the accuracy of identifying the unseen examples correctly, and finally 4) the efficiency of the algorithm i.e. the speed of the algorithm for generating rules [1].
Many factors affect the success of a machine learning algorithm on a given task. The representation and quality of the example data is the most important one. If there is much irrelevant and redundant information present or the data is noisy and unreliable, then knowledge discovery during the training phase is more difficult. Feature subset selection is the process of identifying and removing as much of the irrelevant and redundant information as possible. This reduces the dimensionality of the data and allows learning algorithms to operate faster and more effectively. In some cases, accuracy on future classification can be improved; in others, the result is a more compact, easily interpreted representation of the target concept [2].

1.1 Background and Related Work

Feature selection is one of the important and frequently used techniques in data preprocessing for machine learning and data mining. It reduces the number of features, removes irrelevant, redundant, or noisy data, and brings the immediate effects for applications: speeding up a data mining algorithm, improving mining performance such as predictive accuracy and result comprehensibility [3].
The process of focusing on the most important features and excluding the irrelevant ones is of utmost importance to improve performance on some dependent measures such as learning speed, number

[1] The corresponding author

and simplicity (the generality) of rules produced by the learner, and of course, the classification power of these rules which represents the accuracy of the rules in classifying unseen examples.

Feature selection has been a rich field of research and development since the 1970s in machine learning [4], [5], [6], image retrieval [7], [8], statistical pattern recognition [9], [10], and data mining [11], [12], and widely applied to many fields such as text categorization [13], [14], customer relationship management [15], intrusion detection [16], [17].

Feature selection algorithms fall into three categories: the filter model [18], [19], [20], the wrapper model [21], [22], [23], [24], and the hybrid model [25], [26], [27]. The filter model selects features using a preprocessing step. The main drawback of this approach is that it ignores the effect of the selected features on the performance of the induction algorithm used [28]. One of the algorithms that is located in this category is the FOCUS algorithm [29], [30] which originally applied on noise-free binary domains. It checks all features and selects the minimal subset that is sufficient to determine the class value for all examples in the training set. The Relief algorithm [31], [32] is another example of this category. It is a randomized algorithm that runs also on binary classification problems, which attempts to find out all relevant attributes. It does that by assigning a relevance weight to each attribute to target class value. Relief-F [33] is a general case of Relief, which work on multiple classes.

One of the most interesting approaches in this category is the wrapper approach by [28], [35]. In this approach, the feature selection algorithm performs a forward search [best-first or Hill-climbing] in the space of possible parameters for a good subset using the induction algorithm itself as part of the evaluation function. This is done by running an inductive algorithm on the dataset and using the estimated accuracy of the resulting classifier as its metric. OBLIVION algorithm [36] is an example of this approach which combines the wrapper idea with the nearest-neighbor method, which assigns to new examples the class of the nearest case stored in cash memory during the learning process. The hybrid model attempts to take advantage of the two models by exploiting their different evaluation criteria in different search stages.

Feature selection algorithms are usually either stand-alone or tailored with the inductive algorithm. Stand-alone feature selection algorithms are usually called by induction algorithms as a subprogram, and can be used by any inductive learning algorithm as a preprocessing step prior to running the learning algorithm. Algorithms in this category can be used by any learning algorithm. Its use reduces the dataset significantly which leads to a significant improvements in the efficiency of the algorithm it is used with. One of the drawbacks of this type of algorithms is that it does not take the internal characteristics of the learning algorithm into consideration.

Tailored feature selection algorithms are usually embedded within an inductive learning algorithm itself, and taking advantage of the way the learning algorithm works. The wrapper model is an example of feature selection approaches of this category; it requires one predetermined learning algorithm and uses its performance as the evaluation criterion. It searches for features better suited to the learning algorithm aiming to improve learning performance [24], [34]. Quinlan’s ID3 [43] performs some sort of built-in feature selection.

In this paper, we announce a new version of our ILA inductive algorithm [1] in which it is empowered by new embedded feature selection capabilities, which in turn; as will be seen in the experiments, enhances its performance significantly. The new algorithm is called ILA-3.

We will not focus in this paper on the benefits and good characteristics of ILA since many researches did this. Instead, we will briefly mention it along with all its versions and discuss thoroughly its deficiencies with regarding to efficiency and slowness in the learning process. This is done in the next section followed by a discussion of the new algorithm and the feature selection approach used.

2 ILA Family

ILA had been originally used in 1997 as a framework for connecting decision support systems with expert systems [37], and then announced as an inductive learning algorithm in 1998 [1]. In 1999, a slight modification of ILA had been done as a new algorithm called DCL [38]. DCL produces rules with disjuncts (OR-operator in the LHS of the rules) which enhances the classification of unseen examples significantly. In this year also, a major version of ILA; called ILA-2 [39], had been published in which rules are generated with uncertainty, since ILA produces rules with 100% certainty. A parallel version of ILA had been built in 2000 [40] to let ILA run on parallel machines. In 2009, a Ph.D. study was conducted on ILA to make it suitable to be run on distributed databases. The results of this study can be shown in [41].

ILA had been successfully applied on different research areas such as intrusion detection [17] and text-to-speech synthesis [42].

2.1 Description and Problems of ILA

There is no need to re-discuss ILA here in details again. All details can be found in [1]. Instead, some important aspects of ILA will be mentioned, that help in understanding the justifications for the new version.

ILA generates the minimum number of simplest rules, and has the highest classification power among similar algorithms in the field. ILA based on computing all the combinations of the attributes of the dataset under consideration. It starts with the combination that includes 1 attribute, and extracts its associated rules and then moves to the combinations with 2 attributes and so on until it reaches the combination that includes all attributes. This means that it generates the most general rules first. So, for a dataset that contains 3 attributes attr1, attr2, and attr3, the set of resulted combinations are as shown in Table 1. As noted in Table 1, a dataset with 3 attributes, ILA must generate rules for 7 combinations: 3 combinations with 1 attribute, 3 combinations with 2 attributes, and one combination with 3 attributes.

