Classificação de Fragmentos de Arquivos com Técnica de Aprendizagem de Máquina baseada em Árvores de Decisão

Juliano K. M. Oya[1], Bruno W. P. Hoelz[2]

[1] Perito Criminal, Instituto de Criminalística
Polícia Civil do Distrito Federal, Brasília-DF, Brasil

[2] Perito Criminal, Instituto Nacional de Criminalística
Polícia Federal, Brasília-DF, Brasil

juliano.oya@pcdf.df.gov.br, werneck.bwph@pdf.gov.br

Abstract. *The classification of file fragments is an important problem in computer forensics. This paper describes a flexible method for classifying file fragments using machine learning techniques. We used evidence files from real forensic cases to generate the training and testing fragments. From 12,153 evidence files of 21 different types, we generated and selected over a million of fragments with 1, 2 and 4 kilobytes of size. For each fragment, we extracted 45 attributes, which were subjected to machine learning techniques based on decision trees and, as a result, we obtained an average hit percentage of 98.78% for binary classifiers and 86.05% for multinomial classifiers.*

Resumo. *A classificação de fragmentos de arquivos é um importante problema na computação forense. Este artigo descreve um método flexível para classificar fragmentos de arquivos através de técnicas de aprendizagem de máquina. Foram utilizados arquivos de evidências de casos periciais reais para gerar os fragmentos de treinamento e teste. A partir de um total de 12.153 arquivos de evidências, de 21 tipos diferentes, foram gerados e selecionados mais de um milhão fragmentos de tamanhos de 1, 2 e 4 kilobytes. De cada fragmento foram extraídos 45 atributos, os quais foram submetidos a técnicas de aprendizagem de máquina baseadas em árvores de decisão e, como resultado, obteve-se um percentual de acerto médio de 98,78% para classificadores binários e de 86,05% para classificadores multinomiais.*

1. Introdução

A classificação de fragmentos é uma importante atividade na decodificação de fragmentos de memória, análise de fragmentos de arquivos em tráfegos de rede, detecção de *software* maliciosos, entre outras [Roussev e Quates 2013]. No caso específico da computação forense, a classificação de fragmentos é uma atividade essencial e indispensável para o propósito de recuperação de arquivos armazenados em mídias sem informações sobre o seu sistema de arquivos [Fitzgerald et al. 2012].

O problema da classificação de fragmentos consiste em determinar o tipo do arquivo ao qual pertence o fragmento. Em situações em que o fragmento é parte do cabeçalho ou do final do arquivo, a classificação é uma tarefa mais simples, visto que a maioria dos arquivos armazenam assinaturas ou estruturas de dados que indicam o seu tipo nas áreas iniciais ou finais do arquivo. Entretanto, em situações em que os

fragmentos são partes do meio do arquivo, onde há pouca ou nenhuma informação que identifiquem o seu tipo, a classificação torna-se uma tarefa desafiadora.

A técnica mais utilizada para a classificação e recuperação de arquivos – inclusive fragmentos de arquivos – é por meio da busca por assinaturas conhecidas, as quais ficam localizadas no início e no final de um arquivo. Entretanto, essa técnica tem como pressupostos que o arquivo possui o cabeçalho e o final do arquivo e, ainda, que o arquivo foi alocado de forma contínua na mídia de armazenamento. Como mostrado por Cohen (2007), nem sempre esses pressupostos estão presentes nas mídias de armazenamento.

A fim de verificar a aplicabilidade da técnica de recuperação de arquivos baseadas em assinaturas, foi realizado um teste sobre um conjunto de fragmentos de arquivos sem as informações de cabeçalho e de final do arquivo (o mesmo conjunto de fragmentos, como será apresentado posteriormente, foi utilizado nos experimentos deste trabalho). Para tanto, foi utilizado o arquivo de perfis da ferramenta *Foremost* (2016) – com 80 assinaturas de 40 tipos de arquivos – para classificar os fragmentos de acordo com as assinaturas encontradas dentro dos fragmentos. O resultado, apresentado na Tabela 1, são percentuais de acertos de 0,54%, 0,92% e 1,25% para a classificação dos fragmentos, o que mostra que a técnica não pode ser aplicada na recuperação de fragmentos sem as informações de cabeçalho e de final do arquivo.

