AISEC: an Artificial Immune System for E-mail Classification

Andrew Secker
Computing Laboratory
University of Kent
Canterbury, Kent
UK, CT2 7NF
ads3@kent.ac.uk

Alex A. Freitas
Computing Laboratory
University of Kent
Canterbury, Kent
UK, CT2 7NF
aaf@kent.ac.uk

Jon Timmis
Computing Laboratory
University of Kent
Canterbury, Kent
UK, CT2 7NF
jt6@kent.ac.uk

Abstract- With the increase in information on the Internet, the strive to find more effective tools for distinguishing between interesting and non-interesting material is increasing. Drawing analogies from the biological immune system, this paper presents an immune-inspired algorithm called AISEC that is capable of continuously classifying electronic mail as interesting and non-interesting without the need for re-training. Comparisons are drawn with a naïve Bayesian classifier and it is shown that the proposed system performs as well as the naïve Bayesian system and has a great potential for augmentation.

1 Introduction

Web mining is an umbrella term used to describe three quite different types of data mining, namely content mining, usage mining and structure mining (Chakrabarti, 2003). Of these, we are concerned with web content mining which is described by Linoff and Berry (2001) as "*the process of extracting useful information from the text, images and other forms of content that make up the pages*" (p. 22). The mining of textual data is a common web mining task, often for the purposes of information retrieval. This type of mining is becoming increasingly necessary as finding information on the Internet is almost impossible without automated assistance. The ultimate goal of our work is to further this area by the creation of an immune inspired tool for mining *interesting* information from the web (Liu, Ma & Yu, 2001). By adopting an immune-inspired approach we believe the final system will have the ability to dynamically determine the interestingness of a document, where interestingness may include an estimated measure of novelty or surprisingness. The algorithm described in this paper, named "AISEC" (Artificial Immune System for E-mail Classification), is a step towards such a web content mining system. AISEC is an Artificial Immune System (AIS) capable of continuous learning for the purposes of two-class classification and is illustrated here on the task of electronic mail (e-mail) sorting. In the following pages we briefly describe background immunology, web mining and our motivations for combining the two. We then describe AISEC in some detail and finally present results on a test data set and compare these with a naïve Bayesian classifier.

2 Background Immunology

For a comprehensive review of the biology and inspiration behind artificial immune systems, the reader is direct towards literature such as (Sompayrac, 1999) and (deCastro & Timmis, 2002). We present below a greatly simplified account of the immune principles pertinent to the function of our algorithm.

The natural immune system is based around a set of immune cells called *lymphocytes* comprised of *B and T-cells*. It is the manipulation of populations of these by various processes which give the system its dynamic nature. On the surface of each lymphocyte is a *receptor* and the binding of this receptor by chemical interactions to patterns presented on *antigens* which may activate this immune cell. A subset of the antigens are the *pathogens*, which are biological agents capable of harming the host (e.g. bacteria). Lymphocytes are created in the bone marrow and the shape of the receptor is determined by the use of gene libraries. These are libraries of genetic information, parts of which are concatenated with others in a semi-random fashion to code for a receptor shape almost unique to each lymphocyte. The main role of a lymphocyte in an AIS is encoding and storing a point in the solution space or *shape space* (Perelson & Oster, 1979). The match between a receptor and an antigen may not be exact and so when a binding takes place it does so with a strength called an *affinity*. If this affinity is high, the antigen is said to be within the lymphocyte's *recognition region*. As a lymphocyte may become activated by any antigen within this region a single lymphocyte may match a number of antigenic patterns, an important element of the noise tolerant nature of the immune system. When this binding takes place it stimulates an immune response from the lymphocyte and the cell begins to *clone* and *mutate*. The cloning takes place with a rate proportional to affinity and mutation with a rate inversely proportional to affinity in a process called *clonal selection*. During this process strong selective pressures seek to maximise affinity with the antigen, thus increasing the efficiency of the response. Clonal selection constitutes the core of the immune system's adaptation mechanisms. This, however, is not the whole story as a T-cell requires two signals to become activated. Signal one is a binding via its receptor to an antigenic pattern, the second signal is called *co-stimulation* and is given by an *antigen presenting cell* as a confirmation that the bound antigen really is pathogenic.

Once the pathogen has been removed a small number of clones with high affinities to the pathogen will live on to provide memory of the event. The immunological details of this process are under discussion, but this simple explanation of immune memory is of use in the artificial domain.

