ANTONIO AUGUSTO CONTI FERNANDES LEÃO

CLUSTERIZAÇÃO DE DADOS USANDO ALGORITMOS IMUNOINSPIRADOS

LAVRAS - MG

2010

ANTONIO AUGUSTO CONTI FERNANDES LEÃO

CLUSTERIZAÇÃO DE DADOS USANDO ALGORITMOS IMUNOINSPIRADOS

Monografia apresentada ao Colegiado do Curso de Ciência da Computação, para a obtenção do título de Bacharel em Ciência da Computação.

Orientador

Prof. Dr. Joaquim Q. Uchôa

LAVRAS - MG

2010

ANTONIO AUGUSTO CONTI FERNANDES LEÃO

CLUSTERIZAÇÃO DE DADOS USANDO ALGORITMOS IMUNOINSPIRADOS

Monografia apresentada ao Colegiado do Curso de Ciência da Computação, para a obtenção do título de Bacharel em Ciência da Computação.

Aprovada em *16 de Novembro de 2010*

Prof. Dr. Ahmed Ali Abdalla Esmin

Prof. Dr. Cláudio Fabiano Motta Toledo

Prof. Dr. Joaquim Q. Uchôa

Orientador

LAVRAS - MG

2010

Dedico esta monografia à minha mãe, Denise

AGRADECIMENTOS

Agradeço à todos que me ajudaram, diretamente ou indiretamente na conclusão desse trabalho, pois estes tiveram a paciência de me aguentar em momentos de cansaço.

Agradeço ao meu orientador, Joaquim Q. Uchôa pelos conselhos, correções e ajuda imprescindível no trabalho.

Agradeço também a minha mãe, Denise, pelo constante suporte nessa minha etapa da vida.

De resto, aos colegas e amigos que, mesmo muitas vezes sem querer, me ajudaram muito.

RESUMO

Este trabalho tem como objetivo apresentar o CAIS (*Clustering with Artificial Imumune System*), um algoritmo inspirado no funcionamento do sistema imune para a resolução do problema da clusterização de dados. O algoritmo é baseado na teoria da rede imunológica e tem o algoritmo aiNet como inspiração. Para o seu desenvolvimento, foi utilizado o framework AISF (*Artificial Immune System Framework*), um conjunto de classes e funções em Python para a implementação de algoritmos imunoinspirados. Para testar a validade do algoritmo, ele foi avaliado utilizando-se de base de dados textuais e observando seu potencial de classificação dos dados.

Palavras-chave: Mineração de Dados, Algoritmos Imunoinspirados, Inteligência Artificial

ABSTRACT

This paper aims to present the CAIS (Clustering with Artificial Imumune System), an algoritm inspired on the immune system for the resolution of the clustering problem. The algoritm is based on Immune Network, using the aiNet algoritm as inspiration. For the development, it uses the AISF (Artificial Immune System Framework), a collection of classes and functions in Python. To evaluate the algoritm, it was tested using a text database and observing the potential of the data classification.

Keywords: Data Mining, Artificial Immune Systems, Artificial Inteligence

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 Introdução

O sistema imunológico humano é altamente distribuído, altamente adaptativo, naturalmente auto-organizado, mantendo a memória de encontros passados e tem a habilidade de continuamente aprender sobre novos encontros. Sob esse ponto de vista, o sistema imunológico tem muito o que oferecer como meio de inspiração para o desenvolvimento de algoritmos e modelos computacionais para a resolução de problemas que exigem essas habilidades.

Os Sistemas Imunológicos Artificiais (SIAs) utilizam dessa exata inspiração para modelar e aplicar princípios imunológicos no desenvolvimento de novas ferramentas computacionais. E hoje em dia isso é utilizado nas mais diferentes áreas, como por exemplo segurança computacional, reconhecimento de padrões, otimização e clusterização de dados. Porém, é uma área nova metodológica, onde o primeiro workshop conhecido foi realizado em 1996, no Japão.

O objetivo desse trabalho é o desenvolvimento de um novo algoritmo para a clusterização de dados utilizando os SIAs e tendo como base os algoritmos CLONALG e aiNet, ambos publicados em De Castro (2002). Para o desenvolvimento proposto foi utilizado o *framework* AISF (*Artificial Immune System Framework*), apresentado em Uchôa (2009).

Para validação do algoritmo, foi utilizado duas bases de dados: KDD99 (base de dados amplamente conhecida e utilizada na literatura) e *Packet Datasets* (base de dados atualizada, apresentada em Uchôa (2009)). Com isso, pode-se comparar o resultado do algoritmo proposto com outros já conhecidos.

Os principais diferenciais do algoritmo proposto para os já publicados anteriormente são: a grande quantidade de parâmetros de entrada, que permite o usuário personalizar o algoritmo afim de otimizá-lo para cada utilização; o uso de algumas informações celulares, como TTL (*Time to Live*); a maneira em que cada

cluster é criado, para que não haja dois ou mais idênticos ou muito parecidos; espaço para futuramente a utilização de mais de um tipo de linfócito, como células T.

Assim, esse trabalho multidisciplinar apresentará no Capítulo 2 alguns conceitos fundamentais de Clusterização de Dados, de Sistemas Imunes e de Sistemas Imunológicos Artificiais, apresentando os algoritmos que servirão de base. No Capítulo 3, será exposto um novo algoritmo para a clusterização de dados. No Capítulo 4 será analisado e discutido os resultados dos testes desse novo algoritmo.

2 Referencial Teórico

Para uma melhor compreensão do trabalho apresenta-se nesse capítulo os referencial teórico, relativos aos principais assuntos em estudo: Primeiramente, “Clusterização de Dados”, então “Imunologia” e posteriormente, "Imunoinformática".

2.1 Clusterização de Dados

2.1.1 O problema da Clusterização

O termo “clusterização” vem do inglês *to cluster*, que significa “agrupar”, “aglomerar”. Assim, no presente contexto, se faz o ato de agrupar dados. A clusterização de dados, ou simplesmente agrupamento de dados, segundo Cortês et al (2002) é uma técnica que visa detectar a existência de diferentes grupos (*clusters*) dentro de um determinado conjunto de dados e, em caso da existência, determinar estes grupos. Para ilustrar o problema, iremos usar um exemplo trivial: é apresentado um conjunto de dados, conforme ilustrado na Figura 1, onde será necessário separá-los em grupos. Sem muito esforço, conseguimos agrupar os dados pela cor que cada um representa. Essa facilidade se dá pelas informações visíveis relativas aos dados apresentados e também pela pequena quantidade desses.

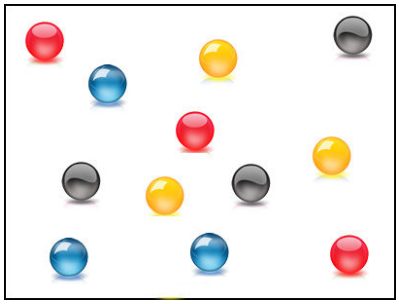

Figura 1: Exemplo de Conjunto de Dados

Porém, há situações em que os dados não contém informações tão evidentes ou conhecidas. Uma análise mais profunda seria necessária para conseguir separar os dados em grupos afins. Segundo Carlantonio (2001), a Clusterização é uma tarefa prévia à classificação, pois, sem os grupos criados, como classificar um novo objeto?

Ainda segundo este autor, a clusterização permite determinar qual o número de grupos a ser criado. E, tendo esses grupos, é possível analisar cada elemento que compõem cada grupo. Isso permite identificar as características comuns aos seus elementos e assim podendo criar um nome ou descrição que represente esse grupo, transformando-o agora em uma classe.

Após a criação dessas classes, ao recebermos um novo objeto é possível classificá-lo (alocá-lo) corretamente.

Abaixo estão algumas definições sobre clusterização, já publicadas:

- Carlantonio (2001) afirma que o objetivo da clusterização é, após recebido um conjunto de dados, de objetos, tentar agrupá-los de forma que os elementos que compõem cada grupo sejam mais parecidos entre si do que parecidos com os elementos dos outros grupos. É colocar os semelhantes juntos num mesmo grupo e os desiguais em grupos distintos.

- Hruschka e Ebecken (2001) definem clusterização como sendo uma tarefa onde se busca identificar um conjunto finito de categorias (ou *clusters*) para descrever os dados. Uma descrição genérica do objetivo da clusterização é a de maximizar a homogeneidade dentro de cada *cluster* enquanto se maximiza a heterogeneidade entre *clusters*. A tarefa da clusterização envolve a partição do conjunto de objetos em uma coleção de subconjuntos mutuamente disjuntos.

- Ankerst et al. (1999) destacam que o objetivo de descobrir *clusters* é de encontrar a partição de um banco de dados de registros tal que os registros que tem características semelhantes são agrupados juntos. Isso, então, permite que as características de cada grupo possam ser descritas.