Table 1. The Combinations for a dataset with 3 attributes

# of attributes	Combinations
1	(attr1), (attr2), and (attr3)
2	(attr1,attr2), (attr1,attr3), and (attr2,attr3)
3	(attr1,attr2,attr3)

Where number of combinations for a dataset that contains n attributes can be computed by equation 1, and number of combinations with k attributes in a dataset with n attributes is computed by equation 2.

Total number of combinations in a dataset with n attributes = 2n-1 ……………………..…..…………. (1)

Number of combinations with k attributes = n!/(k!*(n-k)!) ……………………...……….……. (2)

Table 2. Number of combinations for dataset with different number of attributes

# of attributes	# of combinations
1	1
2	3
5	31
6	63
10	1023
20	1048575
50	$11X10^{14}$
100	$12X10^{29}$
.	
N	2^n-1

The processing of combinations is the main problem of ILA, which in turn affects negatively the efficiency of the algorithm significantly, especially for large datasets. Table 2 shows number of combinations for datasets with different number of attributes. It is obvious from the table that as number of attributes gets high, number of combinations increases dramatically. Number of combinations is almost doubled for each extra one attribute.

To illustrate the behavior of ILA on large datasets, let’s consider the letter recognition dataset2. Table 3 shows the characteristics of this dataset.

An experiment has been conducted in which ILA has been run on this dataset for different number of attributes where they were selected randomly and data are manipulated in such a way that duplicates and contradictions were removed. The aim of this

[2] University of California Irvine Repository of Machine Learning Databases and Domain Theories via anonymous ftp to charlotte.ics.uci.edu : pub/machine-learning-databases.

experiment is just to test the efficiency of ILA with different number of attributes, so number of rules and other statistics are not shown. Table 4 shows the results. This experiment had been performed on an IBM compatible machine with 64-bit Windows 7 professional OS, 3.2 GHz Intel Core i5-4460 processor, and 8.00 GB RAM.

Table 3. The characteristics of letter recognition dataset.

Domain Characteristic	**Letter recognition dataset**
Number of attributes	16+1
Number of examples	20000
Average Values per attribute	≅ 15
Number of Class Values	26
Average Distribution of Examples Among Class Values	≅770

As noted in Table 4, ILA worked efficiently for up to 6 attributes, and it spent a long time to give results for 7 to 9 attributes, but it continued beyond this limit with extremely large time spans. Actually it works better with other datasets, but not with the letter recognition dataset. This is due to the fact that this dataset, as shown in Table 3, is considered large with respect to all of its parameters; i.e. number of attributes which is 16, number of class values which is 26, number of examples which is 20000, and finally with respect to average number of attribute values which is around 15.

So any reduction in number of attributes is considered a gain with respect to the efficiency. The suggested feature subset selection approach is discussed in details in the next subsection, and will be tailored with ILA in the following subsection.

3 The suggested Feature subset selection approach

Let's first discuss the way the original ILA algorithm works and then suggest a new approach for selecting the most relevant features accordingly in such a way to get benefit of the internal processes of ILA to include as much relevant features as possible.

Table 4. Running ILA on letter recognition dataset with different number of attributes

# of attributes	**# of combinations**	**# of examples**	**Time (hh:mm:ss:ms)**
2	3	130	00:00:00.2064561
3	7	530	00:00:00.3506684
4	15	1201	00:00:18.0517892
5	31	3089	00:09:33.8527276
6	63	7933	00:32:25.4105325
7	127	13075	00:49:26.7303153
8	255	15729	01:05:13.4654722
9	511	16784	01:49:27.8553275
. . .			
16	65535	20000	04:13:37.2124665

ILA, as mentioned above, works in an iterative fashion with respect to rules generation. In the first iteration it takes the attributes with combination 1, i.e. the single attributes and generates their associated rules, and then takes the attributes with combination 2, that is each pair of attributes in the dataset will be considered, and generates their associated rules, and so on. Let's consider a dataset with 10 attributes and computes its associated combinations. This dataset contains a total of 1023 combinations. These combinations and their associated number of elements are depicted in Table 5.

As noted in Table 5, there are only 10 combinations with single attribute in each combination for this dataset, and since the majority of feature selection approaches manipulate only single attributes, ILA, if applied with these approaches, is prohibited from getting benefit from excluding the irrelevant combinations in the other remaining 1013 combinations. This means that a significant amount of reductions in the dataset can be obtained which is reflected positively on the efficiency of ILA.

So if we succeeded to building a new approach in such a way to be able to exclude combinations of attributes rather than single attributes, it will be of great benefit to ILA.

Before describing the algorithm, it is good to discuss some important points and definitions.

Definition (1) duplicated examples
Two examples in a dataset are said to be duplicated if they have identical values of their attributes and the class attribute, that is: two examples Ψ and δ are duplicated with respect to a class **Ω** iff attributes(Ψ) = attributes(δ) and class attribute(Ψ) = class attribute(δ). For example, the following two examples are duplicated, knowing that α and β are the attributes, while **Ω** is the class attribute:
Ψ: α , β , **Ω**, and δ: α , β , **Ω**. In this case, all duplicated examples should be eliminated keeping only one of them. Number of duplicates is denoted by the variable Cd. Cd in this case is 2.

Definition (2) contradicted examples
Two examples are said to be contradicted if their attributes are identical and have different class attribute, i.e. the two examples Ψ and δ contradicted each other iff attributes(Ψ) = attributes(δ) and class attribute(Ψ)≠class attribute(δ). Examples Ψ and δ contradict each other in the following case, knowing that α and β are the attributes, while µ is the class attribute:
Ψ: α , β , **Ω**, and δ: α , β , µ. In this case, both examples should be removed from the dataset. Number of contradicts is denoted by Cc and its value in this case is 2.