3 Text Mining and Web Mining

We consider the immune system particularly suitable inspiration for a web content mining algorithm (as described in the introduction) because of certain properties inherent in most immune inspired algorithms. Work in (deCastro & Timmis 2002) describes these properties and many parallel the desirable features of a web mining algorithm. Examples of these include:

1. **Pattern recognition**: The ability to recognize patterns of data similar to training examples is a common characteristic found in classification tools and of use in the web mining domain.
2. **Diversity**: Like the immune system, the Internet is diverse. It carries many different information formats, from plain text e-mails to fully animated web pages. Similarly the immune system is capable of recognising and classifying a diverse variety of invaders.
3. **Dynamically changing coverage**: The topology and content of the web is always changing. A system able to keep track and adapt to these changes can be an important feature of a web mining system.
4. **Distributivity**: The structure of the web consists of countless servers linked to countless end machines. The advantages of distributing a system over these systems are many, not just for fault tolerance but also for the possibility of parallel processing.
5. **Noise tolerance**: The natural immune system is tolerant towards noise. An AIS has the potential to filter noisy data and uncover an underlying concept.
6. **Self-organization**: Because of this little user input may be required to initially define parameters. These may automatically change to suit user preferences and changing underlying data.

This work is concerned with turning adaptive systems towards web content mining. As the number of web pages and other information on the web has grown, so has the study of techniques for mining and manipulating this information. The literature describes a vast array of systems for web content mining but one of particular interest is (Liu, Ma & Yu, 2001) as it has inspired our ultimate goal to mine interesting information from the Internet. As the scale of the Internet grows, more adaptive systems must be realized to keep pace with the accelerating change in web-based information. With such an overwhelming quantity of data available on the Internet users may suffer from information overload. Filters and search tools are a must for almost any Internet user. However, we believe in the future these tools will endeavour to become more intelligent. At present a simple keyword search on an Internet search engine may yield far more pages than a user could ever cope with. We believe a more intelligent, user-driven approach is needed such that pages a user would consider surprising, novel or unexpected are returned from a search. One further advantage of this approach is that users would have returned pages which are highly relevant to their search but not necessarily contain any of the keywords originally entered. Our ultimate goal is to mine interesting information from the web as described above but based on a current context. This context may take into account previously learned user expectations and preferences.

A literature search revealed that the area of web content mining using immune approaches are somewhat unexplored. At the time of writing one significant published investigation could be found (Twycross & Cayzer, 2003). In this, the authors detail an immune inspired concept learner turned towards the classification of HTML pages taken from the Syskill and Webert Web Page Ratings (Blake & Merz, 1998). The system appears successful, achieving a higher predictive accuracy than a Bayesian approach on most occasions. Nevertheless, it is a “one-shot” learning technique as apposed to the continuous learning system we describe here.

4 An AIS for E-mail Classification

Our chosen task is to distinguish between uninteresting e-mail and e-mail which to the user is important or interesting, based on previous experience. AISEC, is designed for use in a continuous learning scenario where the concept of what the user finds interesting will change over time and so may the content of uninteresting e-mails. An example of this is the use of the word “ca$h” where the word “cash” was once used in advertisements. We consider e-mail classification essentially a web content mining task as defined in the introduction as the text contained in the e-mail is used for the purposes of classification and an e-mail is part of the Internet environment. A further reason for this choice of testing scenario was that the problem of receiving uninteresting or junk e-mail is one faced by most who use e-mail on a day-to-day basis. It is a well understood problem with a number of references in the literature which propose solutions. Of particular interest are (Segal & Kephart, 1999) and (Crawford, Kay, & McCreath, 2002) both of which propose intelligent systems for sorting e-mail into categories. It is important to note that we are not proposing a “spam” (unsolicited bulk e-mail) filter. These are highly specialized pieces of software specially written for the task of removing mass-mail from a user’s inbox. As described in (Graham, 2003), the rules of spam filtering are somewhat different to the generic text-mining task our algorithm is designed to investigate. For example, a legitimate e-mail incorrectly classified and removed can be disastrous. These are therefore written specially to minimize the risk of such an occurrence.

4.1 The “AISEC” Algorithm

AISEC seeks to classify unknown e-mail into one of two classes based on previous experience. It does this by manipulating the populations of two sets of artificial immune cells. Each immune cell captures a number of features and behaviours from natural B-cells and T-cells but for simplicity we refer to these as B-cells throughout. These two sets consist of a set of naïve (sometimes called free) B-cells and a set of memory B-cells. Once the algorithm has been trained each B-cell represents an example of an uninteresting e-mail by containing words from that e-mail’s subject and sender fields in its feature vector. New e-mails to be classified are considered to be antigens and so to classify an e-mail it is first processed into the same format of feature vector as a B-cell and then presented to all B-cells in the algorithm. If the affinity between the antigen and any B-cell is higher than a threshold, the B-cell is said to recognise the antigen and thus classified as uninteresting. If this antigen is later confirmed by a user to represent an uninteresting e-mail, the B-cell which classified it as such is useful and is rewarded by promotion to a long-lived memory B-cell (assuming it was not already). At this time it is also selected to reproduce by clonal selection. This constant reproduction combined with appropriate cell death mechanisms are features that afford our algorithm its dynamic nature. The user feedback will be given asynchronously to classification but on a regular basis. As the algorithm is designed to address concept drift over long periods, reasonable pauses in this feedback should not cause an undue drop in classification accuracy.