2.1.2 Métodos de Clusterização

Hruschka e Ebecken (2001) definem alguns métodos mais gerais de clusterização: hierárquico, particionado, baseados em densidade, baseados em grades e baseados em modelos. Os métodos mais utilizados são o hierárquico e o particionado. As definições desses métodos podem ser descritas resumidamente da seguinte forma:

- O método hierárquico cria uma decomposição hierárquica da base de dados. A decomposição hierárquica é representada por um dendrograma, uma árvore que iterativamente divide a base de dados em subconjuntos menores até que cada um desses subconjunto consista de somente um objeto (ESTER et al., 1998);

- Os algoritmos de clusterização por particionamento dividem a base de dados em k grupos, onde o número k é dado pelo usuário. Os objetos são divididos entre os k *clusters* de acordo com a medida de afinidade adotada, de modo que cada objeto fique no *cluster* que forneça o menor valor de distância entre o objeto e o centro do mesmo. Então é utilizada uma estratégia iterativa de controle para determinar que objetos devem mudar de *cluster* de forma que otimizemos a função objetivo usada (CARLANTONIO, 2001).

Neste trabalho será utilizado ambos, pois será possível tanto determinar a quantidade específica de *clusters* como poderá deixar à cargo do algoritmo.

A definição matemática para o método particionado foi feita por Hruschka e Ebecken (2001):

Considerando a clusterização de um conjunto de n objetos X = {X_1, X_2, ..., X_n} , onde cada $X_i \in \Re^p$ é um vetor de p medidas reais que dimensionam as características do objeto, estes devem ser clusterizados em grupos disjuntos C = {C_1, C_2, ... ,C_k}, onde k é o número de *clusters*, de forma que tenhamos as seguintes condições respeitadas:

- $C_1 \cup C_2 \cdots \cup C_k = X$
- $C_i \neq \varnothing$
- $C_i \cap C_i = \varnothing$, para $i \neq j$

2.1.3 Aplicações

Segundo Ochi, Dias e Soares (2004), o problema de clusterização tem aplicações nas mais variadas áreas de pesquisa, sendo algumas: computação visual e gráfica, computação médica, biologia computacional, redes de comunicações, engenharia de transportes, redes de computadores e sistemas de manufatura. Além dessas, é possível também aplicar em ambientes comerciais, como em compras efetuadas em supermercados, análise química de petróleo e de solos, transações bancárias, perfil de usuários da internet, etc. Algumas exemplos foram publicados:

- Cole (1998) cita o trabalho na área de psiquiatria em que foi usado a clusterização para desenvolver uma classificação da depressão.
- Guha et al. (1999) afirmam que os *clusters* obtidos através da clusterização de uma base de dados de clientes podem ser utilizados para caracterizar diferentes grupos de clientes, podendo ser usadas em marketing direto e

permitir que produtos específicos sejam direcionados a certos clientes. As caracterizações podem ser usadas também para predizer padrões de compras de novos clientes baseado nos perfis do *cluster* a que eles pertencem.

- Pimentel, França e Omar (2003) mostram um trabalho aonde apresentam uma identificação de grupos de aprendizes no ensino presencial, utilizando técnicas de clusterização.

- Gu et al. (2008) cria o framework "Botminer" de clusterização para combater *Botnets*, utilizados normalmente para *spam*, *phishing*, *denial-of-service* (DDoS), e outros.

- Arbex (2010) mostra que, usando técnicas de mineração de dados e clusterização hierárquica, pode-se minimizar o erro da criação de grupos conforme o desempenho de um rebanho leiteiro, em relação à técnicas manuais comumente aplicadas.

2.2 Imunologia

Essa seção apresenta, resumidamente os principais conceitos sobre imunologia para o entendimento das próximas seções.

2.2.1 Introdução

Segundo de Castro (2001) a palavra imunologia é derivada do Latim *immunis* ou *immunitas* que significa "isento de carga", podendo "carga" referir-se a enfermidade. Aqueles indivíduos que não se sucumbem a uma doença, quando infectados, são referidos como imunes e uma resistência específica a uma doença é chamada de imunidade. A definição de Klein (1990) nos diz que: "A imunologia é o ramo da biologia responsável pelo estudo das reações de defesa que conferem

resistência às doença". Apesar de sua função ser à primeira vista simples, reconhecer e eliminar qualquer organismo estranho, a execução desta tarefa não é simples. O Sistema Imunológico é extremamente complexo e ainda hoje há divergências sobre o tema, como comentado em Vaz e Faria, (1993, p.1):

> Há um desconhecimento bastante difundido sobre os mecanismos de defesa imunológica. (...) Os mecanismos básicos de operação do sistema imune não são conhecidos, embora conheçamos minuciosamente a maioria de seus componentes e subcomponentes.

2.2.2 Princípios Fundamentais e Elementos Constituídos

O sistema imunológico é capaz de responder imediatamente a uma grande quantidade de patógenos sem a necessidade de uma prévia exposição dos mesmos, com barreiras mecânicas impedindo sua penetração (tecido epitelial, por exemplo), barreira bioquímicas (saliva, ácidos estomacais, etc) ou por atuação de células que efetuam o reconhecimento de padrões associados a diversos tipos de patógenos (PAMP - Pathogen-Associated Molecular Patterns), eliminando ou iniciando o processo de eliminação desse patógeno, sendo este mecanismo chamado de sistema imune inato. Já o sistema imune adaptativo produz anticorpos específicos contra cada patógeno, e por isso gasta-se um tempo maior para a sua produção (UCHÔA, 2009).

Ambos os sistemas (adaptativo e inato) dependem das células brancas (ou leucócitos), que são células produzidas na medula óssea e estão presentes no sangue, linfa, órgãos linfóides e em vários tecidos conjuntivos e podem se diferenciar em neutrófilos, eosinófilos, linfócitos e basófilos. A imunidade inata é mediada principalmente pelos macrófagos e granulócitos, enquanto a imunidade adaptativa é mediada pelos linfócitos, como mostrado na Figura 2 (DE CASTRO, 2001).

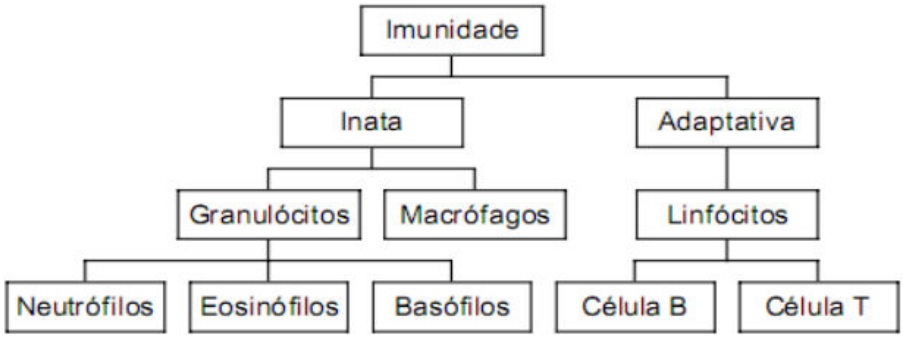

Figura 2: Mecanismos de defesa e seus principais mediadores (DE CASTRO, 2001)

No sistema imune inato, os macrófagos e neutrófilos tem a capacidade de ingerir e digerir diversos micro-organismos e partículas antigênicas. Os macrófagos também podem apresentar antígenos a outras células, sendo assim denominado de célula apresentadora de antígeno (APC - *antigen presenting cells*). Os neutrófilos são os elementos celulares mais numerosos e importantes da resposta do sistema imune inato, e também tem a capacidade de ingerir patógenos (DE CASTRO, 2001). Os eosinófilos são importantes principalmente na defesa contra infecções por parasitas, e a função dos basófilos não é bem conhecida ainda (JANEWAY et al., 2000). Há também as células NK (*Natural Killer* ou Exterminadora Natural) que são portadoras de morfologia granular e matam certas células tumorais.

No sistema imune adaptativo, os anticorpos são produzidos pelos linfócitos B (ou células B) em resposta à infecções e a sua presença em um organismo mostra infecções às quais o mesmo já foi exposto. Os linfócitos são capazes de desenvolver uma memória imunológica, ou seja, reconhecer o mesmo estímulo antigênico se ele entre novamente em contato com o organismo, evitando portanto a reincidência da doença (SPRENT, 1994; AHMED; SPRENT, 1999). Assim, a resposta imune adaptativa é melhorada a cada encontro com um antígeno.

De Castro (2001) diz que os linfócitos que participam de uma resposta imune adaptativa são responsáveis por reconhecer e eliminar os patógenos, proporcionando a imunidade duradoura, a qual pode ocorrer após a exposição a uma doença ou aplicação de vacinas. A maioria dos linfócitos encontra-se em um estado inativo e se ativarão quando houver algum tipo de interação com um estímulo de um antígeno (necessário para a ativação e proliferação dos linfócitos). Como mostrado na Figura 2, existem dois tipos de linfócitos: linfócitos B (ou células B) e linfócitos T (ou células T).

Segundo Uchôa (2009), as células B tem uma proteína globular em suas superfícies, o BCR (*B Cell Receptor* - Receptor da Célula B), conhecidos também como imunoglobulinas de membrana. Quando ativados, as células B diferenciam-se em plasmócitos, secretando essas imunoglobulinas que são liberadas como anticorpos no organismo. Uma imunoglobulina possui uma região de ligação que é responsável pelo reconhecimento de antígenos específicos, como mostrado na Figura 2. Em condições propícias, o processo de ligação ao antígeno irá estimular a célula B a se dividir, formando uma linhagem clonal dessa linhagem. Durante esse processo de clonagem, também podem ser formadas células B de memória, capazes de sobreviver por um longo tempo e possibilitar uma rápida resposta a outra infecção mesmo antígeno no organismo.