Definition (3) missed out classes
For a dataset δ with attributes α1, α2, … αn and a class attribute with values β1, β2, … βm, if some αi where i=1 to n-1 are excluded from δ and this results in disappearing of some βj, where1 ≤ j ≤ m, then this case is called missed out classes.
This happens if some attributes are excluded from a dataset, results in losing some class values completely from the dataset.

Table 5. Number of elements in each combination for a dataset with 10 attributes

combination	# of elements in a combination
1	10
2	45
3	120
4	210
5	252
6	210
7	120
8	45
9	10
10	1

Definition (4) attribute combination irrelevancy description
Combinations of attributes are of three types with respect to irrelevancy:

1. **Strongly irrelevant combination**: it is the combination that if excluded from a dataset, enhancements may be achieved in all or some of the following factors: performance, number of resulted rules, average number of conditions in the resulted rules, but the induction power (accuracy) of the resulted rules on the original dataset remains 100%.
2. **Weakly irrelevant combination**: it is the combination that if excluded from a dataset, enhancements may be achieved in all or some of the following factors: performance, number of resulted rules, average number of conditions in the resulted rules, but the induction power (accuracy) of the resulted rules on the original dataset is less than 100%. The degree of its weakness is measured by the farness of the resulted accuracy from 100%. The required "*farness*" (in the model is denoted by *PredefinedRatio*) can be specified by the user.

3. **Relevant combination**: a combination is relevant if it is not strongly or weakly irrelevant combination. This type of combinations is not excluded at all.

As the main aim of inductive learning algorithms is to maximize their classification power on unseen test examples (which is called the accuracy of the algorithm), we categorize combinations as the above three types according to accuracy and these types will be used to formulate a heuristic function that will guide the suggested model to exclude or not to exclude combinations. These three types are formulated in the following points:

1. A combination is **strongly irrelevant** iff after it is excluded the value $C_c = 0$.
2. A combination is **weakly irrelevant** iff after it is excluded the ratio C_c / $C_d > 0$. This ratio will be denoted hereafter by C_cC_d*Ratio*. The degree if its weakness is measured by the farness of the resulted ratio from 0. The required "*farness*" (in the model is denoted by *PredefinedRatio*) can be specified by the user.
3. A combination is **relevant** if $C_d = 0$ or it causes; when excluded, a *missed out class value* case (denoted by *MissDecisionClassValue).*

The steps of the algorithm are illustrated as listed as Algorithm 1.

As noted, the algorithm excludes the irrelevant combinations each time from the original dataset that contains all attributes. This is because of the fact that an attribute may be relevant alone and irrelevant when combined with other attributes and vice versa. Please note that the CcCdRatio is computed as follows: if, for example, for a dataset Cc = 20 and Cd = 50 then (Cc / Cd)*100 = 40%. This means that this case is 40 far from 0% (the perfect case). But in the literature, it is used to use 100% to denote completeness and perfection. So, to be compatible with what is used in the literature, we subtract the resulted ratio from 100. So, the resulted ratio in the example became 100-40, which is 60%. Both values have the same meaning, but the later value is meaningful and more readable.

4 ILA-3

4.1 General Requirements

1. The examples are to be listed in a table where each row corresponds to an example and each column contains attribute values.
2. A set T of *m* training examples, each example composed of *k* attributes and a class attribute with *n* possible decisions.
3. A rule set, R, with an initial value of ϕ.
4. All rows in the table are initially unmarked.
5. *PredefinedRatio* is the threshold value for accuracy required. Its initial default value is 100. It denotes the required accuracy of the resulted rules. So, %90 for *PredefinedRatio* means that the user wants that the resulted rules to cover 90% of the cases. As its value gets smaller, number of rules gets smaller and simpler.

4.2 The Algorithm

The new algorithm ILA-3 is shown as Algorithm 2. The algorithm divides the table that contains the dataset into sub-tables, one sub-table for each class value. For each sub-table, it calls CombExclude algorithm to exclude the irrelevant combinations and stores them in ExcludedCombinationsList, ILA-3 then continues its usual processes of generating rules for all combinations except those exist in ExcludedCombinationsList, i.e. except the irrelevant combinations. Please note that there is no need to call CombExclude for every combination in each sub-table, so if the combination is already generated in ExcludedCombinationsList then there is no need to generate it again. Actually, CombExclude will be called only for the first sub-table, since all combinations will be generated there.

5 An Illustrative Example

To best understand ILA-3, let's go through a simple illustrative example using an artificial dataset as shown in Table 6. This dataset has 5 attributes in addition to the class attribute.

Applying ILA on this dataset generates 12 rules with 2.333 average conditions on their LHS and .0698 second as their execution time. The list of resulted rules and other details are shown in Fig. 1.