During design a number of special considerations were given to the specialist nature of the text mining problem. The incorporation of these in the final algorithm served to further distance our algorithm from other AIS. These design decisions are discussed below:

Representation of one data class: In a web mining context, the number of documents a user finds interesting may be tiny compared with those a user finds uninteresting. B-cells therefore represent only the uninteresting e-mail class. A helpful and efficient simplification and more akin to the way the natural system works. Natural lymphocytes only encode possible antigenic patterns, everything else is assumed harmless.

Gene libraries: Two libraries of words, one for subject words and one for sender words are used. These contain words known to have previously been used in uninteresting e-mail. When a mutation is performed, a word from this library replaces a word from a cell’s feature vector. Mutating a word in any other way, by replacing characters for example, would result in a meaningless string in almost all cases.

Reproduction by cloning: A random generation of feature vectors as described in (Hofmeyr & Forrest, 1999) has been common but would be wholly inefficient in this application domain for the same reasons as above. Therefore all new cells entering the naïve cell set are mutants of existing cells.

Co-stimulation: E-mail classified as junk is not deleted but removed to a temporary store, interesting e-mail is delivered to the user client in the normal way (and so no longer accessible by the algorithm). B-cells must have become stimulated to classify an e-mail as junk, and therefore it is assumed the first stimulatory signal has already occurred. Feedback from a user is then interpreted to provide (or not provide) a co-stimulation signal. At a time of the user’s convenience this store may be emptied. It will be the actions of the user during this procedure that will drive a number of dynamic processes. If an e-mail is simply deleted from this store we assume the algorithm has performed a correct classification as the user really was not interested in that e-mail and so a co-stimulation signal has occurred. The cell is rewarded by being allowed to reproduce. If, on the other hand, the user does not delete the e-mail the algorithm has performed a misclassification, signal two does not occur and B-cells are removed appropriately.

Two recognition regions: Around each B-cell is a recognition region within which the affinity between this cell and an antigen is above a threshold. It is within this region an antigen may stimulate a lymphocyte. A single region was found to be inefficient for both the triggering of evolutionary processes and classification. A smaller region, a classification region, was introduced for classification only. In empirical studies the introduction of this second region was shown to increase the classification accuracy from around 80% to around 90% on a test set.

Cell death processes: To both counteract the increase in population size brought about by reproduction and keep the algorithm dynamic, cell death processes must be implemented. A naïve B-cell has not proved itself useful to the algorithm and as such is given a finite lifespan when created, although it may lengthen its life by continually recognizing new uninteresting e-mails. Memory B-cells may also die, but these cells have proved their worth and it can be hard for the algorithm to generate clones capable of performing well. For this reason, unlike naïve B-cells, memory cells are purged in a data driven manner. When a new memory cell *mc*, is added to the memory cell set all memory cells recognising *mc* have a stimulation counter reduced. When this count reaches zero they are purged from the algorithm. This dissuades the algorithm from producing an overabundance of memory cells each providing coverage over roughly the same area when a single cell is quite sufficient.

4.2 The Algorithm in Detail

Before we begin, let us establish the following notational conventions:

- Let `BC` refer to an initially empty set of naïve B-cells
- Let `MC` refer to an initially empty set of memory B-cells
- Let `Kt` refer to the initial number of memory cells generated during initialisation/training
- Let `Kl` refer to a constant which controls the rate of cloning

- Let Km refer to a constant which controls the rate of mutation
- Let Kc refer to the classification threshold
- Let Ka refer to the affinity threshold
- Let Ksb refer to the initial stimulation count for naïve B-cells
- Let Ksm refer to the initial stimulation count for memory B-cells