Além das células T citotóxicas e células T auxiliares, existem duas outras classes de linfócitos T para o correto funcionamento do sistema imune, cujas quais são as células T de memória e células T regulatórias (também conhecidas como células T supressoras). As células T supressoras, em geral com receptores CD4, tem a capacidade de interromper a produção de anticorpos e outras respostas imunes (DE CASTRO,2001), sendo isso essencial para a manutenção a tolerância imunológica. Já as células T de memória tem características semelhantes às da cé-

Figura 3: Ligação de Anticorpo a um Antígeno (UCHÔA, 2009)

lula B de memória: capacidade de sobreviver por um longo tempo e possibilidade de uma resposta rápida à novas exposições do antígeno ativador (UCHÔA,2009).

Enquanto a resposta imune adaptativa resulta na imunidade contra a reinfecção ao mesmo agente infectante, a resposta imune inata permanece constante ao longo da vida de um indivíduo, independente da exposição ao antígeno (SCROFERNEKER & POHLMANN, 1998).

Para uma melhor compreensão dos elementos tanto do sistema imune adaptativo quanto do sistema imune inato, é apresentada a Tabela 1, que apresenta e compara a ligação do Anticorpo ao Antígeno dos dois sistemas e a Figura 4, que representa o sistema imunológico em camadas, mostrando primeiro a barreira mecânica de proteção (pele), as barreiras bioquímicas e depois os sistemas imune inato e adpativo.

Tabela 1: Ligação de Anticorpo a um Antígeno (UCHÔA, 2009)

Propriedade	**Sistema Imune Inato**	**Sistema Imune Adaptativo**
Células	Células Dendríticas, Macrófagos, Mastócitos, Células NK	Células T, Células B
Receptores	Codificados pelo genoma, possuindo especificidade fixa, sem rearranjamento	Codificados em segmentos gênicos, com rearranjo somático
Distribuição dos Receptores	não-clonal	clonal
Reconhecimento	Padrões moleculares conservados, selecionados evolutivamente	Detalhes de estruturas moleculares, selecionados durante a vida do individuo
Resposta	Citocinas, Quimiocinas	Expansão clonal, anticorpos, citocinas
Tempo de Ação	Ativação imediata	Ativação lenta
Organismos	Vertebrados e invertebrados	Somente vertebrados

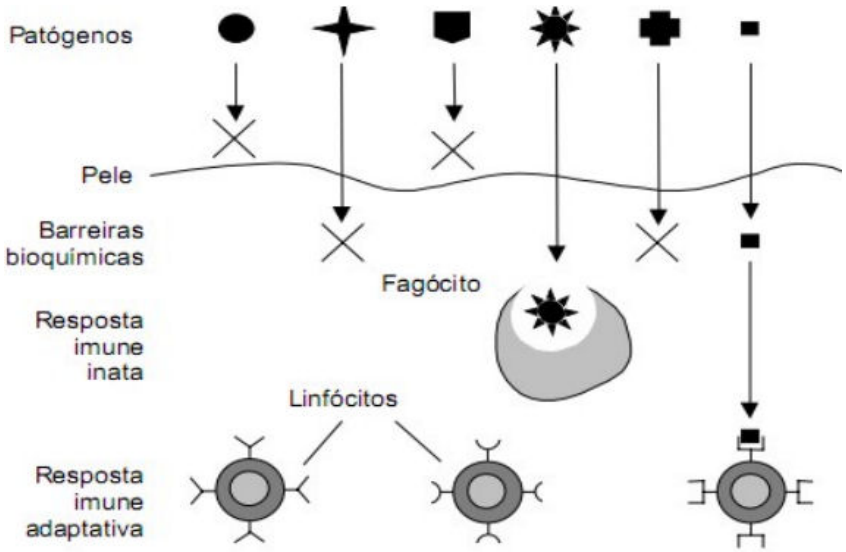

Figura 4: Estrutura multicamadas do sistema imunológico (DE CASTRO, 2001)

2.2.3 Mecanismos Fundamentais de Defesa do Sistema Imunológico

Para que o Sistema Imunológico seja ativado, é necessário que o antígeno seja reconhecido por determinados elementos. A Figura 5 representa um esquema simplificado dos principais mecanismos de reconhecimento e ativação do sistema imunológico.

Células apresentadoras de antígeno (APCs) especializadas, como macrófagos, circulam pelo corpo ingerindo e digerindo os patógenos encontrados, dividindo-os em peptídeos antigênicos (NOSSAL, 1993) (I). Partes destes peptídeos ligam-se à moléculas do complexo de histocompatibilidade principal (MHC - *major histocompatibility complex*) e são apresentados na superfície celular (II). As células T possuem receptores de superfície cuja a função é de reconhecer diferentes complexos MHC/peptídeo (III). Uma vez ativados pelo reconhecimento MHC/peptídeo, as células T se dividem e secretam linfocinas que irão mobilizar outros componentes do sistema imunológico (IV) (DE CASTRO, 2001).

Diferente dos receptores das células T, os receptores das células B são capazes de reconhecer partes livres solúveis dos antígenos, sem as moléculas do MHC (V). As células B, que também possuem moléculas receptoras com especificação única em suas superfícies, respondem a estes sinais. Quando ativas, as células B se dividem e se diferenciam em plasmócitos, secretando altas taxas de anticorpos, que são formas solúveis dos seus receptores (VI). A ligação dos anticorpos aos antígenos encontrados faz com que o patógeno seja neutralizado (VII), levando à sua destruição pelas enzimas do sistema complemento ou por fagócitos. Algumas células B e T se transformam em células de memória, as quais permanecem na circulação garantindo uma resposta rápida e com eficiência contra uma futura exposição àquele antígeno (DE CASTRO, 2001).

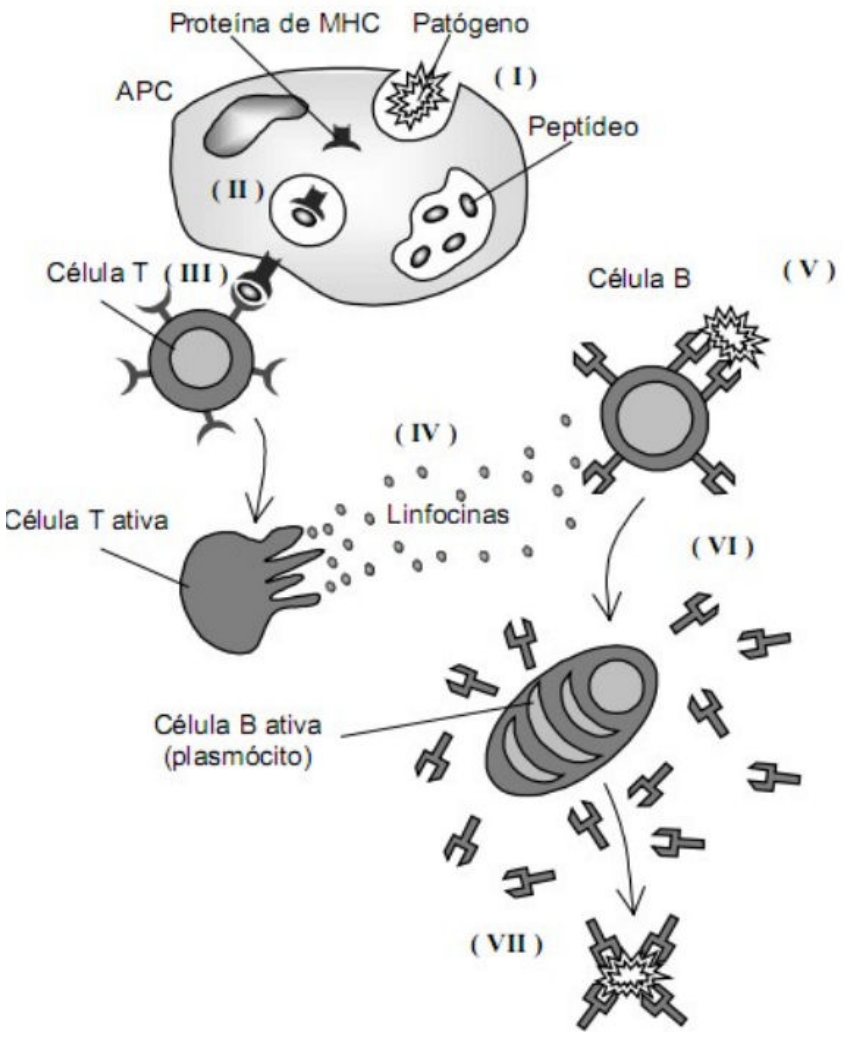

Figura 5: Esquema simplificado dos mecanismos de reconhecimento e ativação do sistema imunológico (DE CASTRO, 2001).

