Combinando Algoritmos de Classificação para Detecção de Intrusão em Redes de Computadores

Alex L. Ramos[1], Cícero N. dos Santos[1]

[1]Mestrado em Informática Aplicada – Universidade de Fortaleza (Unifor)
Fortaleza – CE – Brasil
alex.lacerda@edu.unifor.br, cnogueira@unifor.br

Abstract. *Intrusion Detection is the process of monitoring events occurring in a network and analyzing them for signs of intrusion. The literature contains many papers that use ensemble of machine learning algorithms to solve detection problems. This paper proposes a model of detection in three levels. At each level, classification algorithms are applied and their results are combined in the later level. The last level consists of an ensemble of ensembles, which aims to improve the precision of intrusion detection. The results show that the use of three-level ensemble performs better than other systems proposed in the literature.*

Resumo. *Detecção de intrusão é o processo de monitorar e analisar eventos que ocorrem em uma rede em busca de sinais de intrusão. A literatura apresenta inúmeros trabalhos que utilizam técnicas de comitês de classificadores para resolver problemas de detecção. Este trabalho propõe um modelo de detecção em três níveis. Em cada nível são aplicados classificadores gerados por um mesmo algoritmo base e seus resultados são combinados nos níveis posteriores. O ultimo nível de classificação forma um comitê de comitês, que tenta viabilizar uma maior precisão na detecção. Os resultados apresentados demonstram que o modelo proposto apresenta melhor desempenho em relação a outros trabalhos encontrados na literatura.*

1. Introdução

Segurança de redes é definida como a proteção de sistemas de computadores contra ameaças à confidencialidade, integridade e disponibilidade das informações. Com o rápido desenvolvimento da tecnologia baseada na Internet e a dependência de negócios críticos aos sistemas de informação, novos domínios de aplicação em redes de computadores vêm surgindo [Stallings 2005]. Na medida em que as redes se desenvolvem, existe também o aumento das vulnerabilidades que podem comprometê-las. Neste contexto, a segurança da informação torna-se essencial para garantir a proteção dos dados que trafegam nestas redes.

A cada instante novos tipos de ataque surgem, tornado necessária, a criação de mecanismos de defesa mais sofisticados. Os ataques podem corromper os dados de uma aplicação e, dependendo de seu tipo e intensidade, podem até mesmo levar o sistema a um colapso. Com isso, várias medidas vêm sendo criadas para garantir a segurança contra ataques. Os mecanismos de prevenção, tais como criptografia e autenticação, são a primeira linha de defesa em uma rede, garantindo alguns princípios de segurança

como confidencialidade e integridade [Stallings 2005]. Porém, quando estas medidas não são suficientes para lidar com todos os tipos de ataque, faz-se necessário um segundo mecanismo de segurança, os Sistemas de Detecção de Intrusos (*Intrusion Detection System* - IDS) [Debar et al. 2000]. De modo geral, um IDS analisa o tráfego de rede tentando reconhecer um comportamento ou uma ação intrusiva para alertar o administrador ou automaticamente disparar contramedidas.

As técnicas utilizadas para detectar intrusões normalmente são classificadas em dois tipos: assinatura e anomalia. A detecção por assinatura compara o tráfego com uma base de dados de ataques conhecidos (assinaturas), enquanto a detecção por anomalia compara os dados coletados com registros de atividades consideradas normais no sistema. Um desvio significativo da normalidade é considerado uma ameaça. Ambas as abordagens possuem várias desvantagens. A detecção por assinatura exige atualização frequente dos registros para garantir uma boa detecção, enquanto a detecção por anomalia sofre de uma alta taxa de falsos positivos. Deste modo, o desafio é encontrar uma solução que resolva estes dois problemas, trazendo tanto uma boa detecção quanto uma baixa taxa de falsos alarmes.

Várias abordagens têm sido propostas neste sentido. Entre elas, estratégias baseadas em técnicas de aprendizado de máquina tais como Redes Neurais e Máquinas de Vetor Suporte [Mukkamala et al. 2005]. O desenvolvimento dessas técnicas obedece a duas fases distintas: treinamento e classificação. Na fase de treinamento o IDS aprende o funcionamento do sistema formando um modelo que é utilizado na fase de classificação para classificar o tráfego de rede, distinguindo atividades normais de possíveis ameaças.

Uma técnica conhecida como Comitê de Classificadores [Witten e Frank 2005] vem sendo utilizada nos trabalhos relacionados à detecção de intrusão que utilizam algoritmos de aprendizado de máquina. Métodos de comitê visam melhorar o desempenho da classificação construindo uma combinação da saída de vários classificadores, ao invés de aplicar um único modelo. O resultado da combinação é melhor do que o resultado de qualquer classificador base individual pertencente à combinação. Desta forma, técnicas de comitê trazem uma diminuição significativa das taxas de falsos positivos na detecção de intrusão, além de aumentarem a acurácia da detecção [Witten e Frank 2005].

