Towards Real Time Ion Classification Using a Machine Learning Algorithm (C 4.5): A Case Study of a Conducting Polymer Ion Detector

Afshad Talaie* and Jose A. Romagnoli
Chemical Engineering department, The University of Sydney, Sydney, N.S.W., Australia, 2006

Nasser Esmaili
Artificial Intelligence Department, University of New South Wales, Kensington, N.S.W., Australia, 2033

Takahisa Taguchi
Osaka National Research Institute, Midorigaoka, Ikeda, Osaka 563, Japan

Received: 19 January 1997; accepted 25 August 1997

ABSTRACT

The development of an on-line computer based classification system for the automatic detection of different ions, existing in different solutions, is addressed in this study. Three different parameters (current, mass, and resistance) are collected simultaneously. Then these laboratory measurements are used by an algorithm software as a logged data file, resulting in to inducing a decision tree. Later, a systematic software is designed based on the rules derived from this decision tree, to recognize the type of unknown solution used in the experiment. This is a new approach to data acquisition in chemical industries involving conducting polymers.

Key Words: data acquisition, on-line data processing, real time ion recognition, conducting polymers, PPy/DBS

INTRODUCTION

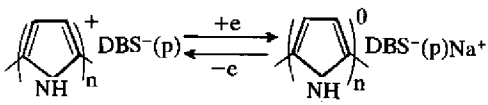
Over the past two decades a range of polymers have been studied with a particular attention for using them as sensing materials. Amongst them, conducting polymers such as polypyrroles [1–3], have been found promissing canditates for different sensing technologies [4–6]. These materials are dynamic, but their unique physical and chemical properties allow us to put their dynamic behaviour under control.

Conducting polymers are easily produced by chemical [7] and electrochemical [8] methods. The electrodeposition of polypyrrole dodecylbenzene-

* Present address: Osaka National Research Institute, Midorigaoka, Ikeda, Osaka 563, Japan.

sulphonic (PPy/DBS) proceeds at the anode with the application of either a constant potential or constant current at the three electrodes of electrochemical cell [9]. The counterion (DBS) is incorporated from the supporting electrolyte used during synthesis. The ability to switch this conducting polymer between the reduced and oxidized states have attracted special attention of material scientists.

This switching process, in solution of 1M NaCl, is described by (Scheme I).



Scheme I

The fact that this material conducts electricity and undergoes various redox (oxidation/reduction) transitions, provides us with a means of communication and control.

The development of intelligent polymeric systems is a multi-deciplinary task, which needs the design of a dynamic data acquisition and processing system.

Recently, different techniques have been introduced [10−14] to monitor the electrical changes occurring at the dynamic surface of polymeric materials. Most recently, a new multi-dimensional technique has been developed [15, 16], which is capable of measuring the current, mass, and resistance of a solution/polymer interface, simultaneously. However, it requires to be combined with a software system which enables us to process a logged data file from laboratory measurement, producing an almost accurate recognition of the type of ion, used in solution. This recognition is based upon a pattern induced from the data. We have already reported the application of different artificial intelligence methods in the conducting polymers sensing technology [17−19].

In our ongoing interests in the design of the integrated/conducting polymer based sensor, we now report on the application of an specific algorithm called C4.5 [20].

In order to employ the algorithm, it is necessary to have a logged data file from the previous laboratory measurements of the polymer/solution interactions. This algorithm has already been applied to variety of dynamic system control applications as diverse as flight simulator control, mobile robot navigation learning, and crane control [21−24]. The previous applications used the algorithm to produce a pattern for the control actions, while in this experiment the induced pattern is used for recognition of the solution.

EXPERIMENTAL

Reagents and Standard Solutions

The polymer synthesis solution consisted of 0.1 M pyrrole (Sigma) and 0.5 M of sodium dodecylbenzenesulphonic acid (Aldrich). These were dissolved in deionized (Milli-Q) water. Nitrogen was used for deoxygenation of the solutions, before the polymerization process.

Instrumentation

Voltametric data were obtained using a BAS CV27 voltamograph (Bio-Analytical Systems, Lafayette, PA, USA). Data were collected using a Maclab (Analog Digital Instruments, Sydney, Australia) interface and Macintosh computer.