2.2.4 Algumas Teorias Sobre o Sistema Imunológico

Como comentado anteriormente, existem vários aspectos desconhecidos do Sistema Imunológico, o que estimula o surgimento de diversas teorias sobre o seu funcionamento. Nessa seção serão apresentadas algumas teorias sobre o seu funcionamento.

2.2.4.1 Teoria da Seleção Clonal Em Burnet (1959) foi proposto que, após a ligação do antígeno, a célula é ativada para proliferar e produzir uma numerosa progênie idêntica, conhecida como clone. Macfarlane Burnet deu assim à sua proposição o nome de teoria da seleção clonal.

Seus postulados básicos são apresentados a seguir (PINTO, 2006):

- Cada linfócito é portador de um só tipo de receptor de especificidade única;
- A interação de uma molécula estranha e um receptor de linfócitos capaz de ligar-se a essa molécula com alta afinidade leva à ativação linfocitária;
- As células efetoras diferenciadas, derivadas de um linfócito ativado, portarão receptores de especificidade idêntica à da célula parental da qual se originou o linfócito;
- Os linfócitos portadores de receptores específicos para moléculas próprias são destruídos em uma fase precoce do desenvolvimento linfóide e, assim, estão ausentes do repertório de linfócitos maduros.

Segundo Pinto (2006) ainda a grande diversidade de receptores de linfócitos significa que pelo menos haverá alguns poucos que possam se ligar a um antígeno estranho. Porém, uma vez que cada linfócito tem um receptor diferente, o número de linfócitos que podem se ligar e responder a um determinado antígeno

é pequeno. Para produzir linfócitos efetores específicos em número suficiente para combater uma infecção, um linfócito com o receptor de especificidade apropriada deve ser ativado e proliferar antes que sua progênie finalmente se diferencie em células efetoras.

2.2.4.2 Teoria da Rede Imunológica Em Jerne (1974) foi proposta uma nova teoria, chamada de Rede Imunológica. Nela o autor diz que o sistema imunológico é composto por uma rede regulada de células e moléculas que se reconhecem mesmo na ausência de antígenos. Este ponto de vista se apresentava em conflito com a teoria da seleção clonal já existente, que assumia que o sistema imunológico era composto por um conjunto discreto de clones celulares originalmente em repouso, sendo que a atividade apenas existiria quando um estímulo externo se apresentasse ao organismo. Os linfócitos poderiam responder positivamente ou negativamente ao sinal de reconhecimento. Uma resposta positiva levaria à ativação e proliferação celular, resultando na secreção de anticorpos, ao passo que uma resposta negativa levaria à tolerância e supressão (DE CASTRO, 2001). Ainda segundo o autor, podemos sintetizar a teoria da seguinte maneira:

> A característica central da teoria da rede imunológica é a definição da identidade molecular do indivíduo, pois a tolerância é uma propriedade global que não pode ser reduzida à existência ou à atividade de um clone específico. Ela surge de uma estrutura em forma de rede que se expressa no início da evolução do sistema imunológico e é seguida pela aprendizagem ontogênica da composição molecular do ambiente no qual o sistema imunológico se desenvolve. A organização em rede impõe um padrão de dinâmica para os anticorpos que é distinto das respostas imunológicas a antígenos externos. Estes padrões de dinâ-

mica são perfeitamente compatíveis com a manutenção da memória que não está localizada em células de memória, mas distribuída pela rede.

2.3 Sistemas Imunológicos Artificiais

Esse capitulo visa mostrar motivações, conceitos e alguns algoritmos imunoinspirados.

2.3.1 Introdução

Os Sistemas Imunológicos Artificiais (SIAs), em comparação com outras áreas de inteligência artificial, é uma área nova e pouco explorada, mas que apresenta uma grande quantidade de conceitos que podem servir de inspiração para a resolução de problemas computacionais das mais diversas naturezas. E como ainda acontecem descobertas constantes na área biológicas, a fontes de inspiração é constante e muito rica. Portanto, estabelecer um único algoritmo que represente o sistema imunológico é no momento uma tarefa bastante difícil, porém, há alguns que tentam representar o todo a partir de teorias da área.

Pesquisas na área da engenharia e informática procuram usar soluções propostas pelos SIAs em robótica, segurança computacional, reconhecimento de padrões, otimização, aprendizado de máquinas.

2.3.2 Algoritmos Imunoinspirados

Nessa seção será apresentado alguns dos algoritmos imunoinspirados mais utilizados e que servirão de base para esse trabalho. Lembrando que, no paradigma dos SIAs, os dados de treinamento serão transformados em linfócitos (células B

e T) e os dados a serem classificados, clusterizados, etc. são considerados os antígenos.

2.3.2.1 CLONALG Como comentado em Abbas et al (2003), cada indivíduo possui diversos anticorpos derivados de clones. Cada clone origina-se de um precursor único, capaz de reconhecer e responder um determinante antigênico distinto e, quando, o antígeno entra, seleciona um clone específico pré-existente, ativando-o. Isso vem a constituir a hipótese ou teoria de seleção clonal e, conforme apontado pelos autores, foi proposto por Niels Jerne em 1955. Porém, deve-se destacar que mesmo sendo essa teoria ser criticada e questionada por muitos autores, é uma das principais inspirações para o SIAs (UCHÔA, 2009).

O algoritmo CLONALG (CLON*al selection* ALG*orithm*) foi inicialmente proposto para resolver problemas de aprendizagem de máquina e reconhecimento de padrões (antígenos), onde uma população aleatória de anticorpos está presente e tem por objetivo aprender a reconhecer um conjunto de antígenos. Dadas suas características adaptativas, o algoritmo foi estendido para aplicações a problemas de otimização (DE CASTRO, 2001).

O algoritmo parte do princípio de que quando um linfócito reconhece um antígeno com um algum grau de afinidade (dada por uma medida de distância, como Euclidiana, por exemplo), ele tende a proliferar e gerar clones (FIGUEREDO, 2008). O pseudo-código do algoritmo pode ser visto na Figura 6

O algoritmo, como apresentado na Figura 6, representa uma implementação computacional simplificada do princípio da seleção clonal dos linfócitos B durante uma resposta imune adaptativa (Seção 2.2).

Sua complexidade computacional para o reconhecimento de padrões é a seguinte:

Algoritmo 1 *CLONALG*

- ◇ Inicializar aleatoriamente uma população de células P.
- ◇ Para cada antígeno faça:

laço

- ◇ Apresentar o antígeno para a população P e determinar a afinidade com relação a cada linfócito;
- ◇ Gere um número *Nc* de clones para as células com maior afinidade, sendo que quanto maior a afinidade, maior o número de clones;
- ◇ Mute cada clone proporcionalmente à aptidão da célula: quanto maior a afinidade, menor a taxa de mutação e vice-versa;
- ◇ Adicione as células modificadas a P e selecione as melhores para formarem a memória celular;
- ◇ Para cada clone, selecione a célula com a maior aptidão e calcule a aptidão média da população selecionada. Determine a aptidão de todos os indivíduos da população;
- ◇ Substituir células de baixa afinidade por outras geradas aleatoriamente;
- ◇ Definir um critério de parada;

fim laço

Figura 6: Pseudo-código do algoritmo Clonalg (FIGUEREDO, 2008).

- O(M(N + N_c.L)) - para o tempo de processamento
- $\propto$ M(N + L(n + N_c + N)) - para a memória

Onde N é o tamanho da população de anticorpos, N_c a quantidade de clones gerada a partir da seleção dos n melhores indivíduos, L é o comprimento das cadeias de atributos, e M a quantidade de antígenos a serem reconhecidos (DE CASTRO, 2001).

Já foram propostas diversas aplicações práticas para a resolução de problemas usando o CLONALG. Algumas dessas são:

- Reconhecimento de caracteres binários, mostrado em (DE CASTRO; VON ZUBEN, 2000a), exemplificado na Figura 7;
- Resolução de problemas de Otimização Multiobjetivo, em (BERBERT, 2008);
- "CLONALG aplicado ao problema de estimação DOA" (*Direction Of Arrival*), apresentado em (BOCCATO et al., 2009);

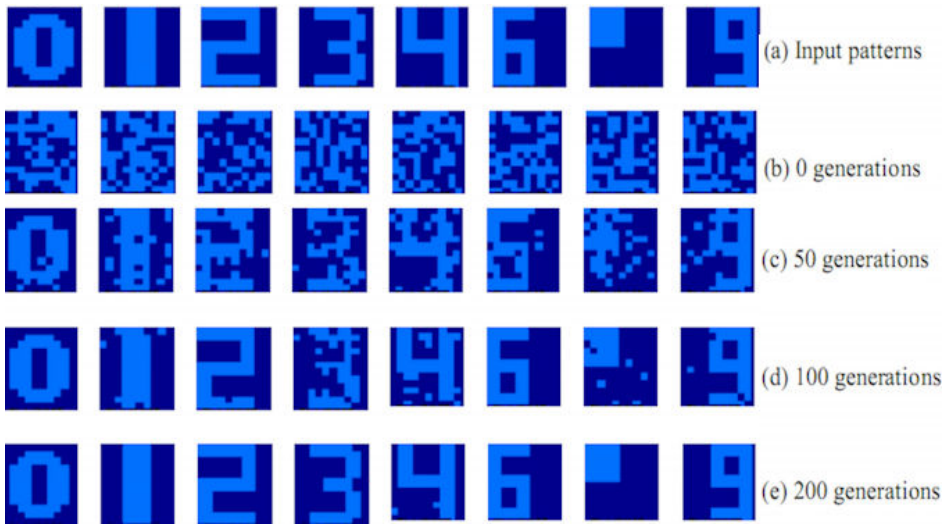

Figura 7: Reconhecimento de caracteres binários após 200 gerações (DE CASTRO; VON ZUBEN, 2000a)