No entanto, os Sistemas de Detecção de Intrusão baseados em Comitês de classificadores propostos recentemente [Chou et al. 2009] [Zainal et al. 2008] ainda apresentam consideráveis taxas de falsos positivos. Por esse motivo, com o intuito de explorar melhor todo o potencial desta técnica, propomos neste trabalho um modelo de comitê de classificadores que segue uma estratégia de classificação em três níveis. No primeiro nível, classificadores são gerados usando algoritmos simples, que não usam estratégias de comitês. No segundo nível, são usadas estratégias de comitê tomando como algoritmos base os que aparecem no primeiro nível. Os modelos de classificação gerados no nível dois são então combinados em um terceiro nível, formando assim um comitê de comitês. O principal propósito do trabalho é abordar a questão da acurácia e da taxa de alarmes falsos em um IDS.

O restante do artigo está organizado como segue. A Seção 2 discute os trabalhos relacionados à detecção de intrusão que utilizam técnicas de comitê. A Seção 3

apresenta a abordagem proposta. Na Seção 4, os algoritmos utilizados nos experimentos são descritos. Na Seção 5, a base de dados utilizada é apresentada. A Seção 6 discute os resultados. Por fim, a Seção 7 apresenta as conclusões do trabalho.

2. Trabalhos Relacionados

Várias abordagens baseadas em aprendizado de máquina como Redes Neurais Artificiais *(*ANNs*)*, Sistemas de Inferência *Fuzzy* e SVM (Support Vector Machine), foram propostas na literatura para realizar detecção de intrusão, como o Polvo-IIDS que realiza detecções através da utilização de um sistema multi-camadas baseado em redes neurais e SVM [Mafra et al. 2008]. Uma revisão a fundo de várias técnicas de detecção baseada em anomalia, para identificação de diferentes tipos intrusões em redes, é apresentada em [Lazarevic et al. 2003].

A literatura sugere que a combinação de múltiplos classificadores pode melhorar a acurácia da detecção. Porém é importante entender que os algoritmos combinados devem ser independentes entre si. Se os classificadores base apresentarem métodos de classificação similares, então nenhuma melhoria significativa pode ser obtida por meio da utilização de comitês. Portanto, a diversificação dos classificadores base é crítica para efetividade dos métodos de comitê [Chou et al. 2009].

No trabalho descrito por [Mukkamala et al. 2005], foram propostas três variantes de Redes Neurais, SVM e MARS como componentes de seu IDS. Esta abordagem de comitê demonstrou melhor desempenho quando comparado à abordagem de um classificador único. Porém, neste trabalho, apenas a acurácia da classificação foi apresentada, desconsiderando-se importantes medidas padrão para avaliação, tais como DR (Detection Rate – Taxa de Detecção) e FPR (False Positive Rate – Taxa de Falsos Positivos).

Na proposta de [Abraham et al. 2007], os autores também demonstraram a capacidade de sua estrutura de comitê em modelar um IDS baseado em programação genética. Entretanto, eles realizaram experimentos em uma base com dados selecionados aleatoriamente a partir da base de dados original e os resultados apresentados foram coletados de apenas um experimento, o que torna a classificação enviesada, dependendo somente das conexões que foram selecionadas para compor as bases de treino e teste.

O trabalho de [Borji 2007] é outro exemplo do uso de técnicas comitê. Ele usou o KDDCUP'99 [Lee et al.1999] para classificar suas conexões em cinco classes (*Normal*, *DoS*, *Probe*, U2R e R2L). Primeiramente, ele utilizou quatro classificadores base (Redes Neurais, SVM, K-NN e Árvores de Decisão) para realizar a classificação individualmente e então combinou suas inferências usando três estratégias de comitê: *Belief Function*, *Average Rule* e *Majority Voting* [Kuncheva 2004]. No entanto, ele não apresentou DR e FPR em cada uma das cinco classes, além de ter realizado somente um experimento em uma base com registros selecionados aleatoriamente.

Em [Zainal et al. 2008], os autores propuseram um conjunto de classificadores onde cada um usa diferentes paradigmas de aprendizagem. As técnicas utilizadas no modelo de comitê são: *Linear Genetic Programming* (LGP), *Adaptative Neural Fuzzy Inference System* (ANFIS) e *Random Forest* (RF). A partir dos resultados obtidos por

estes classificadores, a equação de combinação foi formulada. Apesar de essa estratégia ter apresentado uma melhoria na acurácia da detecção, os autores não deixam claro qual versão da base de dados KDDCUP'99 (completa, 10%, treino, teste) foi utilizada. Também não especificam como realizaram a seleção das conexões que compõem a amostra utilizada nos experimentos. Além disso, somente um experimento foi realizado nos dados selecionados aleatoriamente, tornando os resultados enviesados. Outra questão, é que não apresentaram um modelo sistemático que possa justificar a escolha dos pesos utilizados na regra de combinação.

Outro exemplo da utilização de técnicas de comitê na detecção de intrusão é o trabalho de [Chou et al. 2009]. Eles apresentam um multi-classificador com hierarquia de três camadas. Em sua proposta, primeiramente são aplicados três classificadores em cada um dos três grupos diferentes de atributos da base de dados escolhida. Na segunda camada, para cada grupo de atributos, as inferências obtidas pelos diferentes classificadores são combinadas. Por fim, os resultados das combinações de cada grupo são reunidos para produzir uma conclusão final na terceira camada. No entanto, a amostra da base de dados KDDCUP'99 que eles utilizaram para realizar os experimentos não mantém a mesma proporção das classes de ataque da base original. Isto é, a amostra utilizada não possui a mesma distribuição de classes da base original.