A resistometer (developed by CSIRO, Division of Mineral Products, Melbourne, Australia) was employed to collect the polymer's resistance. The quartz crystal microbalance was used to log the data for the changes in the polymer's mass, based on the design previously published [10−12]. During the experiments, the current, mass and resistance data were logged simultaneously into related files so that later can be used and processed for pattern recognition.

Since the C4.5 program is a Unix based software, we have used a DEC work station running X11-R6 system supporting Xwindows. Most parts of the software are in C language. The translating program which is responsible to transfer the resulted decision tree to C language is designed using Awk programming language.

Procedure

The polymer was galvanostatically (current density: 2mA/cm^2 for 2 min) deposited into a gold crystal. An Ag/AgCl reference electrode and a platinum counter electrode were employed.

RESULTS AND DISCUSSION

By employing the S.M.A.C (simultaneous multi-dimensional analysis of conductors) [15], electrochemical characterization (current, mass, and resistance vs. potential) was carried out. Using PPy/DBS, as a test case, this data was collected when pulsed potential waveforms were applied (Figures 1–3). Two potential pulse regions have been chosen at reduction (−0.7 V) and oxidation (+0.5 V) points.

As shown in Figure 1, only 5 μg changes in the mass of PPy/DBS in 1M NaCl was observed, compared with 13 μg and almost 4 μg changes in 1M LiCl (Figure 2) and 0.5M $CaCl_2$ (Figure 3),

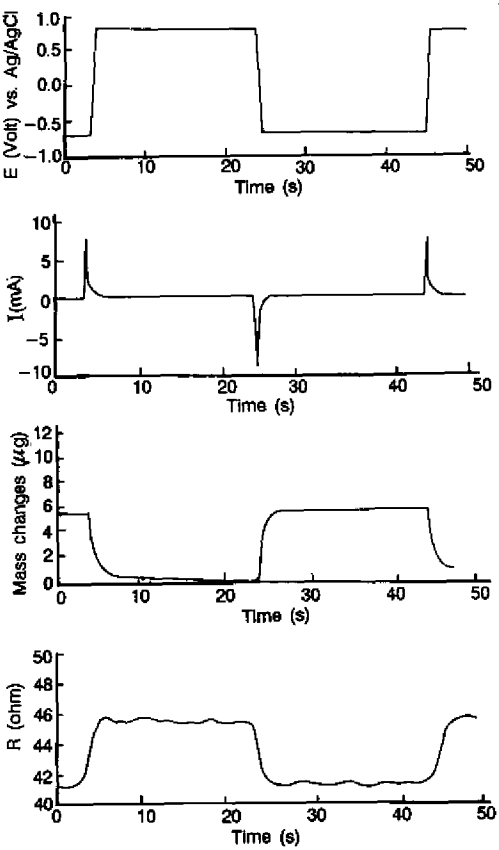


Figure 1. In-situ monitoring of current, mass, and resistance by switching between two different potentials for PPy/DBS in 1M in NaCl.

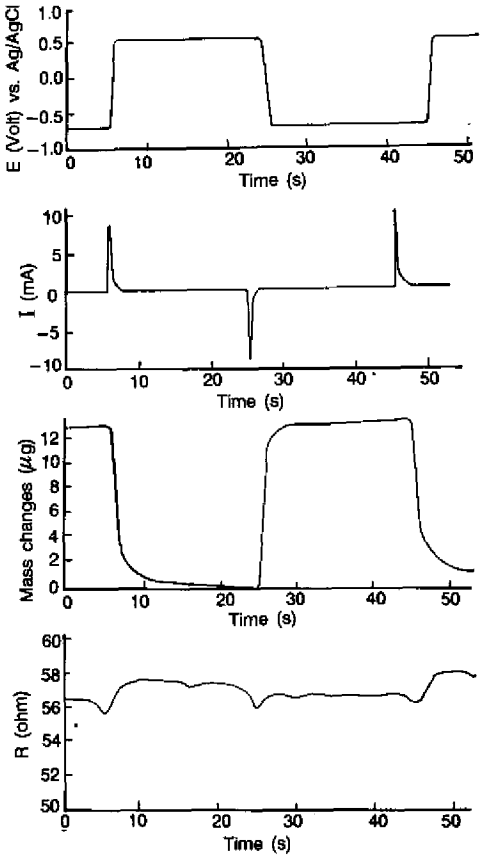


Figure 2. In-situ monitoring of current, mass, and resistance by switching between two different potentials for PPy/DBS in 1M in LiCl.

respectively. Clear changes in reduction/oxidation process in different salts were also observed by considering the current peaks and resistance transition, while the applied potentials were in a pulse waveform.