4.2.1 Representation

A B-cell receptor is represented as a two-part vector. One part of the vector holds words contained in the subject field of an e-mail, the second holds words contained in the sender (and return address) fields. The actual words are stored in the feature vector because once set this vector will not require updating throughout the life of the cell. This can be contrasted to the common practice of using a vector containing binary values as the receptor, each position in which represents the presence or absence of a word known to the algorithm. As words are continually being added and removed from our algorithm each cell's vector would have to be updated as appropriate when this action occurs. The two sub-vectors are unordered and of variable length. Each B-cell will also contain a stimulation counter used for aging the cell. This is initialised to Ksb on cell generation and reset to Ksm if the B-cell is later added to MC.

```
B-cell vector = <subject,sender>
subject = <word 1,word 2,...,word n>
sender = <word 1,word 2,...,word m>
```

4.2.2 Affinity Measure

The affinity between two cells is a measure of the proportion of one cell's feature vector also present in the other. It is used throughout the algorithm and is guaranteed to return a value between 0 and 1. The matching between words in a feature vector is case insensitive but otherwise requires an exact character-wise match. Given bc1 and bc2 are the cells we wish to determine the affinity between, the procedure may be outlined as follows:

```
PROCEDURE affinity (bc1, bc2)
  IF(bc1 has a shorter feature vector
    than bc2)
    bshort ← bc1, blong ← bc2
  ELSE
    bshort ← bc2, blong ← bc1
  count  ← the number of words in
               bshort present in blong
  bs_len ← the length of bshort's
               feature vector
  RETURN count/bs_len
```

Pseudocode 1. Affinity

4.2.3 Algorithms and processes

The AISEC algorithm works over two distinct stages: a training phase followed by a running phase. This running phase is further divided into two tasks, that of classifying new data and intercepting user feedback to drive evolution. An overview of this algorithm is described by Pseudocode 2.

```
PROGRAM aisec
  train(training set)
  wait until (an e-mail arrives or a
              user action is intercepted)
    ag ← convert e-mail into antigen
    IF(ag requires classification)
      classify(ag)
      IF(ag classified as uninteresting)
        move ag into user accessible
         storage
      ELSE
        allow e-mail to pass through
    IF(user has given feedback on ag)
      update_population(ag)
```

Pseudocode 2. AISEC overview

We now detail training, classification and the updating of the population based on user feedback in turn. During the training stage the goal is to populate the gene libraries, produce an initial set of memory cells from training examples, and produce naïve B-cells based on mutated training examples. As the B-cells in the AISEC algorithm represent only one class the training set, here called TE, contains only e-mails the user has explicitly selected as uninteresting.

```
PROCEDURE train(TE)
  FOREACH(te ∈ TE)
    process e-mail into a B-cell
    add subject words and sender words
     to appropriate library
  remove Kt random elements from TE
   and insert into MC
  FOREACH(mc ∈ MC)
    mc's stimulation count ← Ksm
  FOREACH(te ∈ TE)
    te's stimulation count ← Ksb
    FOREACH(mc ∈ MC)
      IF(affinity(mc,te) > Ka)
        clones ← clone_mutate(mc,te)
        FOREACH(clo ∈ clones)
        IF(affinity(clo,bc) >=
         affinity(mc,te))
          BC ← BC ∪ {clo}
```

Pseudocode 3. Training

Now the algorithm has been trained it is available to begin the classification of unknown e-mail and population manipulation processes based on user feedback. During this running phase the algorithm will wait for either a new e-mail to enter the system and so be classified or an action from the user indicating feedback. Upon receipt of either of these the necessary procedure outlined below will become invoked. To classify an e-mail, an antigen ag is created in the same form as a B-cell, taking its feature vector elements from the information in the e-mail. `ag` is then assigned a class based on pseudocode 4.

```
PROCEDURE classify(ag)
  FOREACH(bc ∈ (BC ∪ MC))
    IF(affinity(ag,bc) > Kc)
      classify ag as “uninteresting”
      RETURN
  classify ag as “interesting”
  RETURN
```

Pseudocode 4. Classification

To purge the algorithm of cells which may match interesting e-mails, the AISEC algorithm uses the two signal approach as described in section 2 of this paper. We may assume that signal one has occurred, that is the antigen generated from the classified e-mail has already stimulated a B-cell to have been classified. Signal two comes from the user in the form of interpreting the user’s reaction to this e-mail. It is during this stage that useful cells are stimulated and unstimulated cells are removed from the algorithm. `ag` is the antigen (e-mail) on which feedback has been given.

```
PROCEDURE update_population(ag)
  IF(classification was correct)
    FOREACH(bc ∈ BC)
      IF(affinity(ag,bc) > Ka)
        increment bc’s stimulation count
    bc_best ← element of BC with
      highest affinity to ag
    BC ← BC ∪ clone_mutate(bc_best,ag)
    bc_best ← element of BC with
      highest affinity to ag
    mc_best ← element of MC with
      highest affinity to ag
    IF(affinity(bc_Best,ag)>
     affinity(mc_best,ag))
      BC ← BC \{bc_best}
      bc_best’s stimulation count ← Ksm
      MC ← MC ∪ {bc_best}
      FOREACH(mc ∈ MC)
        IF(affinity(bc_best,mc) > Ka)
          decrement mc stimulation count
    add words from ag’s feature vector
       to gene libraries
  ELSE
    FOREACH(bc ∈ (MC ∪ BC))
      IF(affinity(bc,ag) > Ka)
        remove all words in bc’s feature
         vector from gene libraries
        delete bc from system
  FOREACH(bc ∈ BC)
    decrement bc’s stimulation count
  FOREACH(bc ∈ (MC ∪ BC))
    IF(bc’s stimulation count = 0)
      delete bc from system
```