O presente trabalho difere dos citados anteriormente em dois aspectos principais: (a) neste trabalho, um mesmo algoritmo base é utilizado para gerar os diferentes classificadores. (b) este trabalho propõe um modelo de comitê em que as técnicas de classificação são aplicadas em três níveis, sendo que o terceiro nível gera um comitê de comitês.

3. Abordagem Proposta

Tarefas de classificação são realizadas em duas etapas conhecidas como fase de treinamento e fase de classificação. Primeiramente, um algoritmo de aprendizado de máquina é aplicado em uma base de dados (fase de treinamento) para gerar um modelo capaz de classificar os registros de uma segunda base (fase de classificação).

Neste trabalho, a detecção de ataques é realizada a partir da classificação dos registros de uma base de dados de conexões TCP/IP, no qual cada conexão é classificada como sendo normal ou pertencendo a algum tipo de ataque.

Este trabalho apresenta uma proposta de detecção de ataques que usa três níveis de classificação. No *nível 1*, a classificação é realizada por modelos gerados por um mesmo algoritmo base. No *nível 2*, um algoritmo de comitê é aplicado a vários modelos do classificador do *nível 1*. Finalmente no *nível 3*, os resultados do *nível 2* são combinados por um segundo algoritmo de comitê, gerando um comitê de comitês. A Figura 1 ilustra o modelo proposto. Observe que, em cada nível, os resultados de classificadores do nível anterior são combinados para gerar uma classificação mais precisa.

A principal proposta do trabalho é verificar se comitês de comitês podem produzir melhores sistemas de detecção de intrusão em redes de computadores. A escolha por três níveis deu-se por três motivos: (1) para formar um comitê de comitês, precisa-se de pelo menos três níveis, (2) de acordo com os experimentos realizados, a

utilização de mais de três níveis não apresenta melhorias no desempenho da detecção, pelo contrário, chega até a reduzi-lo e (3) o custo computacional da adição de um quarto nível seria muito grande. Com isso, apenas o resultado dos três níveis propostos são apresentados neste artigo.

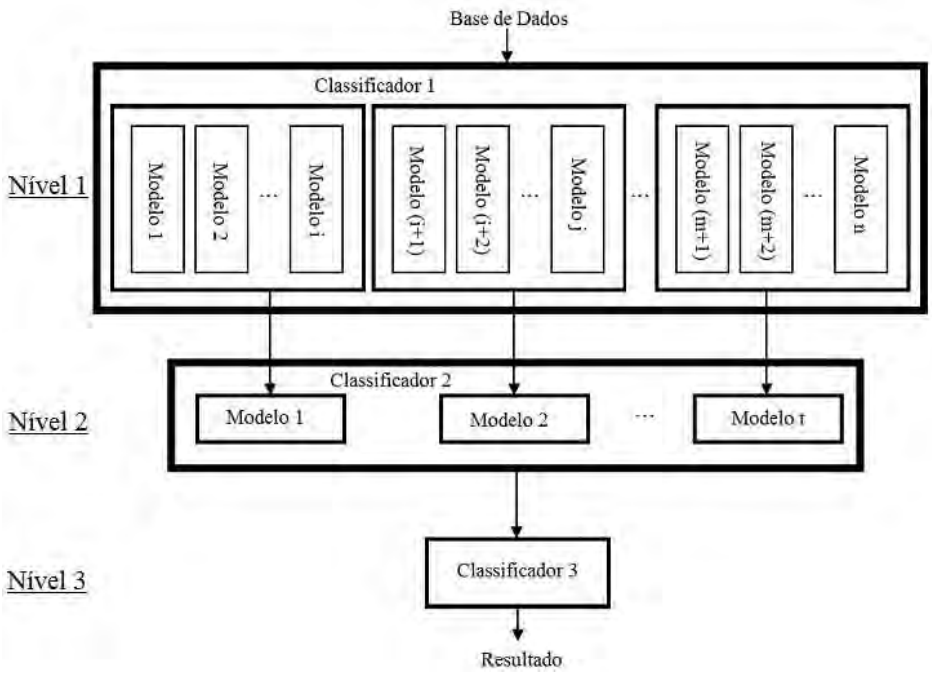

Figura 1. Abordagem de Classificação em três níveis

4. Algoritmos de Aprendizado de Máquina Aplicados à Detecção de Intrusão

Para avaliar a abordagem proposta, foram testadas três configurações de comitês de classificadores, como apresentado na Tabela 1. Com o objetivo de analisar o desempenho de vários algoritmos com paradigmas de aprendizado distintos, cada configuração utiliza três algoritmos diferentes. Os algoritmos selecionados foram os que apresentaram melhores desempenhos em relação a outros algoritmos do mesmo paradigma de aprendizado. Por exemplo, o algoritmo base *Random Tree* [Geurts et al. 2006] apresentou o melhor desempenho em relação a outros algoritmos base que utilizam árvores de decisão. Da mesma forma, *Naive Bayes* apresentou o melhor desempenho em relação a outros algoritmos que utilizam o Teorema de *Bayes* [John e Langlay 1995].