Ten different experimental works have been performed for freshly prepared PPy/DBS in 1M NaCl, 1M LiCl, and 0.5 M $CaCl_2$. During the experiments, ten pulses at the same potential and time duration were applied. Then, two different samples were taken out from each experiment's results

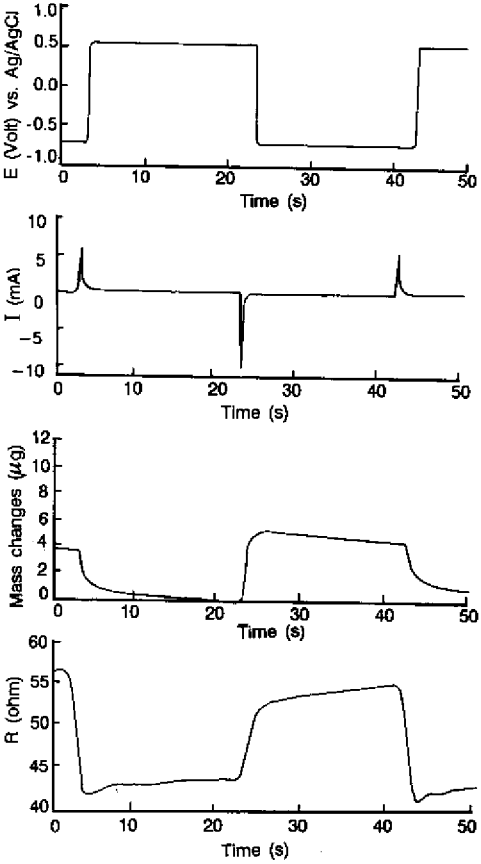


Figure 3. In-situ monitoring of current, mass, and resistance by switching between two different potentials for PPy/DBS in 0.5M $CaCl_2$.

(pulses 9 and 10). This resulted in producing the files of current (I), mass (M) and resistance (R) as a function of potential (E) for each salt. Then combining the attribute files together, so E, I, M and R, respectively, form a record (row) of data, a file created for each salt as shown in Table 1, which contains a portion of such typical data files.

Later, all the files were combined together to form just one huge file and to be fed to a processing program. Since C4.5 cannot deal with the numbers as constant attributes, the data file was fed to a preprocessor to change the positive voltages to 'Pos' and negative voltages to 'Neg'.

C4.5 requires at least two input files. The first is the preprocessed data file, 'DBS.Data' (shown in Table 2a), describing the training cases from which decision trees and/or rule sets are to be constructed. Each line describes one case, providing the values for all the attributes and then the case's class.

The second file, the fundamental file for the task, provides names for classes, attributes, and attribute values, 'DBS.Names' (shown in Table 2b). The first line gives the class names, and the rest of the lines are to name the attributes and also show the specification of the values that the attribute can take. The 'continuous' value indicates that the attribute has a numeric value.

Later, the preprocessed files were used by the C4.5 to produce the decision tree. Using default argument [20] for the test file, C4.5 uses part of the given data file as unseen (training) data, and a small part of it as test (first part of Appendix A). The numbers in parentheses following each leaf, indicate the number of training cases associated with each leaf and the number of them misclassified by the leaf. The program also contains heuristic methods for simplifying decision trees, with the aim of producing more comprehensible structures without compromising accuracy on unseen cases. The second part of Appendix A shows the simplified tree. The output (under the evaluation on trainning data) shows how both the original and simplified trees perform on the training set from which they were constructed. The original tree of 35 nodes misclassifies none of the

Table 1(a–c). Data for Ca salt (a), for Li salt (b), and Na salt (c).