Pseudocode 5. Update B-cell population

The process of *cloning and mutation* which has been used throughout this section is detailed in pseudocode 6. `bc1` is the B-cell to be cloned based on its affinity with `bc2`. `Kl` and `Km` are constants used to control the rate of cloning and mutation. The symbol $\lfloor x \rfloor$ denotes the “floor” of *x*. That is, the greatest integer smaller than or equal to the real-valued number *x*.

```
PROCEDURE clone_mutate(bc1,bc2)
  aff ← affinity(bc1,bc2)
  clones ← ∅
  num_clones ← ⌊aff * Kl⌋
  num_mutate ← ⌊ (1-aff) * bc’s feature
   vector length * Km ⌋
  DO(num_clones)TIMES
    bcx ← a copy of bc1
    DO(num_mutate)TIMES
        p ← a random point in bcx’s
         feature vector
        w ← a random word from the
         appropriate gene library
        replace word in bcx’s feature
         vector at location p with w
    bcx’s stimulation level ← Ksb
    clones ← clones ∪ {bcx}
  RETURN clones
```

Pseudocode 6. Cloning and mutation

5 Results

To determine the relative performance of AISEC, it was necessary to test it against another continuous learning algorithm. The well-known naïve Bayesian classifier was chosen as a suitable comparison algorithm, even though a fundamental assumption of the Bayesian approach, that all attributes are independent, is violated in this situation. Mitchell (1997) states, “*probabilistic approaches such as the one described here [naïve Bayesian] are among the most effective currently known to classify text documents*” (p. 180) and this technique remains very popular for classification of e-mail even today (Graham, 2003). A variation of naïve Bayesian was adapted to intercept user input in the same way as AISEC. This was implemented according to the equation taken from Mitchell (p. 177).

$$v_{NB} = \underset{v_j \in V}{\operatorname{argmax}}\, P(v_j) \prod_i P(a_i \mid v_j)$$

Where the set *V* = {junk, not junk}, $P(v_j)$ is the probability of mail belonging to class V_j and calculated based on the frequency of occurrence of class V_j in the training set. The term $P(a_i|v_j)$ is the probability of the e-mail containing word a_i given the e-mail belongs to class V_j. This probability is calculated using observed word frequencies over the training data. In this modified algorithm these observed word frequencies are updated based on user input much as in AISEC. Consideration must be given to words not yet encountered by the algorithm yet contained in an e-mail requiring classification. The probability of this unknown word occurring in either class of e-mail cannot be taken as 0, as the equation would resolve to 0. Instead, it is given a probability of occurrence of *1/k* where *k* is the total number of words known to the system.

5.1 Experimental Setup

Experiments were performed with 2268 e-mails of which 742 (32.7%) were manually classified as uninteresting, the remaining 1526 (67.3%) were assumed of some interest. Due to the unsuitability of the few publicly accessible e-mail datasets which are traditionally used for single shot learning, unlike the continuous learning scenario discussed throughout this paper, we were unable to test the algorithm on a standard e-mail dataset. All e-mails used were received between October 2002 and March 2003, and importantly their temporal ordering was preserved. Only the words contained in the subject and sender fields of the e-mail were used, but the sender information also included the return address, as these fields may differ. The fields were tokenized using spaces and the characters ".", ",", "(", ")", "!", "@", "<", ">" as delimiters and each token inserted into a separate element of the correct feature vector. Simulated user feedback was given to both algorithms after the classification of each e-mail. Throughout the algorithm a single pseudo-random number generator was used. This was an implementation of the Mersenne Twister algorithm (Matsumoto & Nishimura, 1998) written in Java by Sean Luke (Luke, 2000). During the reported runs of the AISEC algorithm, the same values for all parameters were used. These values (shown in Table 1) were arrived at by trial and error during initial verification, and as a result tend to work well over this dataset. A legal range for each parameter is also indicated.