Tabela 1. Algoritmos avaliados em cada configuração de comitê

	Configuração 1	**Configuração 2**	**Configuração 3**
Nível 1	*Random Tree*	*J48*	*Naive Bayes*
Nível 2	*Random Forest*	*Bagging*	*Dagging*
Nível 3	*Random Committee*	*AdaBoost.M1*	*MultiBoostAB*

A ferramenta WEKA (*Waikato Environment for Knowledge Analysis*) [Hall et al. 2009], versão 3.6.3, foi utilizada para aplicação dos algoritmos. Escolhemos essa ferramenta por ser amplamente utilizada em atividades de aprendizado de máquina [Zai-an et al. 2010].

A seguir, os algoritmos utilizados em cada configuração são descritos com mais detalhes, de forma a destacar suas principais diferenças. Visto que, por restrições de espaço, não é possível discuti-los minunciosamente.

4.1. Configuração 1

Essa configuração foi criada utilizando apenas algoritmos randômicos, descritos a seguir:

1. **Algoritmo base:** *Random Tree* – este classificador é uma árvore de decisão que considera apenas alguns atributos escolhidos aleatoriamente para cada nó da árvore.
2. **Algoritmo de comitê:** *Random Forest* [Breiman 2001] – este algoritmo de comitê é um conjunto de árvores de classificação. Cada árvore dá um voto que indica sua decisão sobre a classe do objeto. A classe com o maior número de votos é escolhida para o objeto.
3. **Algoritmo de comitê de comitês:** *Random Committee* [Lira et al. 2007] – ele utiliza classificadores que tem funcionamento aleatório como base. Cada modelo de classificação gerado é construído usando uma semente de número aleatório diferente (mas baseado nos mesmos dados). A previsão final é uma média das previsões geradas pelos modelos base individuais.

4.2. Configuração 2

Os algoritmos utilizados nesta configuração são os seguintes:

1. **Algoritmo base:** J48 – este algoritmo é uma implementação em Java, na ferramenta WEKA, do classificador C4.5 [Quinlan 1993]. Ele gera árvores de decisão usando uma metodologia de informação teórica. O algoritmo C4.5 usa estratégia de dividir e conquistar.
2. **Algoritmo de comitê:** *Bagging* – o ***B****ootstrap* ***agg****regat****ing*** [Breiman 1996] pode ser descrito da seguinte maneira: dado uma base de dados, ele gera várias outras bases a partir da inicial, duplicando alguns registros e excluindo outros. Então, os modelos gerados a partir de cada nova base são combinados por votação, deste modo, para cada registro a ser classificado, a classe mais votada é escolhida.
3. **Algoritmo de comitê de comitês:** *AdaBoost.M1* – ele é uma das versões do algoritmo AdaBoost [Freund e Schapire 1996]. Este classificador funciona executando repetidamente um determinado algoritmo base em várias distribuições sobre a base de treino e, então, combina os classificadores produzidos pelo algoritmo base em um classificador único composto.

4.3. Configuração 3

A última configuração avaliada é formada pelos seguintes algoritmos:

1. **Algoritmo base:** *Naive Bayes* – este classificador é baseado na teoria da probabilidade condicional para executar a decisão de um problema de classificação. Ele usa o Teorema de *Bayes* com suposições de independência, o que pressupõe que, dado uma classe, conjuntos de características são condicionalmente independentes uns dos outros.
2. **Algoritmo de comitê:** *Dagging* [Ting e Witten 1997] – ele cria um número de partes estratificadas disjuntas a partir dos dados de entrada e alimenta cada bloco de dados com uma cópia do classificador base fornecido. As previsões são feitas via *Majority Voting*. Este algoritmo é bastante parecido com o *Bagging*, porém ao invés de utilizar a técnica de *bootstrapping*, ele utiliza a técnica de disjunção.
3. **Algoritmo de comitê de comitês:** *MultiBoostAB* [Webb 2000] – ele é uma extensão à técnica *AdaBoost* para a formação de comitês de decisão. Este algoritmo pode ser visto como a combinação entre *AdaBoost* e *Wagging* [Webb 2000]. Ele tem a capacidade de tirar proveito tanto do forte viés do *AdaBoost* quanto da redução de variância do *Wagging*.

5. Base de Dados para Detecção de Intrusão

A base de dados escolhida para aplicação dos algoritmos de classificação foi a DARPA KDDCUP'99. Ela é uma das poucas bases de dados de tráfego de rede disponíveis publicamente, devido a questões de legalidade, privacidade e segurança, como discutido em [Paxson 2007]. Apesar de ter sido criada a mais de dez anos, ela é a base mais utilizada para testar Sistemas de Detecção de Intrusão baseados em anomalia (Anomaly-based Intrusion Detection Systems) [Tavallaee et al. 2009]. Isso permite que nossa proposta seja comparada a outros trabalhos.