(a)	(b)	(c)
−0.7, −0.02, 3.70139, 56.1408	−0.7, −0.03, 12.9925, 56.5	−0.7, −0.02, 5.10189, 41.0698
−0.7, −0.02, 3.70139, 56.1488	−0.7, −0.03, 13.0376, 56.4999	−0.7, −0.02, 5.10189, 41.056
−0.7, −0.02, 3.70139, 56.173	−0.7, −0.03, 13.0827, 56.4282	−0.7, −0.02, 5.05679, 41.0556
−0.7, −0.02, 3.6563, 56.2134	−0.7, −0.02, 13.0827, 56.3729	−0.7, −0.02, 5.10189, 41.1836
−0.7, −0.02, 3.6563, 56.2683	−0.7, −0.02, 13.1278, 55.918	−0.7, −0.02, 5.10189, 41.3422
0.5, 0.07, 0.589493, 42.1379	0.5, 1.17, 5.28039, 55.9527	0.5, 0.34, 1.6292, 44.6907
0.5, 0.07, 0.544395, 42.1918	0.5, 0.52, 3.83719, 56.2249	0.5, 0.22, 1.3135, 45.0874
0.5, 0.06, 0.544395, 42.2282	0.5, 0.11, 1.4018, 57.192	0.5, 0.16, 1.088, 45.3495
0.5, 0.06, 0.499298, 42.2548	0.5, 0.09, 1.2665, 57.2592	0.5, 0.09, 0.772297, 45.6072
0.5, 0.06, 0.499298, 42.2846	0.5, 0.08, 1.1763, 57.3085	0.5, 0.08, 0.682098, 45.5903

6010 cases, nor the simplified tree, but the program predicts that it will have a higher error rate of 0.4% on unseen cases.

All the above have been carried out without examining the test cases located in file 'DBS.Test', containing the test set which evaluates the classifier produced by the training set. The file of 600 cases is read at this point and each case is classified by the original and simplified trees. The simplified tree and the original tree misclassify 3 and none of the unseen cases, respectively.

The final part of the output is a confusion matrix for the simplified tree on the test cases, showing how the misclassification were distributed. There are 200 cases of class Ca, all of which are correctly classified as Ca, while none are classified as either Li or Na. Similarly, the program shows the correct classifications for classes Li and Na to be 200 without any misclassification. This simply implies that the data collected from the experiments have good correlation together, which helps the system to predict with lower error rate. In other words, the difference between current, mass and resistance data related to the salts are so different that makes it easy to recognize the salt without making any mistake.

By feeding the simplified tree to a shell program, the rules are transformed to C language 'if-else' statements (shown in Appendix B). In order to make this portion a complete program capable of recognizing the unknown solution, other necessary parts are also added. As shown in Appendix A, the first attribute asked for prediction of the salt would be the resistance. This is because the resistance is the root of the decision tree (Table 3 shows a typical run of the prediction

Table 2a. Part of file 'DBS.Data'.

Ca	Li	Na
Neg, −0.02, 3.65653517, 56.5660	Neg, −0.03, 13.13554933, 56.6	Neg, −0.75, 2.98555768, 41.3
Neg, −0.02, 5.01022745, 53.8781	Neg, −0.02, 13.15593024, 56.0	Neg, −0.29, 3.84823848, 41.4
Neg, −0.02, 4.88279687, 53.9664	Neg, −1.56, 6.19504433, 55.9	Neg, −0.18, 4.38710051, 41.3
Pos, 0.02, 0.04833574, 43.5298	Pos, 3.43, 7.32772446, 55.8	Pos, 0.05, 0.59349281, 45.7
Pos, 0.59, 2.80738094, 41.4694	Pos, 1.72, 6.01643363, 56.2	Pos, 0.03, 0.23138895, 45.5
Pos, 0.37, 2.44152809, 41.0051	Pos, 0.01, 0.00370819, 56.4	Pos, 0.02, 0.27683843, 45.5

Table 2b. a copy of file 'DBS.Names'.

Ca, Li, Na	Potential: Pos, Neg Current: continuous Mass: continuous Resistance: continuous

program).