Parameter	Value	Range
Kc (classification threshold)	0.2	0 - 1
Ka (affinity threshold)	0.5	0 - 1
Kl (clone constant)	7.0	>= 1
Km (mutation constant)	0.7	<= 1
Ksb (Naïve B-cell stimulation level)	125	> 0
Ksm (Memory cell stimulation level)	25	> 0
Kt (initial number of memory cells)	20	> 0

Table 1. Parameters

The naïve Bayesian algorithm was trained on the first 25 e-mails as both classes are required. In contrast the AISEC algorithm was trained on the first 25 junk e-mails only. The remainder were used as the continuous test set.

Unlike traditional single shot learning, where there is a fixed test set, we address continuous learning where the algorithm is continually receiving e-mails to be classified. Each time a new e-mail is classified the algorithm can use the result of this classification (the information about whether or not the class assigned was correct) to update its internal representation. This continuous learning scenario calls for a slightly different measure of accuracy to that which is normally applied. Conceptually, as there is no fixed "test set" the algorithm keeps track of its performance over the past 100 classification attempts. As each e-mail is classified an average accuracy over these previous attempts is reported. The final classification accuracy is determined by taking the mean of these values. As AISEC is non-deterministic the results presented in Table 2 are the mean values for ten independent runs using a different random seed each time. The value after the "±" symbol represents the standard deviation. Since it is deterministic, the result for the naïve Bayesian algorithm has no standard deviation associated with it as only a single run was performed.

5.2 Classification Accuracy

Algorithm	Classification Accuracy	Recall	Precision
Bayesian	88.05%	67.76%	93.93%
AISEC	89.09% ±0.97	81.13 ±4.71	82.20% ±2.63

Table 2. Predictive accuracy for continuous learning task

Table 2 summarises the results over the continuous test set. Precision is the percentage of messages classified as uninteresting really are uninteresting, and recall is the percentage of uninteresting messages classified as uninteresting. AISEC shows a better balance between these two measures. The naïve Bayesian classifier achieves a higher precision at the expense of recall. This demonstrates the naïve Bayesian classifier blocks fewer uninteresting messages, but the ones it does clock are more likely to be uninteresting and is due to a Bayesian classifier's bias towards assigning the majority class to an example. Even though, overall, AISEC yielded the slightly higher accuracy we do not claim it classifies with higher accuracy in general. Instead we believe it is reasonable to conclude that our algorithm performs with accuracy comparable to that of the naïve Bayesian algorithm but with somewhat different dynamics. The line chart Figure 1 details the predictive accuracy after the classification of each mail. This uses the accuracy measure described above and details the results for the test set from 100 classification attempts onwards. It can be seen that both algorithms are closely matched in general but there are certain areas where the changing data causes them to behave differently. Of interest are the areas 1,000 to 1,250 and 1,900 to 2,100 e-mails classified. In both situations AISEC exhibits an increase in accuracy while there is a decrease in accuracy from the naïve Bayesian algorithm. One explanation of this could be that AISEC is faster to react to sudden changes. Consider for example a word that has been very common among uninteresting e-mail. AISEC will represent this detail as the presence of this word in a number of B-cells. The Bayesian algorithm will represent this as a high frequency of occurrence in this uninteresting class compared to the other class. Consider now this word begins to be used in interesting e-mail. The AISEC algorithm will react quickly by deleting any cells containing this word that would result in a misclassification. By contrast the Bayesian algorithm will