ALGORITHM 1. ***CombExclude****(in: D, PredefinedRatio, j, out: ExcludedCombinationsList(j))*

```
Input: Dataset D that contains n attributes, where $n \geq 2$ and one decision class
Output: ExcludedCombinationsList(j): the set that contains the excluded irrelevant combinations that contains j attributes
Process:
  1.  ExcludedCombinationsList(j) = $\Phi$
  2.  $D_{temp}$= D
  3.  i = 1
  4.  #Comb = n!/(j!*(n-j)!)
  5.  Do While (i<=#Comb) and ($C_cC_dRatio \geq PredefinedRatio$)
          5.1.  $MaxC_cC_dRatio = -\infty$
          5.2.  MissDecisionClassValue = false
          5.3.  repeat
                    5.3.1.  Generate a new combination Nc from $D_{temp}$ with j attributes
                    5.3.2.  Remove Nc from $D_{temp}$ along with its data
                    5.3.3.  $C_d$ = number of duplicates in $D_{temp}$
                    5.3.4.  If ($C_d$ = 0) or (MissDecisionClassValue = true) then go to step 5.3.11.
                    5.3.5.  Eliminate duplicates from $D_{temp}$
                    5.3.6.  $C_c$ =  number of contradicts in $D_{temp}$
                    5.3.7.  Eliminate contradicts from $D_{temp}$
                    5.3.8.  If ((MissDecisionClassValue = true) then go to step 5.3.11.
                    5.3.9.  $C_cC_dRatio$ = 100-(($C_c$ / $C_d$)*100 with upper limit = 100)
                    5.3.10.  If ($C_cC_dRatio > MaxC_cC_dRatio$) and ($C_cC_dRatio >= PredefinedRatio$)  then ($MaxC_cC_dRatio = C_cC_dRatio$)
                             and ($Max_{Comb}$= Nc)
                    5.3.11. Restore Nc into $D_{temp}$ along with its data
                    5.3.12. Increase i by 1
                Until  (i>#Comb )
          5.4.  If  $Max_{Comb} <> \Phi$  then
                    5.4.1.  Append combination ($Max_{Comb}$) into ExcludedCombinationsList(j)
                    5.4.2.  Remove combination ($Max_{Comb}$) from $D_{temp}$ permanently along with its data
                    5.4.3.  Eliminate duplicates from $D_{temp}$
                    5.4.4.  Eliminate contradicts from $D_{temp}$
                    5.4.5.  let $Max_{Comb} = \Phi$
                    5.4.6.  n=n-j
                    5.4.7.  if $n \leq 0$ then go to step 6
                    5.4.8.  go to step 4
                end if
      end while
  6.  END
```

ALGORITHM 2. ILA-3(in: *PredefinedRatio*, out: R)

Step1: Input the *PredefinedRatio* value (≥0 and ≤100)

Step2: Partition the table which contains *m* examples into *n* sub-tables. One table for each possible value of the class attribute.

(* steps 3 through 9 are repeated for each sub-table *)

Step3: Initialize attribute combination count j as j = 1.

Step4: if *ExcludedCombinationsList(j)* ≠ *Φ* then
    Call *CombExclude*(out: dataset T, *PredefinedRatio*, j, in: *ExcludedCombinationsList(j)*)

Step5: For each combination of j attributes not in *ExcludedCombinationsList(j)*, count the number of occurrences of attribute values that appear under the same combination of attributes in unmarked rows of the sub-table under consideration but at the same time that should not appear under the same combination of attributes of other sub-tables. Call the first combination with the maximum number of occurrences as max-combination.

Step6: If max-combination = ϕ, increase *j* by *1* and go to Step 4.

Step7: Mark all rows of the sub-table under consideration, in which the values of max-combination appear, as classified.

Step8: Add a rule to R whose left hand side comprise attribute names of max-combination with their values separated by AND operator(s) and its right hand side contains the decision attribute value associated with the sub-table.

Step9: If all rows are marked as classified, then move on to process another sub-table and go to Step 3. Otherwise (i.e., if there are still unmarked rows) go to Step 5. If no sub-tables are available, exit with the set of rules obtained so far.

In order to understand ILA-3 well, let us go through its steps one by one on the above mentioned dataset. It is worth to mention that this dataset is small and artificial, and is used only to illustrate the steps of ILA-3.

As the first step of ILA-3 indicates, let us assign the value 100 to PredefinedRatio. This means that we will consider only the combinations that if excluded, the accuracy stay at 100%, i.e. only strong irrelevant combinations will be removes. Applying the second step, the original dataset is divided into two sub-tables: sub-table1 for class value 0 and sub-table2 for class value 1. All the following steps will be executed for each sub-table.

According to step 3, j is assigned a value 1, and then CombExclude is called to exclude the irrelevant combinations that include 1 attribute each. Moving to CombExclude algorithm, it will compute number of combinations as 5 as follows: A, B, C, D, and E. It will find the most irrelevant combination out of these, by computing their CcCdRatios and choosing the maximum ratio among them. The results are depicted in Table 7.

This means that the combinations B and C are strongly irrelevant. According to the algorithm, the first combination with maximum value, i.e. B will be appended to ExcludedCombinationsList(1)and will be excluded from the dataset with its data. Now the previous process will be repeated on the dataset without B. The results are shown in Table 8.

This means that, after excluding B, the combination C is a strong irrelevant combination, and can be excluded with its data from the dataset and appended to ExcludedCombinationsList(1). Now, the above process is repeated without B and C. Table 9 shows the results.

Since all ratios are less than the predefined ratio, this step stops, ILA-3 generates the rules with 1 attribute in their left hand side, the original dataset is restored, and moves to the second iteration of the algorithm, i.e. j=2.

Table 6. The artificial dataset used in the illustrative example

A	B	C	D	E	class
1	1	2	3	2	1
1	2	1	1	1	0
1	2	1	1	2	1
1	2	1	2	1	0
1	2	1	2	1	0
1	3	1	2	1	0
1	3	1	2	1	0
1	3	1	3	2	1
1	3	2	2	1	0
1	3	2	2	2	0
1	3	2	2	2	0
1	3	2	2	2	0
1	3	2	2	2	0
1	3	2	3	1	1
2	1	1	2	1	1
2	1	1	2	2	0
2	1	1	2	2	0
2	2	1	1	1	1
2	3	1	2	1	1
2	3	1	2	1	1
2	3	1	3	1	0
2	3	1	3	2	1
3	1	2	3	2	0
3	2	1	1	1	1
3	2	1	1	1	1