If the consulting program cannot decide, based on the resistance, the user will be asked to enter a value for another attribute, which upon the case would be either current or mass. Then again,

Table 3. A typical run of the consultation program.

```
C4.5 [release 5] decision tree interpreter
Neg, −0.02, 3.70139003, 56.1, Ca
Resistance: 56.1
Mass: 3.70139003
Decision: Ca CF = 1.00 [0.98−1.00]

Retry, new case or quit [r, n, q]: r
       Pos, 0.02, 0.00319672, 0.4, Ca
       Resistance [49.6]: 0.4
       Current: 0.02
       Decision: Ca
Retry: Neg, −10.24, 0.95029402, 46.1, Ca
       Resistance [0.4]: 46.1
       Decision: Ca
Retry: Neg, −0.02, 13.08269978, 56.5, Li
       Resistance [46.1]: 56.5
       Mass [3.7139]: 13.08269978
       Decision: Li
Retry: Pos, 0.69, 4.42349005, 56.3, Li
       Resistance [56.5]: 56.3
       Mass: [13.218]: 4.42349005
       Decision: Li
Retry: Neg, −0.02, 5.10189009, 41.1, Na
       Resistance [56.3]: 41.1
       Current [0.02]: −0.02
       Decision: Na
Retry: Pos, 0.02, 0.14089601, 45.4, Na
       Resistance [41.1]: 45.4
       Current [−0.02]: 0.02
       Decision: Na
```

in the absence of a solid decision, a value for the next attribute is required. If still the program cannot make a decision, a value for applied potential must be entered to the program. At this stage, the consulting program will specify the type of the salt used as the operational environment. The number next to the name of the salt in Table 3, is the confidence rate (CF), which suggests the accuracy of the procedure. This number will only be displayed if the program cannot predict the answer for less than 99.9%.

The bases for the prediction strategy are the rules extracted from the decision tree as presented in Appendix C. The rules are constructed after the original decision tree is pruned, while each rule contains more than one branch of the decision tree. Looking at the evaluation part of Appendix C, it is clear that in this special case there is no error on the induction at all. Also, the table shows that 6 rules are related to Ca, 2 for Li, and finally 3 rules represent Na. The error column shows the probability of the error during the prediction stage.

The last part of Appendix C, again, specifies a confusion matrix for the training set (DBS.Data in this case). This matrix indicates that there is no misclassification on the data, and so the error rate for the classification (induction) is nill.

Therefore, PPy/DBS has a reliable behaviour in conjunction with the salts involved in these experiments, producing rigid, and easily recognizable patterns.

CONCLUSION

As we challenge the specific issues concerned with utilizing the dynamic attributes of conducting polymers, the tasks before us in the pursuit of intelligent sensing systems is now more obvious. Therefore, the need for a new dynamic, innovative, and intelligent processing method is realized to develop a reliable molecular communication tool.

Appreciating C4.5 machine learning method, our system, now, is capable of monitoring the dynamic nature of the polymers during the redox reaction, by collecting different attributes of the

solution in on-line/real time manner. Potentially, this platform is useable for recognition of different supporting electrolytes, if appropriate polymeric electrode has been cautiously chosen.