2.3.2.2 aiNet O algoritmo aiNet (*Artificial Immune NETwork*) é inspirado na teoria da rede imunológica apresentada em Jerne (1974), já comentada na seção anterior deste trabalho. Originalmente proposto por de Castro & Von Zuben (2000b), no aiNet será usado o princípio da seleção clonal para controlar a quantidade e forma dos anticorpos da rede, enquanto técnicas de clusterização hierárquica e teoria de grafos serão utilizadas para definir e interpretar a estrutura final. Abaixo, uma definição sobre o algoritmo:

Definição 1. *"A rede imunológica artificial, chamada aiNet, é um grafo com conexões ponderadas, não necessariamente totalmente interconectado, composto por um conjunto de nós, denominados anticorpos, e conjuntos de pares de nós chamados conexões, com um valor característico associado, chamado de peso da conexão ou simplesmente peso." (DE CASTRO, 2001)*

Ainda segundo o autor, dado um conjunto Ag de antígenos, onde cada antígeno (padrão ou amostra de treinamento) Ag_i, i = 1,...,K, é descrito por L variáveis

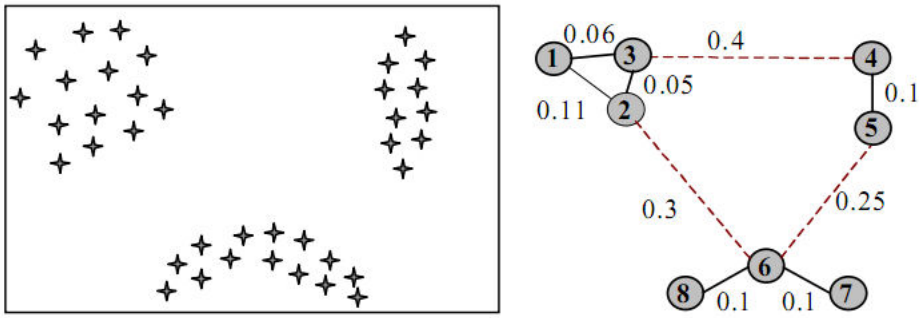

Figura 8: Ilustração do aiNet (DE CASTRO; VON ZUBEN, 2001)

(atributos) em um espaço de formas Euclidiano, uma rede imunológica artificial deverá ser construída para responder às seguintes questões: (i) Existe algum grupo ou subgrupo intrínseco aos antígenos? (ii) Se existir(em), quantos são? (iii) Quais são as propriedades relevantes destes grupos de antígenos? (iv) Como podemos gerar regras de decisão para classificar novos antígenos? (v) Qual é a conformação dos grupos no espaço de formas?

A Figura 8 exemplifica o funcionamento do aiNet. Em (a) mostra um conjunto de antígenos com 3 regiões de alta concentração antigênica (densidade de dados) e, em (b) é uma arquitetura hipotética de como o aiNet construiria a rede após o treinamento. O pseudo-código do aiNet é apresentado na Figura 9.

Segundo França (2005), de forma similar ao CLONALG, o aiNet inicia gerando uma população de anticorpos **Ab** de forma aleatória. Para cada antígeno apresentado à rede, o algoritmo calcula a afinidade do antígeno a todos os anticorpos, selecionando os n melhores e executando a clonagem seguida do processo de mutação:$C_k^* = C_k + \alpha(Ag_j - C_k)$, onde o clone k do anticorpo i é movido na direção do antígeno j com um passo a (0 < a < 1) proporcional à distância entre eles (quanto mais próximo menor o passo). Após esse processo, são selecionados os

```
[Ab{m},S] = Função aiNet(Ag,L,max_it,n,ζ,σd,σs,d);
      Ab := gera(N0,L);
      Para t = 1 .. max_it,
            Para j = 1 .. M,
                  f(j,:) = afinidade(Ab,Ag(j,:));
                  Ab{n}(j,:) = seleciona(Ab,f(j,:),n);
                  C = clona(Ab{n},1,f(j,:));
                  C* = mutação(C,Ag(j,:),f(j,:));
                  f(j,:) = afinidade (C*,Ag(j,:));
                  M = seleciona(C*,f(j,:),ζ);
                  [M,f(j,:)] = suprime(M,1/f(j,:),σd);
                  S = afinidade (M,M);
                  [M,S] = suprime(M,S,σs);
                  Ab{m} = insere(Ab{m},M);
            Fim
            S = afinidade(Ab{m},Ab{m});
            [Ab{m},S] = suprime(Ab{m},S,σs);
            Ab{d} = gera(d,L);
            Ab = insere(Ab{m},Ab{d});
      Fim
Fim
```

Figura 9: Pseudo-código do algoritmo aiNet (FRANÇA, 2005)

ς clones com menor similaridade em relação à população de anticorpos de acordo com a função de afinidade e estes são colocados em uma população de memória **M**. Em seguida, os elementos dessa população são suprimidos de duas maneiras: i) quando a afinidade deles em relação ao antígeno for menor ou igual a um limiar σ_d; ou ii) quando existirem dois elementos da população que tenham uma distância menor ou igual a um limiar σ_s, conhecido como limiar ou fator de supressão. Finalmente, os elementos que restaram entram para a população de anticorpos **Ab**. Em seguida, o segundo critério de supressão é aplicado a toda a população **Ab** e, então, *d* novos elementos são inseridos.

3 Metodologia

Esse capítulo visa mostrar a metodologia usada para se alcançar os objetivos desse trabalho, que é a proposta de um novo algoritmo para a clusterização de dados. Portanto, primeiramente o *framework* utilizado como base e, posteriormente, o algoritmo.

3.1 AISF - *Artificial Immune System Framework*

Para a implementação do algoritmo de clusterização, foi utilizado o AISF - *Artificial Immune System Framework*, uma biblioteca de classes e funções criadas na linguagem de programação Python, com suporte a um grande número de processos e elementos do sistema imune, que foi apresentado em Uchôa (2009).

Na Figura 10 é possível verificar alguma das classes do *framework*, destacando a *ImmuneSystem* como a principal, sendo a grande maioria dos métodos usados no algoritmo criado pertencentes a ela, como por exemplo a geração da população inicial de células, a apresentação dos antígenos às células, seleção dos melhores clones e atualização do tempo de vida da população.

Já na Na Figura 11 pode-se ver os tipos celulares implementados. A classe abstrata *ImmuneCell* contém a maioria dos métodos para o processamento das células como a ativação, mutação, supressão e o retorno dos dados. Entre seus elementos, merece destaque o parâmetro receptor, que é uma instância da classe *AntigenReceptor*, responsável pela etapa de ligação do antígeno à célula. Outro parâmetro importante a ser citado, herdado da classe básica *Cell*, é o TTL, que indica o tempo de vida da célula em questão (UCHÔA, 2009).

Vale ressaltar alguns recursos que o *framework* disponibiliza e que não são citados nos algoritmos propostos na Seção 2.3.2. Um deles é que cada célula carrega de informação um TTL (*Time To Live*, que representa o tempo de vida

Figura 10: AISF - Classes ImmuneSystem, Microorganism e CellPopulation (UCHÔA, 2009)

dela. A cada apresentação de um novo antígeno, o TTL de toda população celular é atualizado, sendo que as células que atingem uma afinidade mínima com esse antígeno ganham um bônus e as que não são atingem, há uma redução nesse TTL. Um segundo recurso é o fato de que os receptores das células não são gerados totalmente aleatórios. Eles são criados a partir de um trecho dos receptores de antígenos.

3.2 Algoritmo e Modelo Proposto

O algoritmo proposto, CAIS (*Clustering with Artificial Immune System*), foi essencialmente inspirado no aiNet. Isso significa que somente células B foram utilizadas. Porém, elementos diferentes do algoritmo original foram utilizadas para a escolha dos *clusters*, como por exemplo o número de ativações e o tempo de vida (TTL - *Time To Live*) das células. A seguir será descrito o algoritmo em formas mais gerais:

1. **Cria-se uma população inicial de células B**. Seus receptores utlizam-se de "sementes", que são trechos selecionados aleatoriamente da base de dados de entrada. Com isso, tem-se que o receptor dessas células é criado, intencionalmente para reconhecer um trecho da entrada. Como o receptor é menor que a entrada, há uma boa proporção de aleatoriedade.

2. **Enquanto a quantidade das células B que reconhecem os antígenos não atingir uma porcentagem determinada, os próximos passos serão executados**. Isso garantirá que no final da execução o sistema terá pelo menos uma proporção de células B que reconheça os antígenos, e isso é essencial pois será essa população que será usada para a criação dos *clusters*.