Table 7. C_cC_dRatio for all combinations with 1 attribute

Excluded combination	**# duplicates**	**# contradictions**	C_cC_d***Ratio***
A	10	6	40%
B	10	0	100%
C	9	0	100%
D	9	4	55.555%
E	9	6	33.333%

```
Data Set File Name: artificial dataset.txt
 Number of Attributes: 5
 Number of Classes: 2
 Number of Samples: 25
 Evaluation Method       :Random Sampling
 Percentage of unseen is       :0
 Number of experiments is      : 1
 Number of training samples is : 25
 Number of unseen samples   is : 0
=====================================
Number of rules: 12
 Average Number of conditions: 2.333333
 Rules:
 If A = 1 and D  = 3 => 1
 If A = 3 and B  = 2 => 1
 If A = 2 and B  = 2 => 1
 If B = 2 and E  = 2 => 1
 If B = 1 and E  = 1 => 1
 If A = 2 and D  = 2 and E  = 1 => 1
 If B = 3 and C  = 1 and E  = 2 => 1
 If A = 1 and D  = 2 => 0
 If D = 2 and E  = 2 => 0
 If A = 3 and B  = 1 => 0
 If A = 1 and C  = 1 and E  = 1 => 0
 If A = 2 and D  = 3 and E  = 1 => 0
======================================
========     Final Result          ============
======================================
 Average Number of rules: 12
 Average Number of conditions: 2.3333332538
 Average Accuracy: 100%
 Average Precision: 1
 Average Recall: 1
 Average F1 Score: 1
 Total time Consumed is: 00:00:00.0698611
```

Fig. 1. Rules resulted from applying ILA on the artificial dataset

Table 8. *C_cC_dRatios* for all combinations with 1 attribute, except the combination B.

Excluded combination	**# duplicates**	**# contradictions**	***C_cC_dRatio***
A	2	6	0%
C	12	0	100%
D	3	8	0%
E	2	6	0%

Table 9. *CcCdRatio* for all combinations with 1 attribute, except the combinations B and C.

Excluded combination	**# duplicates**	**# contradictions**	***C_cC_dRatio***
A	3	8	0%
D	3	8	0%
E	2	6	0%

This means that the previous process will be repeated, but on combinations with 2 attributes, which are: AB, AC, AD, AE, BC, BD, BE, CD, CE, and DE. Without going into details, the combination BC has the maximum ratio which is 100% with 12 duplicates and 0 contradictions. So it will be appended into ExcludedCombinationsList(2) and removed from the dataset along with its associated data, the process will be repeated on the rest combinations. Since all their computed CcCdRatios are 0% (i.e. no one of them is greater than or equal the predefined ratio), this step stops, ILA-3 generates the rules with 2 attribute in their left hand side, the original dataset is restored, and moves to the third iteration of the algorithm, i.e. j=3.

All the 10 three-attribute combinations have CcCdRatios less than the predefined ratio. So, none of them will be excluded and ILA-3 generates the rules with 3 attributes, the original dataset is restored, and moves to the fourth iteration of the algorithm, i.e. j=4. The same situation as when j=3 happened also here. So, none of them will be excluded and ILA-3 generates the rules with 4 attributes, and moves to the last iteration of the algorithm, i.e. j=5. Now for j=5, only one combination will result with all attributes, and since no duplications or contradictions should exit in the original dataset, ILA-3 will generate the rules with 5 attributes. The algorithm stops here and the resulted rules are shown in Fig. 2.

It is noted from Fig. 2 that a reduction in number of rules has been obtained, from 12 rules with ILA to 11 rules with ILA-3, and a reduction in execution time has been noted also, with an increase of 0.03 on the complexity of the resulted rules, but with the same accuracy i.e. 100%. Actually, the results of both algorithms are too close, because the dataset considered here is a very small one used for illustration purposes of ILA-3. The experiments below show that significant reductions can be obtained as the datasets get larger.

6 Experiments and Results

Three sets of experiments have been conducted in this paper. In the first set, ILA-3 is compared with the original algorithm ILA. While in the second set, ILA-3 has been applied and tested on several datasets that are ranged from small to large. In the third set, ILA-3 has been applied on the large dataset: the letter recognition dataset mentioned earlier in the paper. All experiments in this section have been conducted on an IBM compatible machine with 64-bit Windows 7 professional OS, 3.2 GHz Intel Core i5-4460 processor, and 8.00 GB RAM.

In the first set of experiments, ILA-3 is compared with the original ILA. For this purpose, we used seven training sets with different number of attributes, examples, and class values. The characteristics of these training sets are summarized in Table 10 in Appendix A (all the tables of the experiments are in Appendix A). These training sets are automatically generated realistic data sets, using the Synthetic Classification Data Set Generator (SCDS) version 2. Using synthesized data sets is another important way to assess inductive learning algorithms.

Table 11 summarizes all the results obtained through this experiment. The first column of the table shows number of combinations in the dataset, the second column shows number of rules generated by original ILA, while the third column shows number of combinations excluded, number of rules generated by ILA-3, and the accuracy of these rules for classifying the examples in the original dataset.

It is noted from Table 11 that ILA-3 reduces number of rules generated with acceptable levels of accuracy on the original dataset. It is noted also that ILA-3 works well on large datasets where number of combinations are large.

Table 12 shows that the efficiency of ILA-3 is much better than that of the original ILA. This is because of the fact that in ILA the entire dataset is manipulated without excluding any combination. But with ILA-3, all irrelevant combinations are excluded and not involved in the induction process itself, especially if we know that the time spent in excluding the combinations is much less that the time consumed in the case they involved in the induction process itself.