REFERENCES

1. Talaie A., Sadik O. and Wallace G.G., *J. Int. Mat. Sys. Struc.*, **4**, 123, 1993.
2. Talaie A., *Amir Kabir J. of Sci. and Technol.*, **7**, 26, 1994.
3. Talaie A., and Wallace G.G., *Synth. Met.*, **63**, 83, 1994.
4. Boyle A., Genies E.M. and Lapkowski M., *Synth. Met.*, **28**, C769, 1989.
5. Gustafsson G., Cao Y., Treacy G.M., Klavetter F., Colaneri N. and Heeger A.J., *Nature*, **375**, 477, 1992.
6. Talaie A., *Chemistry in Australia*, 570, December 1996.
7. Skotheim T.A. (Ed.), *Handbook f Conducting Polymers*, Marcel Dekker, New York, 1986.
8. Talaie A., and Wallace G.G., *Solid State Ionics*, **70**, 692, 1994.
9. Talaie A., *Polymer*, **38**, 1145, 1997.
10. Bruckenstein S. and Swathirajan S., *Electrochim Acta*, **30**, 851, 1985.
11. Reynolds J.R., Sundaresaan N.S., Pomerantz M., Basak S. and Baker C.K., *J. Electroanal. Chem.*, **250**, 355, 1988.
12. Naoi K., Lien M. and Smyrl W.H.J., *Electrochem. Soc.*, **138**, 440, 1991.
13. Talaie A., *5th Asian Chemical Congress (Invited Lecture)*, May 1997.
14. Deutscher R.L., Fletcher S. and Hamilton J.A., *Electrochim. Acta*, **31**, 585, 1986.
15. Talaie A., *Solid State Ionics*, **74**, 219, 1994.
16. Talaie A., in: *Solid State Ionic Materials*, Eds. B.V.R. Chodari, M. Yahaya, I.B. Talib and M.M. Salleh (World Scientific, Singapore) 335, 1994.
17. Talaie A. and Romagnoli, *J. Iran. Polym. J. (Eng. ed.)*, **6**, 1, 53, 1997.
18. Talaie A., Shahri A.M. and Talaie F., *Synth. Metals*, **79**, 63, 1996.
19. Talaie A. and Esmaili M., *Smart Matter. Struc.*, **5**, 1, 1996.
20. Quinlan J.R., C4.5: *Programs for Machine Learning*, Morgan Kaufmann, 1993.
21. Weiss S.M. and Kapouleas I., , Proceeding of the 11th International Joint Conference on Artifitcial Intelligence, Michigan, U.S.A., 781, 1990.
22. Sammut C., Michie d., et al., *Learning to Fly*, In: D. Sleeman and P. Edwards (Eds.), Proceedings of 9th International Workshop on Machine Learning, Morgan Kaufmann, 385, 1992.
23. Urbanic T. and Bratko I., *Learning to Control Dynamic Systems*, In Michie D. et al. (Eds.), Machine Learning, Neural and Statistical Classification, Ellis-Howard, 1993.
24. Esmaili N., Mann G.A., Sammut C., *Navigation Learning by a Mobile Robot*, in: Proceedings of Iran. Conf. on Elect. Eng., (ICEE-94), Tarbiat-Modarres University, Tehran, 142, 1994.

APPENDIX A

C4.5 [release 5] decision tree:

Options:

File stem <dbs>

Trees evaluated on unseen cases

Read 6010 cases (4 attributes)from dbs. data

Decision Tree

Resistance<= 55.6894:
Resistance> 45.8: Ca (820.0)
Resistance<= 45.8:
Current<= 0.01:
Resistance>41.0496: Na (1211.0)
Resistance<= 41.0496:
Potential= Pos: Ca (20.0)
Potential= Neg: Na (9.0)
Resistance < 43.5431
Current<= 0.65: Ca (1038.0)
Current> 0.65:
Current<= 1.7: Ca (12.0)
Current> 1.7: Na (13.0)
Resistance>43.5431:
Current<= 0.77: Na (760.0)
Current> 0.77:
Resistance<= 43.9: Na (7.0)
Resistance> 43.9: Ca (20.0)
Resistance> 55.6894:
Resistance<= 56.3992:
Mass<= 3.71351: Ca (67.0)
Mass> 3.71351: Li (140.0)
Resistance> 56.3992:

Resistance<= 56.4982:
Resistance<= 56.4: Li (102.0)
Resistance> 56.4: Ca (21.0)
Resistance> 56.4982:
Resistance> 56.3992: Li (1393.0)
Resistance<= 56.5824:
Mass> 7.33275: Li (359.0)
Mass<= 7.33275:
Current<= −0.02: Ca (12.0)
Current> −0.02: Li (6.0)

Simplified Decision Tree
Resistance<= 55.6894:
Resistance> 45.8: Ca (820.0/1.4)
Resistance<= 45.8:
Current<= 0.01:
Resistance> 41.0496: Na (1211.0/1.4)
Resistance<= 41.0496:
Potential= Pos: Ca (20.0/1.3)
Potential= Neg: Na (9.0/1.3)
Current>0.01:
Resistance<= 43.5431:
Current<= 0.65: Ca (1038.0/1.4)
Current> 0.65:
Current<= 1.7: Ca (12.0/1.3)
Current> 1.7: Na (13.0/1.3)
Resistance> 43.5431:
Current<= 0.77: Na (760.0/1.4)
Current> 0.77:
Resistance<= 43.9: Na (7.0/1.3)
Resistance> 43.9: Ca (20.0/1.3)
Resistance> 55.6894:
Resistance<= 56.3992:
Mass<= 3.71351: Ca (67.0/1.4)
Mass> 3.71351: Li (140.0/1.4)
Resistance> 56.3992:
Resistance> 56.5824: Li(1393.0/1.4
Resistance<= 56.5824:
Mass> 7.33275: Li (447.0/1.4)
Mass<= 7.33275:
Current<= −0.02: Ca (33.0/1.4)
Current> −0.02: Li (20.0/1.3