Ela foi concebida através da simulação de um ambiente de uma rede militar da força aérea dos Estados Unidos (*U.S. Air Force*). A rede foi operada em um ambiente real, alimentada por conexões TCP *dump*, mas sendo bombardeada por uma sequência de múltiplos ataques. Para cada conexão (sequência de pacotes TCP) foram extraídos 41 atributos adicionados de um rótulo que identifica se a conexão é do tipo normal ou um tipo de ataque, como mostrado em [Elkan 2000]. Os tipos de ataque desta base de dados são agrupados nas seguintes categorias:

- *Probe*: Nessa classe, os ataques se caracterizam por varrer a rede automaticamente a procura informações ou vulnerabilidades a serem exploradas. Ex.: *port scanning e port sweep*.
- *DoS* (*Denial of Service*): Também chamado de ataque de negação de serviço, se caracteriza por deixar um serviço ou rede parada ou muito lenta, Ex.*: ping-of-death e SYN flood.*
- U2R (*User to root*): Essa classe de ataques se caracteriza por iniciar o ataque como um usuário normal no sistema e explorar vulnerabilidades para ganhar acesso de usuário root. Ex.: ataques *buffer overflow*.

- R2L (*Remote to Local*): Chamado de ataque de usuário remoto, essa classe se caracteriza pelo envio de pacotes a uma máquina de uma rede, a partir daí são exploradas vulnerabilidades da máquina para ganhar acesso ilegal de usuário local. Ex.: *guessing password*.

A base de dados utilizada corresponde a 10% da base KDDCUP'99. Porém, alguns ajustes foram feitos. As alterações são semelhantes às realizadas por [Borji 2007].

Primeiramente, foram removidos os registros redundantes, que são uma das maiores deficiências desta base de dados por tornarem os algoritmos de classificação enviesados em relação aos registros frequentes [Tavallaee et al. 2009].

Em seguida, 11.982 registros foram selecionados aleatoriamente para compor as bases de treino e teste [Mukkamala et al. 2005]. O número de registros selecionados de cada classe é proporcional ao seu tamanho na base sem registros redundantes, com exceção da classe U2L que foi completamente incluída. A Tabela 2 apresenta a quantidade de registros por classe após o pré-processamento realizado.

Tabela 2. Quantidade de registros por classe

Base	Normal	Probe	DoS	U2R	R2L	Total
Original (10%)	97.278	4.107	391.458	52	1.126	**494.021**
Sem registros redundantes	87.832	2.131	54.572	52	999	**145.586**
Com registros aleatórios	7.200	175	4.473	52	82	**11.982**

Um número de 6.890 registros da base total (11.982) foi selecionado aleatoriamente para formar a base de teste e o resto (5.092) foi utilizado como base de treino, como descrito em [Mukkamala et al. 2005].

Por fim, tipos de ataque (como *buffer overflow*, *guessing password*, etc.) foram mapeados para uma das cinco classes possíveis (0 para *Normal*, 1 para *DoS*, 2 para *Probe*, 3 para R2L, 4 para U2R), como descrito em [Elkan 2000], de modo que a tarefa de classificação pudesse ser realizada.

6. Resultados e Discussão

Os experimentos consistem em várias sessões de treinamento e teste. Na fase de treinamento, os classificadores são construídos utilizando a base de treino. Em seguida, os dados de teste são introduzidos em cada classificador treinado, gerando uma classificação para cada fluxo de teste.

Os valores de parâmetros padrão da ferramenta WEKA são utilizados para configuração de cada algoritmo. A única exceção é o algoritmo *Naive Bayes*, que foi configurado para utilizar *Kernel Estimator* para atributos numéricos, ao invés de uma distribuição normal. Esta alteração foi realizada por trazer diferenças significativas aos resultados deste classificador.

O desempenho da tarefa de detecção de intrusão foi avaliado por meio de medidas padrão, tais como a taxa de detecção (DR – *Detection Rate*) e a taxa de falsos

positivos (FPR – *False Positive Rate*). Estas medidas são calculadas com as seguintes equações:

$$DR = \frac{TP}{TP + FN} \times 100\% \quad (1) \qquad FPR = \frac{FP}{FP + TN} \times 100\% \quad (2)$$

Onde, TP (*True Positive*) é a quantidade de conexões classificadas como ataques que realmente são ataques. FN (*False Negative*) é a quantidade de conexões classificadas como normais, quando na verdade são ataques. FP (*False Positive*) é a quantidade de conexões normais que são classificadas como ataque. TN (*True Negative*) é a quantidade de conexões classificadas como normais que são realmente normais. Mais detalhes sobre essas medidas podem ser encontradas em [Osareh e Shadgar 2008].

Devido à seleção aleatória dos registros das bases de dados testadas, dez iterações de treino-teste foram executadas para cada algoritmo. Isso minimiza o fator de imprecisão e variação dos resultados obtidos nos experimentos. Para cada algoritmo o resultado apresentado é a média dos resultados obtidos nas dez iterações. É importante ressaltar que todos os algoritmos apresentaram desvios padrão inferiores a 0,5 para DR e FPR, os quais podem ser considerados pequenos, visto que os valores de DR e FPR variam entre zero e 100. Isso nos permite concluir que a média é bastante representativa em relação aos resultados obtidos.