In the second set of experiments, ILA-3 has been applied on several datasets that are ranged from small to large, namely; object classification, monk1, balance, and Vote. These data sets and all datasets used in all experiments are described in Table 13 which are obtained from the University of California Irvine Repository of Machine Learning Databases and Domain Theories via anonymous ftp to charlotte.ics.uci.edu : pub/machine-learning-databases. The original dataset that contains all the attributes will be used as the unseen test dataset to compute the accuracy of the resulted classifiers after excluding attributes.

In this set of experiments, ILA-3 is compared with the original ILA and other two known inductive algorithms which are ID3 [43] and AQ [44]. Table 14 Fig. 2. The resulted rules after applying ILA-3 on the artificial dataset shows the result of applying these four algorithms on the three datasets: Monk1, Vote, and Balance. The first column shows the results of applying ILA, ID3, AQ and ILA-3 on Monk1 dataset, while the second column is applying the four algorithms on Vote dataset, and the third is for applying them on Balance dataset. Three pieces of information are needed here for each application of the induction algorithm on the datasets; namely: number of rules since it is known that as less number of rules that can infer more examples are preferable, average number of conditions on the left hand side of the resulted rules which indicates the simplicity of the rules, and the execution time spent by the algorithm to produce the rules. It is noted from the results that ILA-3 overcomes ILA and other algorithms for all factors.

Table 15 shows the effect of applying ILA-3 on the three datasets. As shown in the table, a considerable amount of reductions has been obtained in the three datasets.

This reduction reduces also number of examples in the datasets. This justifies the efficiency of ILA-3 as shown in Table 14. All this happened with keeping accuracy, as shown in the last row of Table 15, at acceptable levels.

In the last experiment, ILA-3 is applied on the large dataset mentioned earlier in section 2.1, i.e. the letter recognition dataset. Table 16 shows the results. It is clear from the results obtained that ILA-3 enhances the efficiency of ILA significantly.

7 Summary and Conclusions

In this paper, a new version of ILA, called ILA-3 inductive learning algorithm, had been proposed and tested. ILA-3 is the third generation of ILA, after ILA-2. A new feature selection algorithm, called CombExclude had been discussed and tailored with ILA to produce the new inductive algorithm: ILA-3. CombExclude excludes all irrelevant combinations of attributes from the dataset under consideration while ILA-3 is running. The results obtained showed that

```
Data Set File Name: artificial data set.txt
 Number of Attributes: 5
 Number of Classes: 2
 Number of Samples: 13
 Evaluation Method      :Random Sampling
 Percentage of unseen is      : 0
 Number of experiments is      : 1
 Number of training samples is : 13
 Number of unseen samples   is : 25
=================================
 Number of rules: 11
 Average Number of conditions: 2.363636
 Rules:
 If A = 1 and D  = 2 => 0
 If D = 2 and E  = 2 => 0
 If A = 3 and D  = 3 => 0
 If A = 1 and D  = 1 and E  = 1 => 0
 If A = 2 and D  = 3 and E  = 1 => 0
 If A = 1 and D  = 3 => 1
 If A = 2 and D  = 1 => 1
 If A = 3 and D  = 1 => 1
 If D = 1 and E  = 2 => 1
 If A = 2 and D  = 2 and E  = 1 => 1
 If A = 2 and D  = 3 and E  = 2 => 1
 ==================================
 =======     Final Result       ===========
 ==================================
 Average Number of rules: 11
 Average Number of conditions: 2.36363625
 Average Accuracy: 100%
 Average Precision: 1
 Average Recall: 1
 Average F1 Score: 1
 Total time Consumed is: 00:00:00.0643113
```

Fig. 2. The resulted rules after applying ILA-3 on the artificial dataset

the efficiency of ILA had been greatly enhanced by the new version. The experiments showed also that ILA-3 overcomes some other powerful inductive learning algorithms as ID3 and AQ with number of rules generated, simplicity of rules, accuracy of generated rules, and time consumed to produce rules.

References

[1] M. Tolun, and S. Abu-Soud. ILA: An Inductive Learning Algorithm for Rule Extraction, Expert Systems with Applications, 14(3), (1998) 361-370.

[2] M.A. Hall, and , L.A. Smith, Practical Feature Subset Selection for Machine Learning. Proc Australian Computer Science Conference, 181-191, Perth, Australia, 1998.

[3] L. Huan and Y. Lei. Toward Integrating Feature Selection Algorithms for Classification and Clustering, IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 17, NO. 4, APRIL 2005.

[4] A.L. Blum and P. Langley, “Selection of Relevant Features and Examples in Machine Learning,” Artificial Intelligence, vol. 97, pp. 245-271, 1997.

[5] G.H. John, R. Kohavi, and K. Pfleger, “Irrelevant Feature and the Subset Selection Problem,” Proc. 11th Int’l Conf. Machine Learning, pp. 121-129, 1994.

[6] K. Kira and L.A. Rendell, “The Feature Selection Problem: Traditional Methods and a New Algorithm,” Proc. 10th Nat’l Conf. Artificial Intelligence, pp. 129-134, 1992.

[7] Y. Rui, T.S. Huang, and S. Chang, “Image Retrieval: Current Techniques, Promising Directions and Open Issues,” Visual Comm. and Image Representation, vol. 10, no. 4, pp. 39-62, 1999.

[8] D.L. Swets and J.J. Weng, “Efficient Content-Based Image Retrieval Using Automatic Feature Selection,” IEEE Int’l Symp. Computer Vision, pp. 85-90, 1995.

[9] M. Ben-Bassat, “Pattern Recognition and Reduction of Dimensionality, ”Handbook of Statistics-II, P.R. Krishnaiah and L.N. Kanal, eds., pp. 773-791, North Holland, 1982.

[10] A. Jain and D. Zongker, “Feature Selection: Evaluation, Application, and Small Sample Performance,” IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 2, 153-158, Feb. 1997.