   (a) Apresente os antígenos às células B

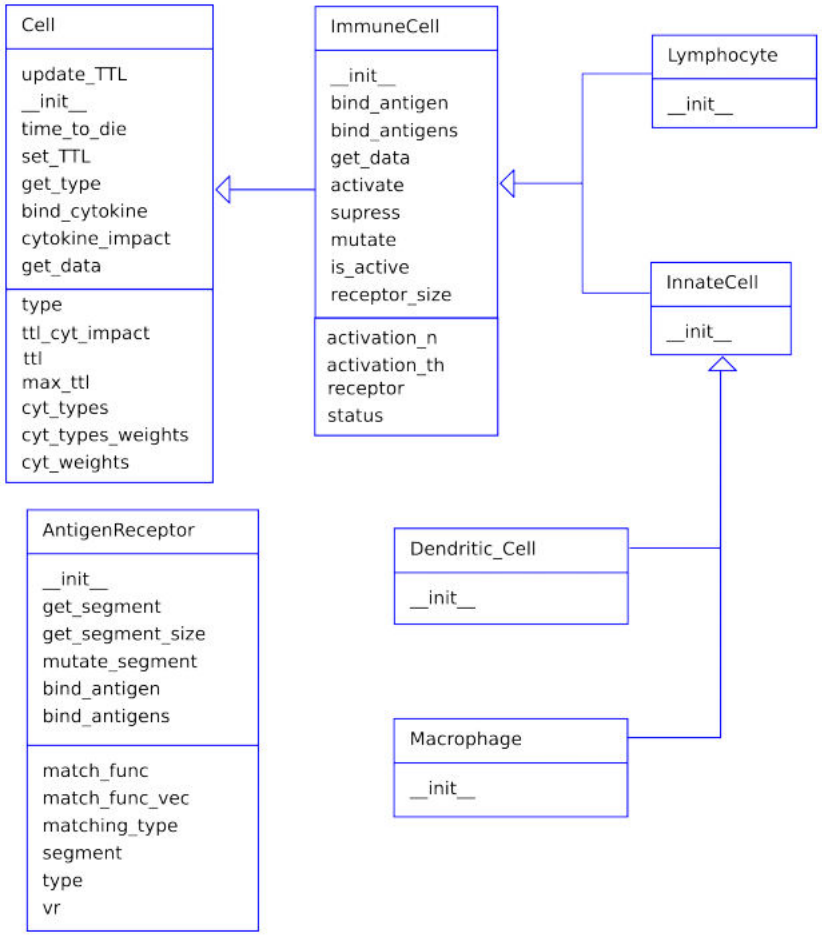

Figura 11: AISF - Classes Básicas de Células e Classes do Sistema Imune Inato (UCHÔA, 2009)

(b) Para cada célula B que atinge uma taxa mínima de ativação:

- Ative a célula
- Gere clones
- Faça mutação nos clones

(c) Verifique os clones, apresentando-os para os antígenos

(d) Se há clone que reconheça algum antígeno:

- Selecione os melhores clones
- Adicione esses melhores clones na população de células

(e) Atualize o tempo de vida de todas as células

(f) Adicione novas células à população

É importante observar que esse passos do algoritmo é fortemente inspirado no CLONALG, tendo muitos elementos semelhantes a ele.

Para o código real equivalente e esses primeiros passos, consulte Algoritmo 1.

3. **Pegue todos os receptores da população sobrevivente após o treinamento inicial e coloque em uma estrutura de dados**. Essa etapa é preparatória para o próximo passo.

4. **Para cada receptor, os próximos passos serão executados**. É aqui que inicia-se o equivalente ao aiNet. Cada receptor será apresentado para os outros receptores, para determinar se existem células parecidas uma com as outras para formar as conexões do grafo.

   (a) Apresente um receptor à população

   (b) Para cada célula B da população e a taxa de afinidade com o receptor:

- Se o receptor atual tiver uma taxa de afinidade com a célula B maior que um valor determinado:
  - Salve a célula B em uma estrutura de dados
  - Salve a taxa de afinidade em uma estrutura de dados

(c) Junte as duas estruturas criadas anteriormente em uma só.

(d) Ordene, por afinidade, essa estrutura.

(e) Para todas as células salvas nessa estrutura de dados, excluindo a com maior afinidade, elimine todas as outras.

Para o código real equivalente e esses passos intermediários, consulte Algoritmo 2. Com isso encerra-se a etapa de treinamento.

5. **Para cada item do arquivo de testes, siga os passos seguintes:** Aqui cada entrada da base de dados a ser clusterizada será colocado em um *cluster*.

   (a) Cria-se uma tupla com três campos: id do cluster, taxa de afinidade do dado de entrada com o cluster, o dado de entrada e coloque-a em uma estrutura de dados.

   (b) Apresenta-se o item a ser clusterizado para a população de células B atual.

   (c) Para cada célula B e sua taxa de afinidade com o item de teste:

   - Se a taxa de afinidade salva na tupla for menor que a taxa de afinidade com a célula B atual:
     - Atualize a tupla com id do cluster, taxa de afinidade do dado de entrada com o cluster e o dado de entrada atuais.

6. **Ordene a estrutura de dados com as tuplas por ordem de ID dos *clusters*.**

7. **Para cada ID único, imprima o dado de entrada agrupados.**

Algoritmo 1: Código representado nos passos 1 e 2

```
antigens_binded = []
num_clones = 0

cais.generate_initial_lymphocytes(cell_type.bcell, data_seeds, [])

while (float(len(set(antigens_binded)))/len(data_seeds)) <
    DATARATIO:
    [bcells_binded,antigens_binded,rate_binding] = cais.
        present_to_cell_pop(data_seeds, cell_type.bcell)
    all_clones = []
    for bcell,rate in zip(bcells_binded,rate_binding):
        bcell.activate()
        clones = cais.clone(bcell, rate)
        cais.hypermutate(clones, rate)
        all_clones += clones

    [clones_binded,antigens_c_binded,clone_binding] = cais.present
        (data_seeds, all_clones)

    if clones_binded:
        clones_selected = cais.select_best_clones(clones_binded,
            clone_binding)
        cais.add_to_cell_population(cell_type.bcell,
            clones_selected)

    cais.update_all_cells()
    cais.add_new_lymphocytes(cell_type.bcell, data_seeds, [])
```

Algoritmo 2: Código representado nos passos 3 e 4

```
cluster_data = []
aux = cais.cellpopulations[cell_type.bcell].get_cells_data()
for cell in aux:
    cluster_data.append(cell[7])

for rec in cluster_data:
    list_rate = []
    list_bcell = []
    [bcells_binded,antigens_binded,rate_binding] = cais.
        present_to_cell_pop([rec], cell_type.bcell, True)
    for bcell,rate in zip(bcells_binded,rate_binding):
        if rate >= rate_similar:
            list_bcell.append(bcell)
            list_rate.append(rate)

    bcell_similar_seg = zip(list_bcell,list_rate)
    bcell_cluster_it = sort_by_col(bcell_similar_seg,(1))

    for bcell,rate in bcell_cluster_it[0:-1]:
        bcell.time_to_die()

    cais.update_all_cells()
```

Algoritmo 3: Código representado no passo 5 a 7

```
for i in xrange(0,len(test_data)):
    q = 0
    r = 0
    tp = (0,0,test_data[i])
    classification.append(tp)
    td = [test_data[i]]
    [bcells_binded,antigens_binded,rate_binding] = cais.
        present_to_cell_pop(td, cell_type.bcell, True)
```

```
    for bcell,rate in zip(bcells_binded,rate_binding):
        if classification[i][1] < rate:
            classification.pop()
            tp = q,rate,test_data[i]
            classification.append(tp)
        q += 1

resultado = sort_by_col(classification,(0))

for i in xrange(1,len(resultado)):
    if resultado[i-1][0] != resultado[i][0]:
        print '\n--------\n' + resultado[i][2]
    else:
        print resultado[i][2] + '\n'
```

3.3 Definição dos Parâmetros

Deve-se destacar que o algoritmo aqui mostrado contém uma grande variedade de parâmetros que irão influenciar diretamente o seu desempenho. A Figura 12 apresenta um exemplo de arquivo de configuração utilizado no CAIS.

Na seção *general* são definidos os parâmetros gerais do sistema, como o nível de *debug*, os tipos celulares utilizados, número gerado de clones e taxa de mutação. A próxima seção refere-se à características particulares das células B que, no caso do CAIS, é o único tipo usado. É nela que são definidos algumas informações que merecem destaque, como o tamanho máximo e mínimo da população, a quantidade de células geradas inicialmente e também por iteração, o TTL inicial e o TTL bônus caso a célula seja ativada, a taxa mínima para que haja uma ativação celular.

```
[general]
debug= 1
window_size = 0

dcells: no
macrophages: no
bcells: yes
tcells: no
tregs: no
thelpers: no
tcyts: no

STDDEV_H = 0.0
STDDEV_S = 0.0

max_clones = 2
sel_clones = 4
mutation_factor = 0.4


# a basic cell, to be
# inherited by others
[basics]
creation_method: none
start_cells: 0

max_pop_size: 150
min_pop_size: 0
```

```
max_ttl = 100
ttl_cyt_imp = 0

cyt_weights = [[], []]

min_iter_generation = 0
max_iter_generation = 500000

[bcells]
like: basics

segment_size = 10
segment_type = 1
activation_th: 0.9
status = 0

matching_type = 0

ttl_start = 300
ttl_bonus = 2

cells_by_generation = 50
cells_by_turn = 10
```

Figura 12: Exemplo de arquivo de configuração

4 Resultados e Análise

Essa seção visa mostrar o resultado de testes efetuados, detalhando o processo para conseguir chegar nos resultados, as bases de dados e os parâmetros utilizados.