A Tabela 3 apresenta o desempenho dos três algoritmos do *nível 1*, aplicados individualmente na base de dados. Os resultados mostram que o algoritmo *Random Tree* apresenta o melhor desempenho, possuindo a maior taxa de detecção (DR) e a menor taxa de falsos positivos (FPR). Isso se deve ao fato de que o *Random Tree* se trata de uma árvore com base aleatória, capaz de obter bons resultados em uma base com uma grande quantidade de atributos, como é o caso do KDDCUP'99. O classificador *Naive Bayes* obteve os piores resultados, estando bastante distante dos outros algoritmos. Isso provavelmente se deve ao fato deste algoritmo não ser adequado para bases com grande quantidade de atributos devido à sua *suposição de independência* (*independence assumption)* [Rish 2001].

Tabela 3. Desempenho dos classificadores do *nível 1*

	Random Tree		**J48**		**Naive Bayes**	
Classes	DR	FPR	DR	FPR	DR	FPR
Normal	99,53	0,87	99,58	1,10	99,20	5,65
Probe	91,70	0,07	89,88	0,10	47,28	0,10
DoS	99,66	0,26	99,63	0,19	97,25	0,44
U2R	70,94	0,07	64,31	0,12	41,69	0,08
R2L	81,36	0,11	76,41	0,09	25,82	0,31
Total	**99,25**	**0,61**	**99,16**	**0,73**	**97,00**	**3,58**

Ainda na Tabela 3, pode-se observar que todos os algoritmos obtém melhores resultados nas classes *Normal*, *Probe e DoS*. Isso ocorre porque essas classes possuem mais registros, portanto fornecem mais informações durante a formação do modelo de

cada algoritmo. Além disso, é difícil definir características que identifiquem bem os ataques do tipo U2R e R2L, portanto os atributos do KDDCUP'99 não favorecem a classificação destes dois tipos de ataque [Lee et al.1999].

A Tabela 4 apresenta o desempenho dos classificadores no *nível 2*, cada um usando um algoritmo base diferente, como mostrado na Tabela 1. Observe que todos os algoritmos de comitê apresentam melhores resultados do que os algoritmos do *nível 1*. O algoritmo *Random Forest* apresenta o melhor desempenho para ambas as DR e FPR. Já o algoritmo *Dagging* não foi capaz de prover uma melhora significativa na taxa de detecção do *Naive Bayes*, porém foi capaz de reduzir a taxa de falsos positivos consideravelmente. Estes resultados sugerem que algoritmos de comitê são a melhor abordagem para prover alta taxa de detecção e baixa taxa de falsos positivos. Isto acontece, devido à função complementar de cada modelo utilizado no comitê, pois a aleatoriedade gerada pelos classificadores de comitê para cada modelo os torna significantemente diferentes uns dos outros [Witten e Frank 2005].

Tabela 4. Desempenho dos três classificadores do *nível 2*

	Random Forest		***Bagging***		***Dagging***	
Classes	DR	FPR	DR	FPR	DR	FPR
Normal	99,88	0,66	99,71	1,04	98,43	4,59
Probe	93,06	0,00	91,68	0,04	60,83	0,09
DoS	99,89	0,13	99,68	0,18	97,66	0,48
U2R	79,39	0,02	63,57	0,07	26,10	0,01
R2L	82,04	0,03	76,16	0,06	53,57	0,75
Total	**99,59**	**0,44**	**99,30**	**0,69**	**97,03**	**2,95**

Na Tabela 5, temos o resultado dos algoritmos do nível 3 (aplicados aos algoritmos do nível 2). É possível observar que o algoritmo *Random Committee* obteve o melhor desempenho. Nota-se ainda que, todos os algoritmos do nível 3, melhoraram os resultados do nível dois, mostrando que é interessante acrescentar mais um nível de comitê à classificação.

Tabela 5. Desempenho dos três classificadores do *nível 3*

	Rand. Committee		***AdaBoost.M1***		***MultiBoostAB***	
Classes	DR	FPR	DR	FPR	DR	FPR
Normal	99,90	0,47	99,82	0,53	98,62	3,81
Probe	95,24	0,00	97,50	0,00	69,72	0,06
DoS	99,95	0,07	99,89	0,07	98,01	0,39
U2R	82,86	0,03	80,81	0,05	46,11	0,08
R2L	87,26	0,02	85,46	0,02	55,96	0,62
Total	**99,70**	**0,31**	**99,64**	**0,36**	**97,47**	**2,43**

A Tabela 6 apresenta o percentual de melhoria (aumento para DR e queda para FPR) alcançado após a aplicação dos classificadores de comitê. No *nível 2*, o algoritmo *Random Forest* apresentou os melhores índices de melhoria (aumento de 0,34% para DR e queda de 27,87% para FPR em relação ao *nível 1*). Já no *nível 3*, o algoritmo *MultiBoostAB* foi o que mais aperfeiçoou a taxa de detecção (aumento 0,45% em relação ao *nível 2*) e o AdaBoostM.1 foi o que obteve melhor ganho em relação à taxa de falsos positivos (queda de 47,83% em relação ao *nível 2*). Observe ainda que o *nível 3* apresentou os melhores índices de melhoria, demonstrando que utilizar um comitê de comitês é bastante vantajoso para a tarefa em questão.