[11] M. Dash, K. Choi, P. Scheuermann, and H. Liu, “Feature Selection for Clustering-a Filter Solution,” Proc. Second Int’l Conf. Data Mining, pp. 115-122, 2002.

[12] M. Dash and H. Liu, “Feature Selection for Classification, ”Intelligent Data Analysis: An Int’l J., vol. 1, no. 3, pp. 131-156, 1997.

[13] K. Nigam, A.K. Mccallum, S. Thrun, and T. Mitchell, “Text Classification from Labeled and Unlabeled Documents Using EM,” Machine Learning, vol. 39, 103-134, 2000.

[14] Y. Yang and J.O. Pederson, “A Comparative Study on Feature Selection in Text Categorization,” Proc. 14th Int’l Conf. Machine Learning, pp. 412-420, 1997.

[15] K.S. Ng and H. Liu, “Customer Retention via Data Mining,” AI Rev., vol. 14, no. 6, pp. 569-590, 2000.

[16] W. Lee, S.J. Stolfo, and K.W. Mok, “Adaptive Intrusion Detection: A Data Mining Approach,” AI Rev., vol. 14, no. 6, pp. 533-567,2000.

[17] S. Abu-Soud, PaSSIL: A New Keystroke Dynamics System for Password Strengthening Based on Inductive Learning. The WSEAS Transactions on Information Science and Applications, Volume 13, 2016.

[18] M.A. Hall, “Correlation-Based Feature Selection for Discrete and Numeric Class Machine Learning,” Proc. 17th Int’l Conf. Machine Learning, pp. 359-366, 2000.

[19] H. Liu and R. Setiono, “A Probabilistic Approach to Feature Selection-A Filter Solution,” Proc. 13th Int’l Conf. Machine Learning, pp. 319-327, 1996.

[20] L. Yu and H. Liu, “Feature Selection for High-Dimensional Data: A Fast Correlation-Based Filter Solution,” Proc. 20th Int’l Conf. Machine Learning, pp. 856-863, 2003.

[21] R. Caruana and D. Freitag, “Greedy Attribute Selection,” Proc.11th Int’l Conf. Machine Learning, pp. 28-36, 1994.

[22] J.G. Dy and C.E. Brodley, “Feature Subset Selection and Order Identification for Unsupervised Learning,” Proc. 17th Int’l Conf. Machine Learning, pp. 247-254, 2000.

[23] Y. Kim, W. Street, and F. Menczer, “Feature Selection for Unsupervised Learning via Evolutionary Search,” Proc. Sixth ACM SIGKDD Int’l Conf. Knowledge Discovery and Data Mining, pp. 365-369, 2000.

[24] R. Kohavi and G.H. John, “Wrappers for Feature Subset Selection,” Artificial Intelligence, vol. 97, nos. 1-2, pp. 273-324, 1997.

[25] S. Das, “Filters, Wrappers and a Boosting-Based Hybrid for Feature Selection,” Proc. 18th Int’l Conf. Machine Learning, pp. 74-81, 2001.

[26] A.Y. Ng, “On Feature Selection: Learning with Exponentially Many Irrelevant Features as Training Examples,” Proc. 15th Int’l Conf. Machine Learning, pp. 404-412, 1998.

[27] E. Xing, M. Jordan, and R. Karp, “Feature Selection for High-Dimensional Genomic Microarray Data,” Proc. 15th Int’l Conf. Machine Learning, pp. 601-608, 2001.

[28] R. Kohavi and G. John. Wrappers for feature subset selection, Artificial Intelligence 97, 1997, pp. 273-324.

[29] H. Almuallim and T. Dietterich. Learning with many irrelevant features, in: Proceedings AAAI-91, Anaheim, CA (MIT Press, Cambridge, MA, 1991) 547-552.

[30] H. Almuallim and T. Dietterich. Learning Boolean concepts in the presence of many irrelevant features, Artificial Intelligence 69, 1994, 279-306.

[31] K. Kim and L.A. Rendell. The feature selection problem: Traditional methods and a new algorithm, in: Proceedings AAAI-92, San Jose, CA (MIT Press, MA, 1992) 129-134.

[32] K. Kim and L.A. Rendell. A practical approach to feature selection, in: Proceedings 9th International Conference on Machine Learning, Aberdeen, Scotland (Morgan Kaufmann, Los Altos, CA, 1994).

[33] I. Kononenko. Estimating attributes: analysis and extensions of Relief, in: F. Bergadano and L. De Raedt, eds., Proceedings European Conference on Machine Learning (1994).

[34] P. Langley, “Selection of Relevant Features in Machine Learning, ”Proc. AAAI Fall Symp. Relevance, pp. 140-144, 1994.

[35] G.H. John, R. Kohavi and K. Pfleger. Irrelevant features and the subset selection problem, in: Proceedings 11th International Conference on Machine Learning, New Brunswick, NJ (Morgan Kaufmann, San Mateo, CA, 1994) 121-129.

[36] P. Langley and S. Sage. Oblivious decision trees and abstract cases, in: Working Notes of the AAAI-94 Workshop on Case-Based Reasoning, Seattle, WA, 1994, 113-117.

[37] Abu-Soud S., “A Framework for Integrating Decision Support Systems and Expert Systems with Machine Learning”, Proceeding of the 10th International Conference on Industrial and Engineering Applications of AI and ES, June 1997, Atlanta, USA.

[38] Abu-Soud S., “A Disjunctive Learning Algorithm for Extracting General Rules”, Journal of Institute of Mathematics and

Computer Science (Computer Science Series), Vol. 10, No. 2 (1999) 201-217.

[39] Oludag M., Tolun M., Sever H., and Abu-Soud S., “ILA-2: An Inductive Learning Algorithm for Knowledge Discovery”, Cybernetics and Systems: An International Journal, vol. 30, no. 7, Oct.-Nov. 1999.