Evaluation on Training Data (6010 items):

Before Pruning		After Pruning		
Size	Errors	Size	Errors	Estimate
35	0(0.0%)	31	0(0.0%)	(0.4%) <<

Evaluation on Test Data (600 items):

Before Pruning		After Pruning		
Size	Errors	Size	Errors	Estimate
35	3(0.5%)	31	0(0.0%)	(0.4%) <<

(a)	(b)	(c)	<-classified as
200			(a): class Ca
	200		(b): class Li
		200	(c): class Na

APPENDIX B

C4.5 [release 5] C Module Generator

```
if (Resistance <= 55.6894) {
if (Resistance > 45.8) Ca;
else {
if (Current <= 0.01) {
if (Resistance > 41.0496) Na;
else {
if (Potential == Pos) Ca;
if (Potential == Neg) Na;
}
}
if (Current > 0.01) {
if (Resistance <= 43.5431) {
if (Current <= 0.65) Ca;
else {
if (Current <= 1.7) Ca;
else Na;
}
}
else {
if (Current <= 0.77) Na;
else {
if (Resistance <= 43.9) Na;
else Ca;
}
}
}
```

```
}
}
if (Resistance > 55.6894) {
if (Resistance <= 56.3992) {
if (Mass <= 3.71351) Ca;
else Li;
}
else {
if (Resistance <= 56.5824) Li;
else {
if (Mass > 7.33275) Li;
else {
if (Current <= -0.02) Ca;
else Li;
}
}
}
}
```

APPENDIX C

C4.5 [release 5] Rule Generator
Options:
File stem <dbs>
Read 6010 cases (4 attributes) from dbs

Final rules from tree

Rule 5
Current > 0.01
Current <= 1.7
Resistance <= 43.5431
-> class Ca [99.9%]

Rule 10
Resistance > 45.8
Resistance <= 55.6894
-> class Ca [99.8%]

Rule 1
Potential = Pos
Resistance <= 41.0496
-> class Ca [99.3%]

Rule 11
Mass <= 3.71351
Resistance > 55.6894
Resistance <= 56.3992
-> class Ca [98.0%]

Rule 9
Current > 0.77
Resistance > 43.9
Resistance <= 55.6894
-> class Ca [97.3%]

Rule 15
Current <= -0.02
Mass <= 7.33275
Resistance > 56.3992
-> class Ca [95.9%]

Rule 12
Mass > 3.71351
Resistance > 55.6894
-> class Li [99.9%]

Rule 16
Current > -0.02
Resistance > 55.6894
-> class Li [99.9%]

Rule 7
Current <= 0.77
Resistance > 43.5431
Resistance <= 45.8
-> class Na [99.9%]

Rule 2
Potential = Neg
Resistance <= 45.8
-> class Na [99.8%]

Rule 6
Current > 1.7
Resistance <= 43.5431
-> class Na [89.9%]

Default Class

Evaluation on training data (6010 items)

Rule	Size	Error	Used	Wrong	Advantage	
5	3	0.1%	1050	0(0.0%)	885(885|0)	Ca
10	2	0.0%	820	0(0.0%)	790(790|0)	Ca
1	2	0.7%	20	0(0.0%)	20(20|0)	Ca
11	3	2.0%	67	0(0.0%)	67(67|0)	Ca
9	3	2.7%	20	0(0.0%)	20(20|0)	Ca
15	3	4.1%	33	0(0.0%)	33(33|0)	Ca
12	2	0.1%	1050	0(0.0%)	1000(1000|0)	Li
16	2	0.1%	950	0(0.0%)	950(950|0)	Li
7	3	0.1%	1070	0(0.0%)	0(0|0)	Na
2	2	0.2%	910	0(0.0%)	0(0|0)	Na
6	2	10.1%	13	0(0.0%)	0(0|0)	Na

Tested 6010, errors 0 (0.0%) <<

(a)	(b)	(c)	<-classified as
2010			(a): class Ca
	2000		(b): class Li
		2000	(c): class Na