Tabela 6. Percentual de melhoria no desempenho total em cada nível de comitês em relação ao anterior

	Nível 2			**Nível 3**		
Medida	***Random Forest***	***Bagging***	***Dagging***	***Random Committee***	***AdaBoost. M1***	***MultiBoost AB***
DR – Taxa de aumento	**0,34 %**	0,14 %	0,03 %	0,11 %	0,34 %	**0,45 %**
FPR – Taxa de queda	**27,80 %**	5,48 %	17,60 %	29,55 %	**47,83 %**	17,63 %

No entanto, a utilização do terceiro nível traz uma desvantagem em relação ao custo computacional, pois seu tempo de treinamento é maior do que nos outros dois níveis. Entretanto, esse custo adicional pode ser compensado para sistemas em que a segurança é crítica. Além disso, a fase de treinamento é realizada apenas na parte inicial da tarefa de detecção. Portanto, após o modelo de detecção ter sido criado na fase de treinamento, as detecções seguintes são realizadas de maneira mais rápida, com tempo de processamento próximo ao dos outros níveis.

Na Tabela 7, é apresentado um comparativo entre os algoritmos de comitê abordados neste trabalho e os métodos de combinação apresentados na proposta de [Borji 2007]. O desempenho dos métodos abordados por [Borji 2007] são próximos aos obtidos neste trabalho, com uma diferença elevada apenas no *MultiBoostAB* que mesmo melhorando o desempenho do *Naive Bayes*, não foi capaz de mostrar resultados comparáveis aos demais algoritmos. Entre todos os métodos utilizados, o algoritmo *Random Committee*, abordado neste trabalho, apresentou o melhor desempenho, obtendo resultados melhores que o método *Belief Function*, principalmente em relação à taxa de falsos positivos.

Tabela 7. Comparativo com resultados obtidos por [Borji 2007]

	Classificação em três níveis			**Borji**		
Medida	***Random Committee***	***AdaBoost. M1***	***MultiBoost AB***	***Majority Voting***	***Bayesian Average***	***Belief Function***
DR	**99,70**	99,64	97,47	99,18	99,33	**99,68**
FPR	**0,31**	0,36	2,47	1,20	1,03	**0.87**

Apesar da taxa de detecção do método *Belief Function* estar bastante próxima do *Random Committee*, os resultados obtidos neste trabalho são mais precisos, visto que

foram calculados a partir da média de dez experimentos, de modo a diminuir a chance de se utilizar uma base de dados enviesada. No trabalho de [Borji 2007] não é especificado se foi realizado mais de um experimento com bases aleatórias diferentes ou se apenas uma foi utilizada. Também não foram apresentadas DR e FPR para cada classe de detecção.

A Tabela 8 mostra o desempenho do melhor resultado obtido pelo modelo em três níveis comparado aos melhores resultados dos trabalhos de [Abraham et al. 2007] e [Zainal et al. 2008]. O *Random Committee* apresenta melhor DR para as classes normal e DoS. Já a proposta de [Zainal et al. 2008] apresenta melhor FPR. Observe que as propostas dos outros autores não apresentam DR e FPR para a detecção da base de dados total, apenas para cada classe. Além disso, as taxas de falsos positivos de [Abraham et al. 2007] não são mostradas porque ele as calculou utilizando uma outra fórmula, portanto não podem ser comparadas.

É importante ressaltar que os resultados apresentados por [Abraham et al. 2007] e [Zainal et al. 2008] se baseiam em apenas um experimento, diferente da nossa proposta, que foi baseada em dez experimentos, sendo, portanto, mais confiável.

Tabela 8. Comparativo de detecção com outras propostas

	Rand. Committee		***Abraham et al.***		***Zainal et al.***	
Classes	DR	FPR	DR	FPR	DR	FPR
Normal	**99,90**	0,47	99,60	-	99,71	0,29
Probe	95,24	0,00	**99,90**	-	99,14	0,00
DoS	**99,95**	0,07	91,80	-	97,43	0,00
U2R	82,86	0,03	43,70	-	**88,00**	0,00
R2L	87,26	0,02	**98,90**	-	98,58	0,00

7. Considerações Finais

A utilização de técnicas automáticas de detecção por anomalia reduz ou elimina a necessidade de intervenção humana, tornado o sistema capaz de analisar o tráfego de redes em busca de ataques, de maneira muito mais rápida e precisa. Os trabalhos recentes de detecção apresentam a importância da utilização de comitês de classificadores para aumentar o desempenho da detecção de intrusão.

Este trabalho apresenta um modelo de classificação em três níveis, que realiza um comitê de comitês de classificadores. Os experimentos realizados demonstram que esse modelo em três níveis apresenta melhores resultados do que (1) a aplicação individual de algoritmos e (2) aplicação de apenas um nível de comitê. Quando comparado com outras propostas, o modelo mostrou-se superior em vários aspectos. No entanto, é importante notar que a aplicação de um terceiro nível de classificação exige uma maior quantidade de processamento, aumentando o tempo para realizar a classificação. Isso pode tornar o modelo inviável para bases de dados muito grandes. Entretanto, o modelo é adequado para sistemas que requerem alto nível de precisão na detecção ou que possuem uma quantidade média de dados a serem analisados.