[40] Haj Hassan M. and Abu-Soud S., “A Parallel Inductive Learning Algorithm,” AMSE journal, France, Dec. 2000.

[41] Abu-Soud S. and Al Ibrahim A., DRILA: A Distributed Relational Inductive Learning Algorithm, WSEAS Transactions on Computers, Issue 6, Volume 8, June 2009, ISSN: 1109-2750.

[42] Abu-Soud S., ILATalk: A New Multilingual Text-To-Speech Synthesizer with Machine Learning, International Journal of Speech Technology, Volume 19, March 2016, Issue 1, pp 55-64.

[43] J.R. Quinlan. Learning efficient classification procedures and their application to chess end games, in: R.S. Michalski, J.G. Carbonell and T.M. Mitchell, eds., Machine Learning: An Artificial Intelligence Approach (Morgan Kaufmann, San Mateo, CA, 1983).

[44] R.S. Michalski, I. Mozetic, J. Hong, and N. Lavrac. The Multipurpose Incremental Learning System AQ15 and Its Testing Application to Three Medical Domains, Proc. of the Fifth National Conference on Artificial Intelligence, Philadelphia, PA: Morgan Kaufmann, 1986, 1041-1045.

Appendix A: Tables of Experiments

Table 10.Characteristics of the seven Training Sets.

	Number Of Examples	Number Of Attributes	Average Values Per Attribute	Number Of Class Values	Distribution Of Examples Among Class Values
Data Set 1	100	3+1	20	3	1. 44.0% are X 2. 28.0% are Y 3. 28.0% are Z
Data Set 2	300	5+1	5	2	1. 50.0% are X 2. 50.0% are Y
Data Set 3	500	5+1	3	4	1. 47.8% are X 2. 29.4% are Y 3. 16.4% are Z 4. 6.4% are W
Data Set 4	500	19+1	6	4	1. 46.8% are X 2. 29.6% are Y 3. 19.0% are Z 4. 4.6% are W
Data Set 5	1000	12+1	6	3	1. 83.8% are X 2. 13.0% are Y 3. 3.2% are Z
Data Set 6	5000	15+1	10	2	1. 51.2% are X 2. 48.8% are Y
Data Set 7	10000	25+1	100	6	1. 33.7% are X 2. 25.8% are Y 3. 19.5% are Z 4. 11.1% are X 5. 7.9% are Y 6. 2.0% are Z

Table 11.Comparison between *ILA* and *ILA-3* algorithms on the seven datasets.

Algorithm / dataset	ILA			ILA-3		
	# of Combinations	**Accuracy %**	**# rules**	**# combinations excluded**	**Accuracy %**	**# rules**
Data Set 1	7	100%	17	1	93.04	13
Data Set 2	31	100%	51	14	91.73	33
Data Set 3	31	100%	64	17	92.44	41
Data Set 4	524287	100%	111	123000	94.12	52
Data Set 5	4095	100%	203	1457	94.68	101
Data Set 6	32767	100%	627	12903	90.15	237
Data Set 7	33554431	100%	892	7482385	98.61	169

Table 12. Comparison of the speed of ILA and ILA-3 algorithms on the seven datasets.

Dataset	**ILA (hh:mm:ss:ms)**	**ILA-3 (hh:mm:ss:ms)**	**% Reduction in time**
Data Set 1	00:00:00.3188432	00:00:00.2943221	%7.69
Data Set 2	00:00:01.7600326	00:00:01.2703279	%27.82
Data Set 3	00:00:01.9106653	00:00:01.2857665	%32.70
Data Set 4	01:52:47.9898752	01:04:22.6785534	%42.92
Data Set 5	01:08:52.0047736	00:55:36.1200744	%19.27
Data Set 6	01:28:23.6338642	01:01:28.8755375	%30.44
Data Set 7	03:37:55.2643865	01:27:18.1232709	%59.93

Table 13. Description of the Domains

Domain / Characteristic	**Object Classification**	**Monk1**	**Balance**	**Vote**
Number of attributes	3+1	6+1	4+1	16+1
Number of examples	7	124	625	300
Average Values per attribute	3	2.83	5	2
Number of Class Values	2	2	3	2
Distribution of Examples Among Class Values	57.14% are yes 42.86% are no	50% are 0 50% are 1	46.08% are L 07.84% are B 46.08% are R	61.33% are democrat 38.67% are republican

Table 14. The results of applying ILA, ID3, AQ , and ILA-3 on the datasets: monk1, Vote, and Balance.

	Monk1				**Vote**				**Balance**			
	ILA	**ID3**	**AQ**	**ILA-3**	**ILA**	**ID3**	**AQ**	**ILA-3**	**ILA**	**ID3**	**AQ**	**ILA-3**
Number of Rules	32	54	49	***28***	42	68	53	***35***	303	401	312	***273***
Average number of conditions	3.28	4.62	3.39	***3.19***	3.45	3.75	3.49	***3.06***	3.41	3.85	3.53	***3.17***
Execution Time (s)	1.72	2.52	2.19	***1.35***	4.17	5.23	4.52	***3.25***	0.87	1.84	1.39	***0.47***

Table 15.The results of applying ILA-3 on monk1, Vote, and Balance.

	Monk1	**Vote**	**Balance**
Number of Examples (original/after Exclude)	124/25	300/21	625/416
Combinations (excluded/original)	17/63	1470/65535	4/15
Accuracy on original dataset	96%	98%	95%

Table 16.The results of applying ILA and ILA-3 on letter recognition dataset.

	ILA	**ILA-3**
Number of Rules generated	2314	2167
Average number of conditions	8.427	7.051
Execution Time (hh:mm:ss:ms)	04:13:37.2124665	02:26:18.7728432
Combinations (excluded/original)	65535	3253/65535
Accuracy on original dataset	100%	100%