4.1 Ambiente Computacional

Para os testes descritos aqui foi utilizado um computador *desktop*, com processador AMD PHENON II x3 de 2.80ghz, 4GB de memória DDR3 e Disco 620GB SATA II, com sistema operacional Linux x64 (distribuição OpenSUSE 11.2, kernel 2.6.31). A linguagem de programação utilizada foi Python x64 2.6.5.

4.2 Base de Dados

4.2.1 Banco de Dados I - KDD99

A base de dados utilizada para a realização de testes foi a DARPA/KDD-99, disponível em (HETTICH; BAY, 1999). Dentre sua importância, vale destacar a utilização na *The Third International Knowledge Discovery and Data Mining Tools Competition*[1]. Além disso é amplamente utilizada para testes de algoritmos em aprendizado de máquinas, mineração de dados e detecção de intrusão, como visto em Eskin et. al (2002) e Lee et. al (2008).

Ela possuí mais de cinco milhões de registros com informações de tráfego de rede normais mesclados com informações de ataques computacionais (Classificados em: *back, buffer_overflow, ftp_write, guess_passwd, imap, ipsweep, land, loadmodule, multihop, neptune, nmap, perl, phf, pod, portsweep, rootkit, satan, smurf, spy, teardrop, warezclient* e *warezmaster*). Cada registro é constituído de

[1] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

```
* 0,tcp,http,SF,181,5450,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,8,8,
0.00,0.00, 0.00,0.00,1.00,0.00,0.00,9,9,1.00,0.00,0.11,0.00,0.00,
0.00,0.00,0.00,normal
* 0,tcp,ftp_data,SF,334,0,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,2,2,
0.00,0.00, 0.00,0.00,1.00,0.00,0.00,8,20,1.00,0.00,1.00,0.10,0.00,
0.00,0.00,0.00,warezclient.
* 0,icmp,ecr_i,SF,1032,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,511,511,
0.00,0.00, 0.00,0.00,1.00,0.00,0.00,255,255,1.00,0.00,1.00,0.00,
0.00, 0.00,0.00,0.00,smurf.
```

Figura 13: Exemplo de registros da KDD99

41 características, como por exemplo tipo de protocolo, duração da conexão, e serviço utilizado. A Figura 13 mostra exemplos de registros contidos na base de dados.

4.2.2 Banco de Dados II - *Packet Datasets*

Esse banco de dados, criada e publicada em Uchôa (2009) consiste em pacotes capturados e classificados em duas situações diferentes:

- Navegação normal: foram capturados pacotes provenientes de aplicações de redes usuais, como navegadores web, clientes de *e-mail*, mensageiros instantâneos, VOIP, *stream* de vídeos, dentre outros.

- Navegação anormal: foram capturados pacotes cuja origem são de aplicações que tem intuito de captura de informações não autorizadas e tentativas de intrusão. Para isso foram utilizados softwares como Nmap[2], Metasploit[3] e OpenVAS[4].

Foram coletados pacotes TCP, UDP e ICMP, e informações relevantes de cada um, como seu tamanho, tamanho do cabeçalho, portas TCP e UDP utilizadas,

[2]Nmap: http://nmap.org/
[3]Metasploit: http://www.metasploit.com/
[4]OpenVAS: http://www.openvas.org/

```
tcp,5,52,34014,2,0,64,53666,80,16,24565,0,[dados]
udp,5,28,24152,0,0,44,53110,19707,8,[dados]
icmp,5,56,18238,0,0,64,3,3,0,0,[dados]
```

Figura 14: Exemplo de pacotes capturados

flags existentes e um trecho, caso exista, da área de dados. A Figura 14 ilustra um exemplo de cada pacote.

4.3 Testes Efetuados

Vários testes foram efetuados afim de avaliar a eficácia do algoritmo. Para isso, foi dividido uma amostragem da base de dados KDD99 (contendo cerca de 494 mil registros) em 740 partes iguais de 668 registros cada. Para o treinamento das células, foi usado duas partes (totalizando 1336 registros) enquanto para clusterização, uma das partes. A Tabela 2 ilustra o procedimento descrito.

Tabela 2: Partes Utilizadas para Treinamento e Clusterização no KDD99

Teste	**Treinamento**	**Clusterização**
1	81 e 241	1
2	131 e 431	31
3	281 e 631	61

Para cada teste foi utilizado dois conjuntos de parâmetros diferentes. Os principais podem ser vistos na Tabela 3.

A Tabela 4 apresenta os resultados dos melhores índices de acerto dos testes efetuados. Nesse caso, *clusters* que continham até 2% da quantidade total dos dados de entrada foram descartados.

Para os testes no banco de dados "Packet Datasets", dividiu-se os 11.250 registros em 100 partes de 1.125 registros cada, sendo utilizadas duas partes (2.250 registros) para treinamento e uma parte para teste. A Tabela 5 ilustra o processo.

Tabela 3: Configurações Utilizados nos Testes no KDD99

Parâmetro	**Conf. 1**	**Conf. 2**
População Celular Máxima	120	120
População Celular Inicial	55	55
Células Novas por Iteração	6	3
Máx. Clones por Célula	1	1
Tam. do Receptor das Células	28	30
Limiar Ativação Celular	0.95	0.97
Limiar p/ Células serem Idênticas	0.9	0.85

Tabela 4: Resultados Obtidos no KDD99

Teste	**Config.**	**Qtde. de Clusters**	**Média de Acerto**
1	1	10	97.11%
2	1	4	85.48%
3	1	9	97.57%
1	2	7	98.03%
2	2	4	85.23%
3	2	6	86.42%

Tabela 5: Partes Utilizadas para Treinamento e Clusterização no *Packet Datasets*

Teste	**Treinamento**	**Clusterização**
1	30 e 60	1
2	40 e 70	10

Para ambos os testes foi usada o conjunto de parâmetros descrito na Tabela 6.

A Tabela 7 mostra o melhor índice de acerto dentre os testes efetuados. Nesse caso, *clusters* que continham até 2% da quantidade total dos dados de entrada foram descartados.

Tabela 6: Configurações Utilizados nos Testes do *Packet Datasets*

Parâmetro	**Valor**
População Celular Máxima	350
População Celular Inicial	230
Células Novas por Iteração	8
Máx. Clones por Célula	1
Tam. do Receptor das Células	7
Limiar Ativação Celular	0.98
Limiar p/ Células serem Idênticas	0.8

Tabela 7: Resultados Obtidos no *Packet Datasets*

Teste	**Qtde. de Clusters**	**Média de Acerto**
1	15	97.11%
2	14	97.80%

4.4 Discussão e Comentários Finais

Pelos resultados obtidos, nota-se a relevância dos parâmetros de entrada para o resultado. Por exemplo, quanto menor o limiar para células serem consideradas idênticas ou muito parecidas, menor será a quantidade de *clusters* criados, já que durante a fase de criação dos *clusters*, para que duas ou mais células pertencerem ao mesmo grupo, precisam ser idênticas ou muito parecidas. O tamanho do receptor mostrou-se também ser de grande relevância no resultado final. Além disso, a diferença na média de acerto entre os testes deve-se à divisão feita em cima dos bancos de dados que levou a criação de partes diferentes entre si.

Existe na literatura diversos testes feitos com a base de dados KDD99 usando algoritmos conhecidos, como K-NN[5], SVM[6]. Complementarmente, algoritmos que são variações dos conhecidos foram propostos e testados com boa acei-

[5]http://www.scholarpedia.org/article/K-nearest_neighbor
[6]http://en.wikipedia.org/wiki/Support_vector_machine

tação, como por exemplo o Y-Means, proposto por Guan et. al (2003). A Tabela 8 mostra alguns dos resultados de testes em cima da base de dados.

Tabela 8: Resultados Obtidos no KDD99 por algoritmos Conhecidos

Algoritmo	**Entradas Teste**	**Entradas Treino**	**Acerto Médio**
K-NN[7]	10.000	-	91%
SVM[7]	10.000	-	98%
Y-MEANS[8]	10.000	12.000	82.32%

No banco de dados *Packet Datasets* há um amplos testes feitos em Uchôa (2009). Estes testes foram feitos utilizando-se o mesmo *framework* (AISF) que este trabalho, porém usando algoritmos de classificação criados pelo autor. Vale destacar que em todos testes realizados pelo autor foram utilizados 101.250 entradas para treinamento e 1.125 para teste (classificação). A melhor média de acertos obtida por esses testes foi de 96.06%.

Comparando, o CAIS obteve resultados relevantes e comparáveis aos algoritmos citados. Apesar da quantidade menor de dados para treino e teste, foi mostrado que o algoritmo é válido e mostra-se promissora. Porém, possibilidade de melhoria está clara e será feita em trabalhos futuros.