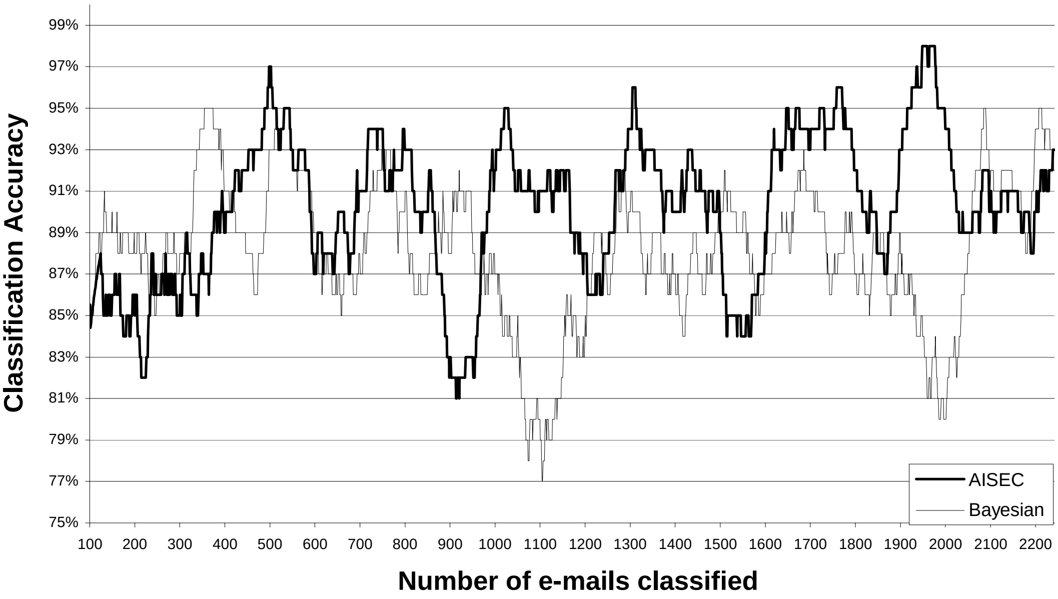

Figure 1. Change in classification accuracy by e-mails classified

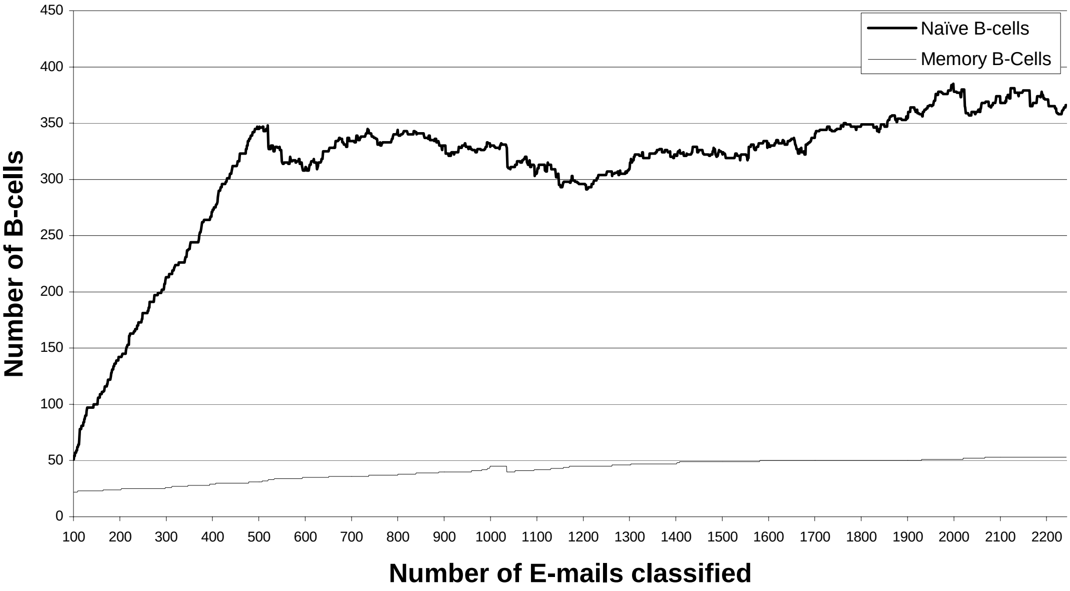

Figure 2. Change in B-cell population sizes by e-mails classified

react by only incrementing the frequency count of this word in the interesting class by one. Given the word has been common in uninteresting e-mail for some time the frequency of occurrence in this class will still be large compared with frequency of occurrence in the interesting class thus resulting in a negligible effect on the differences between the calculated final class probabilities. Only after this word has been used a number of times in confirmed interesting e-mail the differences in the usage frequencies may even out and the difference in the probabilities of this word being used in either class significantly decrease.

Experiments were also undertaken to investigate the hypothesis that AISEC would track concept drift. This was done by presenting the test data in a random order. The ordering was changed for each of ten runs. Results showed that the mean accuracy of AISEC was broadly the same as before at 88.4% while the mean accuracy of the Bayesian classifier reduced to 85.1%. This small difference in mean accuracies suggests AISEC is either not tracking drifting concepts as expected, or drifting concepts are not present in the test data. Even so, the tests did suggest AISEC is more robust than the Bayesian classifier. The accuracy of the Bayesian algorithm differed from 80% to 88% while the accuracy of AISEC stayed within ±1.3% of the mean.