Como trabalhos futuros, é interessante investigar um modelo capaz de melhorar o desempenho da detecção para as classes de ataque R2L e U2L. Também é importante testar a metodologia proposta em outras bases de dados para avaliar a sua robustez ou até mesmo em uma base de dados gerada a partir de tráfego real, como sugerido por [Paxson 2007], visto que os tipos de ataque de hoje diferem dos existentes na base KDDKUP'99.

Referências

Abraham, A., Grosan, C. and Vide, C. M. (2007). Evolutionary Design of Intrusion Detection Programs. In International Journal of Network Security, pages 328-339.

Borji, A. (2007). Combining Heterogeneous Classifiers for Network Intrusion Detection. In Lecture Notes in Computer Science, Volume 4846, pages 254-260. Springer.

Breiman, L. (1996). Bagging Predictors. In Machine Learning 24(3), pages 123–140.

Breiman, L. (2001). Random Forests. In Journal of Machine Learning, Vol.45, pages 5-32. Kluwer Academic, Netherland.

Chou, T. , Fan, J., Fan, S. and Makki, K. (2009). Ensemble of machine learning algorithms for intrusion detection. In Systems, Man and Cybernetic, pages 3976-3980.

Debar, H., Dacier, M. and Wespi, A. (2000).A Revised Taxonomy for Intrusion Detection Systems. Annals of Telecommunications, pages 361-378.

Elkan, C. (2000). Results of the KDD'99 Classifier Learning. In SIGKDD Explorations, ACM SIGKDD.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In Thirteenth International Conference on Machine Learning, pages 148-156.

Geurts, P., Ernst, D. and Wehenkel, L. (2006). Extremely randomized trees. In Machine Learning, Vol. 63, pages 3-42.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, I. H. (2009). The WEKA Data Mining Software: An Update. In SIGKDD Explorations, Volume 11, Issue 1.

John, G. H. and Langley, P. (1995). Estimating Continuous Distributions in Bayesian Classifiers. In Eleventh Conference on Uncertainty in Artificial Intelligence, pages 338-345.

Kuncheva, L. I. (2004). Combining Pattern Classifiers: Methods and Algorithms. John Wiley & Sons, Inc.

Lazarevic, A., Ertoz, L., Kumar, V., Ozgur, A. and Srivastava, J. (2003). A comparative study of anomaly detection schemes in network intrusion detection. In Proceedings of the Third SIAM Conference on Data Mining.

Lee, W., Stolfo, S. J. and Mok, K. W. (1999). A Data Mining Framework for Building Intrusion Detection Models. In IEEE Symposium on Security and Privacy, pages. 120-132.

Lira, M. M. S., de Aquino, R. R. B., Ferreira, A. A., Carvalho, M. A., Neto, O. N. and Santos, G. S. M. (2007). Combining Multiple Artificial Neural Networks Using Random Committee to Decide upon Electrical Disturbance Classification. In International Joint Conference on Neural Networks, pages 2863 - 2868.

Mafra, P. M., Fraga, J. da S., Moll, V., Santin, A. O. (2008). Polvo-IIDS: Um Sistema de Detecção de Intrusão Inteligente Baseado em Anomalias. In VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSEG'08), pages 61-72.

Mukkamala, S., Hung, A.H. and Abraham, A. (2005). Intrusion Detection Using an Ensemble of Intelligent Paradigms. In Journal of Network and Computer Applications, Vol. 28, pages 167-182."

Osareh, A. and Shadgar, B. (2008).Intrusion Detection in Computer Networks based on Machine Learning Algorithms. In International Journal of Computer Science and Network Security, VOL.8 No.11, pages 15-23.

Paxson V. (2007). Considerations and Pitfalls for Conducting Intrusion Detection Research. Keynote, Fourth GI International Conference on Detection of Intrusions & Malware, and Vulnerability Assessment (DIMVA).

Quinlan, R. (1993). C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, San Mateo, CA.

Rish, I. (2001). An empirical study of the naive Bayes classifier. In workshop on Empirical Methods in AI.

Stallings, W. (2005). Cryptography and Network Security Principles and Practices. Prentice Hall, 4th edition.

Tavallaee, M., Bagheri, E., Lu, W. and Ghorbani, A. A. (2009). A Detailed Analysis of the KDD CUP 99 Data Set. In Proceedings of the Second IEEE Symposium on Computational Intelligence in Security and Defense Applications,pages 53-58.

Ting, K. M. and Witten, I. H. (1997). Stacking Bagged and Dagged Models. In Fourteenth international Conference on Machine Learning, pages 367-375.

Webb, G. I. (2000). MultiBoosting: A Technique for Combining Boosting and Wagging. Machine Learning. Vol.40(No.2).

Witten, I. H. and Frank, E. (2005). Data Mining: Pratical Machine Learning Tools and Techniques. Morgan Kaufmann Publishers, 2th Edition.

Zai-an, R., Bin, W., Shi-ming, Z., Zhuang, M. and Rong-ming, S. (2010). A WSRF-enabled Distributed Data Mining Approach to Clustering WEKA4WS-Based. In Proceedings of IEEE Second Symposium on Web Society (SWS), pages 219-226.

Zainal, A., Maarof, M.A., Shamsuddin, S.M. and Abraham, A. (2008). Ensemble of One-class Classifiers for Network Intrusion Detection System. In Proceedings of Fourth International Conference on Information Assurance and Security, pages 180-185.