Um recurso implementado a se destacar é que o algoritmo é capaz de interpretar qualquer base dados textuais, sejam compostas por *string*, números, etc., sendo cada linha considerada um antígeno (entrada) diferente, sendo apenas necessário ajuste de parâmetros para cada caso. Uma funcionalidade ainda em fase de implementação e testes é de poder determinar uma quantidade específica de *clusters*.

[7]Teste detalhado em Eskin et. al (2002)
[8]Teste detalhado em Guan et. al (2003)

5 Conclusão

> *Light thinks it travels faster than anything but it is wrong.*
> *No matter how fast light travels, it finds the darkness has always got there first and is waiting for it.*
>
> Terry Pratchett

Esse trabalho teve como objetivo apresentar uma solução para o problema da clusterização de dados utilizando algoritmos imunoinspirados. Para isso, foi criado o CAIS (*Clustering with Artificial Immune System*), algoritmo criado baseado em algoritmos clássicos na literatura de sistemas imunes, mas com novas opções, parâmetros e uma abordagem diferenciada. Para isso, foi utilizado o framework AISF (*Artificial Immune System Framework*), um conjunto de classes e funções em Python para a implementação de algoritmos imunoinspirados.

A maior contribuição desse trabalho foi de de apresentar um novo algoritmo com uma visão diferente da existente na maior parte dos trabalhos na área. Essa nova visão amplia o leque de trabalhos na área, servindo de inspiração para trabalhos futuros. Além disso, uma outra grande contribuição do trabalho é reforçar a visão que, mesmo sendo uma área nova, os algoritmos imunoinspirados tem um grande potencial e incontáveis aplicações.

O algoritmo proposto é funcional os resultados se mostraram promissores. Porém, algumas implementações ainda precisam ser feitas e não é difícil prever a possibilidade de otimização do código. Além disso, mais testes com variações nos parâmetros podem mostrar resultados diferentes, mais ou menos interessantes dos apresentados, exigindo que o algoritmo seja mais detalhadamente explorado antes de uso em aplicações práticas reais.

6 Referencia Bibliográfica

ABBAS,A.K., LICHTMAN,A.H., POBER,J.S. **Imunologia Celular e Molecular** 4.ed. Rio de Janeiro:Revinter,2003.

AHMED, R., SPRENT, J. **Immunological Memory**, The Immunologist, 7/1-2, pp. 23-26, 1999.

ANKERST, M., BREUNIG, M., M., KRIEGEL, H.-P., et al. **OPTICS: Ordering Points to Identify the Clustering Structure**, In: Proceedings of the ACM SIGMOD Conference on Management of Data, pp. 49-60, Philadelphia, PA, USA, June, 1999.

ARBEX, M. A. **Clusterização de Grupos Contemporâneos com Tamanho Reduzido para as Avaliações Genéticas de Rebanho Leiteiros** - UFRJ/-COPPE Rio de Janeiro, 2010.

BERBERT, P.C. **Sistema imunologico artificial para otimização multiobjetivo** Tese de mestrado - Unicamp, Campinas, 2008.

BOCCATO, L.,ATTUX, R. R. F., KRUMMENAUER, R.; LOPES, A. **Um estudo da aplicação de algoritmos bio-inspirados ao problema de estimação de direção de chegada** Sba Controle & Automação vol.20 no.4 Natal Oct./Dec. 2009

BURNET, F. M., **The Clonal Selection Theory of Acquired Immunity**, Cambridge University Press, 1959.

CARLANTONIO, L. M. **Novas Metodologias para Clusterização de Dados, Coordenação de Programas de Pós-Graduação em Engenharia**, COPPE/UFRJ, 2001.

COLE, R. M., **Clustering with Genetic Algorithms**, M. Sc., Department of Computer Science, University of Western Australia, Australia, 1998.

DE CASTRO, L. N. **Engenharia imunológica: desenvolvimento e aplicação de ferramentas computacionais inspiradas em sistemas imunológicos artificiais**, Tese de doutorado - DCA/FEEC/Unicamp, Campinas, 2001.

DE CASTRO, L. N., VON ZUBEN, F.J. (2000a) **The Clonal Selection Algorithm with Engineering Applications** Proceedings do GECCO 2000 (Genetic and Evolutionary Computation Conference)

DE CASTRO, L. N., VON ZUBEN, F. J., (2000b), **An Evolutionary Immune Network for Data Clustering** Proc. do IEEE SBRN, pp. 84-89, 2000.

DE CASTRO, L. N., VON ZUBEN, F. J., **aiNet: An Artificial Immune Network for Data Analysis** In Data Mining: A Heuristic Approach, Idea Group Publishing, USA, Março 2001.

ESKIN E., ARNOLD A., PRERAU M., PORTNOY L., STOLFO S. **A Geometric Framework for Unsupervised Anomaly Detection: Detecting intrusions in unlabeled data**. In D. Barbara and S. Jajodia, editors, Applications of DataMining in Computer Security. Kluwer, 2002.

ESTER, M., KRIEGEL, H.-P., SANDER, J., et al, **Incremental Clustering for Mining in a Data Warehousing Environment**, In: Proceedings of the 24th International Conference on Very Large Data Bases (VLDB), pp. 323-333, New York City, New York, USA, August, 1998

FRANÇA, F. O. **Algoritmos bio-inspirados aplicados à otimização dinâmica** Dissertação (Mestrado), Unicamp, 2005.

GUAN, Y., GHORBANI, ALI-AKBAR; BELACEL, N. *Y-means: A Clustering Method for Intrusion Detection* Canadian Conference on Electrical and Computer Engineering. Montréal, Québec, Canada. May 4-7, 2003.

GUHA, S., RASTOGI, R., and SHIM, K., **ROCK: A Robust Clustering Algorithm for Categorical** Attributes, In: Proceedings of the 15th International Conference on Data Engineering, pp. 512-521, Sydney, Australia, April 1999.

GU G. , PERDISCI R., ZHANG J., LEE W., **BotMiner: clustering analysis of network traffic for protocol- and structure-independent botnet detection**, Proceedings of the 17th conference on Security symposium, p.139-154, July 28-August 01, 2008, San Jose, CA

HETTICH, S.; BAY, S. D. **The UCI KDD Archive**. Irvine, CA: University of California, Department of Information and Computer Science., 1999. WWW.Disponível em: http://kdd.ics.uci.edu/.

HRUSCHKA, E. R., EBECKEN, N. F. F. **A Genetic algorithm for cluster analysis**, Submitted to: IEEE Transactions on Evolutionary Computation , January 2001.

JANEWAY, C. A.; TRAVERS, P.; WALPORT, M.; CAPRA, J. D. **Imunobiologia: O Sistema Imunológico na Saúde e na Doença**, Artes Médicas, 4a Ed. 2000.

JERNE,N.K. **Towards a Network Theory Of The Immune System** Ann. Immunol. (Inst.Pasteur), v.125C, n.1-2, p.373-89, 1974.

KLEIN, J. **Immunology**, Blackwell Scientific Publications, 2000.

LEE, J.; LEE, J.; SOHN, S.; RYU, J.; CHUNG, T. **Effective Value of Decision Tree with KDD 99 Intrusion Detection Datasets for Intrusion Detection System** in 10th International Conference on Advanced Communication Technology (ICACT), 2008.

LIFSCHITZ, S., CORTÊS S., PORCARO R. **Mineração de Dados - Funcionalidades, Técnicas e Abordagens**, ISSN 0103-9741, PUC-Rio 2002.

NOSSAL, G. J. V. **Life, Death and the Immune System**, Scientific American, 269(3), pp. 21-30, 1993.

OCHI, L. S., DIAS, C. R., SOARES, S. S. F. **Clusterização em Mineração de Dados**, Instituto de Computação - Universidade Federal Fluminense - Niterói, 2004.

PIMENTEL, E. P., FRANÇA, V.F. e OMAR, N. **A identificação de grupos de aprendizes no ensino presencial utilizando técnicas de clusterização**, XIV Simpósio Brasileiro de Informática na Educação - NCE - IM/UFRJ 2003.

PINTO, J. C. L. **Algoritmo de Detecção de Falhas para Sistemas Telefônicos Utilizando a Teoria do Perigo** Tese de Mestrado, Unicamp, 2006.

SCROFERNEKER, M. L.; POHLMANN, P. R. **Imunologia Básica e Aplicada**, Sagra Luzzatto, 1998.

SPRENT, J. **T and B Memory Cells**, Cell, 76, pp. 315-322, 1994.

UCHÔA, J. Q. **Algoritmos Imunoinspirados Aplicados em Segurança Computacional: Utilização de Algoritmos Inspirados no Sistema Imune para Detecção de Intrusos em Redes de Computadores**. Tese de doutorado - UFMG, Belo Horizonte, 2009.

VAZ, N. M.; FARIA, A. M. de; VERDOLIN, B. A.; NETO, A. F. S.; MENEZES, J. S.; CARVALHO, C. R. **The Conservative Physiology of the Immune System. Brazilian Journal of Medical Biological Research**, v. 36, n. 1, p. 13-22, 2003.