5.3 Change in B-cell Population Size

Figure 2 describes the variation in size of the naïve and memory B-cell populations during the run of the algorithm. As expected there are many more naïve B-cells compared with memory cells. The number of cells in the naïve B-cell population, after an initial rapid growth period, appears fairly stable. There is an increase over the duration of the testing (348 naïve cells at 519 e-mails compared with a final value of 366 cells), but this is small relative to the size of the population. All changes appear steady but it is impossible to tell if the slight increase in numbers is due to the nature of the data rather than an underlying problem with the algorithm. On the basis of these results we are content that the process of naïve cell death after a given number of user signals is an effective control mechanism. Similarly, the memory B-cell population size is too acting broadly as hoped. There is no rapid change in the size of this cell set, as would be the case if many of its elements were subject to deletion at once. This would be evidence that the algorithm had failed in the placing of many memory cells. The memory cell population size is increasing over time, but at a decreasing rate. From this evidence it is impossible to tell if the algorithm will reach a state where the creation of new memory cells is exactly balanced by cell death, but as the population size appears to be levelling off as the number of classification attempts increases, we are again satisfied that this strategy is working as expected.

6 Conclusion

As a first step towards an artificial immune system for web mining we have described a novel immune inspired algorithm for classification of e-mail. We have shown that an immune inspired algorithm written with text mining as its primary goal may yield a classification accuracy comparable to a Bayesian approach in this continuous learning scenario. The results presented were encouraging but there are still a number of options available to optimize such a system. An increase in accuracy may be achieved by a change in the data stored in the B-cell's feature vector such as a measure of the relative importance of words using a term frequency/inverse document frequency approach. An improvement in accuracy may also be made by the use of body text from the e-mail or perhaps the use of training data to optimise the algorithm's parameters before classification begins. We certainly consider the affinity function used rather simplistic and highlight an improved affinity function, possibly in conjunction with one of the suggestions above as an area for great improvement. A longer-term project would be to hybrid this algorithm with a more traditional information retrieval technique such as a rule-based system or concept learner.

We have described some adjustments which could improve the function of this algorithm, however as described, we have some longer term goals. We feel the AISEC algorithm has shown an immune-inspired algorithm can perform text-based classification with an accuracy comparable with a naïve Bayesian approach. We now wish to forward this research with a more complex system. The ultimate goal of this work is to develop a web mining system to return web pages based on a measure of interestingness. The AISEC algorithm is step in that direction and it is hoped that continued investigation will lead us further towards this goal.

References

Blake, C. L., & Merz, C. J. (1998). *UCI Repository of machine learning databases*. Retrieved 20 May 2003, 2003, from http://www.ics.uci.edu/~mlearn/ MLRepository.html

Chakrabarti, S. (2003). *Mining the web (Discovering Knowledge from Hypertext Data)*: Morgan Kaufmann.

Crawford, E., Kay, J., & McCreath, E. (2002). *IEMS - The Intelligent Email Sorter.* In Proc. of the Nineteenth International Conference on Machine Learning (ICML 2002), Sydney, Australia.

deCastro, L. N., & Timmis, J. (2002). *Artificial Immune Systems: A New Computational Intelligence Approach*: Springer.

Graham, P. (2003). *A Plan for Spam*. Retrieved 23 April, 2003, from http://www.paulgraham.com/spam.html

Hofmeyr, S., & Forrest, S. (1999). *Immunity by Design: An Artificial Immune System.* In Proc. of the Genetic and Evolutionary Computation Conference (GECCO 1999), San Francisco, USA.

Linoff, G. S., & Berry, M. J. A. (2001). *Mining the web (Transforming Customer Data into Customer Value)*: Wiley.

Liu, B., Ma, Y., & Yu, P. S. (2001). *Discovering unexpected information from your competitors' web sites.* In Proc. of the Seventh International Conference on Knowledge Discovery and Data Mining (KDD 2001), San Francisco, USA.

Luke, S. (2000). *The Mersenne Twister in Java*. Retrieved 20 May 2003, from http://www.cs.umd.edu/users/seanl/gp/

Matsumoto, M., & Nishimura, T. (1998). Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. on Modeling and Computer Simulation, 8*(1), 3-30.

Mitchell, T. M. (1997). Bayesian Learning. In C. L. Liu & A. B. Tucker (Eds.), *Machine Learning* (pp. 154-200): McGraw-Hill.

Perelson, A. S., & Oster, G. F. (1979). Theoretical studies of clonal selection: minimal antibody repertoire size and reliability of self-non-self discrimination. *Journal of Theoretical Biology, 81*(4), 645-670.

Segal, R. B., & Kephart., J. O. (1999). MailCat: An Intelligent Assistant for Organizing E-Mail. *Third International Conference on Autonomous Agents*.

Sompayrac, L. (1999). *How the Immune System Works*: Blackwell Science.

Twycross, J., & Cayzer, S. (2003). *An Immune-based Approach to Document Classification.* In Proc. of the International Conference on Intelligent Information Processing and Web Mining 2003, Zakopane, Poland.