Tabela 1. Resultado da classificação baseada em assinatura dos fragmentos.

	FRAGMENTOS DE 1 KBYTES	FRAGMENTOS DE 2 KBYTES	FRAGMENTOS DE 4 KBYTES
Número de fragmentos	1.215.000	1.215.000	1.210.160
Número de acertos	6.661	11.284	15.245
Número de erros	1.208.339	1.203.716	1.194.915
Não classificados	816.267	696.807	530.402
Percentual de acertos	0,54%	0,92%	1,25%

Alternativamente ao uso de assinaturas na classificação de fragmentos, vários trabalhos foram realizados com a aplicação de técnicas de aprendizagem de máquina para resolver o problema de classificação de fragmentos [Veenman 2007, Calhoun e Coles 2008, Axelsson 2010, Conti et al. 2010, Li et al. 2010, Fitzgerald et al. 2012].

A aprendizagem de máquina, aplicada na classificação de fragmentos de arquivos, consiste em construir um modelo computacional a partir de dados de atributos de fragmentos cujos tipos são conhecidos e, posteriormente, utilizar o modelo construído para predizer ou classificar novos fragmentos cujos tipos são desconhecidos.

Nos trabalhos citados, de modo geral, a aplicação de técnicas de aprendizagem de máquina segue os seguintes passos: coleta de arquivos, fragmentação dos arquivos, extração dos atributos dos fragmentos, consolidação dos atributos em dados de treinamento e validação, treinamento do modelo de classificação e, por fim, a avaliação do modelo.

A proposta apresentada neste trabalho inova ao considerar os atributos utilizados por outros pesquisadores e, ainda, novos atributos extraídos a partir dos fragmentos; ao utilizar uma base de dados de 2.830.472 fragmentos extraídos de arquivos de 21 tipos diferentes; ao utilizar a aprendizagem de máquina baseada em árvores de decisão, cuja precisão média na classificação dos fragmentos se mostrou promissora e superior aos outros trabalhos.

O restante do trabalho está organizado em 5 seções. Na seção 2 são analisados os principais pontos dos trabalhos realizados na mesma área de pesquisa. Na seção 3 é descrita a classificação de fragmentos baseada em árvores de decisão. Na seção 4 é descrita a metodologia de trabalho para a realização do experimento. Na seção 5 é descrito o experimento realizado, o qual é dividido em subseções onde são relatadas a seleção dos arquivos, a extração dos fragmentos, a extração dos atributos, a classificação dos fragmentos e o resultado obtido. Na seção 6, por fim, são apresentadas a conclusão e a indicação de trabalhos futuros.

2. Trabalhos relacionados

Os trabalhos de Veenman (2007), Calhoun e Coles (2008), Axelsson (2010), Conti et al. (2010), Li et al. (2010) e Fitzgerald et al. (2012) também exploraram a aplicação de técnicas de aprendizagem de máquina com o objetivo de classificar fragmentos de arquivos. Em geral, os experimentos realizados nesses trabalhos variam em número e tamanho dos fragmentos utilizados como base de treinamento e validação, número de tipos de arquivos utilizados na classificação, números de atributos extraídos dos fragmentos e, por fim, a técnica de aprendizagem de máquina utilizada.

No experimento realizado por Veenman (2007), foram utilizados um conjunto de 3.000 a 20.000 fragmentos por tipo de arquivos, dentre 11 tipos distintos. Foram utilizados fragmentos de 4 *Kbytes* de tamanho, dos quais foram extraídos os atributos de histograma de 1-grama, entropia de *Shannon* de 1-grama e a complexidade de *Kolmogorov*. O autor utiliza um discriminante linear para classificar os fragmentos e obtêm uma precisão média do classificador de 45%.

Já Calhoun e Coles (2008) alcançaram uma precisão média de 88%, que fora obtida limitando-se a 4 tipos de arquivos (`JPG`, `BMP`, `GIF` e `PDF`), os quais são analisados aos pares, através da aplicação de um classificador de regressão linear. O autor extraiu, dos fragmentos, atributos estatísticos (média, desvio padrão e correlação de valores), frequências de caracteres *ASCII*, entropia de *Shannon* de 1-grama e, ainda, combinou os valores desses atributos.

O trabalho realizado por Axelsson (2010) utilizou o maior número de tipos de arquivos (28 tipos distintos), entretanto, obteve a menor precisão média com o classificador *k-nearest-neighbors* (34%). O autor utilizou 31.541 fragmentos de 512 *bytes*, dos quais extraiu a medida NDC (*Normalised Compression Distance*), que é baseada no cálculo de complexidade de *Kolmogorov*.

O trabalho realizado por Conti et al. (2010) utiliza o classificador *k-nearest-neighbors* aplicado sobre os atributos de medidas estatísticas, entropia de *Shannon* de 2-gramas, peso de *Hamming*, Qui-quadrado e média aritmética, os quais foram extraídos de 14.000 fragmentos de 1 *Kbyte*. Os autores obtiveram uma precisão média de 96%, entretanto, em vez de utilizar tipos específicos de arquivos, os autores utilizaram grupos bastante abrangentes no classificador: randômico/compactado/criptografado, codificado em *Base64*, codificado em *Uuencoding* (*Unix-to-Unix encoding*), código de máquina, texto e *Bitmap*.

Através da aplicação de SVM (*Support Vector Machine*), Li et al. (2010) obtiveram uma precisão média de 81,5%. Para seu experimento, os autores extraíram o histograma de 1-grama de fragmentos de 4 *Kbytes* de 3.600 arquivos. Utilizaram, no entanto, apenas 5 tipos de arquivos (`JPEG`, `DLL`, `EXE`, `MP3` e `PDF`).

Dentre os trabalhos apresentados, os autores Fitzgerald et al. (2012) realizaram o experimento com o maior número de atributos extraídos dos fragmentos de 24 tipos de arquivos. Para cada fragmento foram extraídos: histograma 1-grama, histograma 2-gramas, medidas estatísticas, entropia de *Shannon* 1-grama, entropia de *Shannon* 2-gramas, peso de *Hamming*, média aritmética, taxa de compressão do arquivo, média da distância entre *bytes* consecutivos e maior sequência contígua de *bytes* repetidos. Apesar da grande quantidade de atributos considerados, a precisão média de 47,5% pode ser considerada baixa.

3. Classificação de fragmentos baseada em árvores de decisão

Os algoritmos baseados em árvores de decisão são destinados tanto para a regressão quanto para a classificação. Tais algoritmos envolvem a segmentação recursiva dos dados em subconjuntos, um para cada valor do atributo. Para cada árvore de decisão é definida um conjunto de regras, que definem um caminho a ser percorrido na árvore até a classificação do fragmento.

Os dados para a aprendizagem dos algoritmos podem ser ajustados, de modo a produzir dois modelos de classificadores:

1. *Classificador binomial ou binário*: o classificador resultante será capaz de distinguir dois tipos. Esse modelo foi utilizado por Calhoun e Coles (2008), que realizaram a classificação de pares de tipos na forma `TYPE` vs. `NOT-TYPE`.
2. *Classificador multinomial ou multiclasse*: o classificador resultante será capaz de distinguir múltiplos, ou seja, um conjunto de dois ou mais tipos. Esse modelo foi utilizado por Fitzgerald et al. (2012).

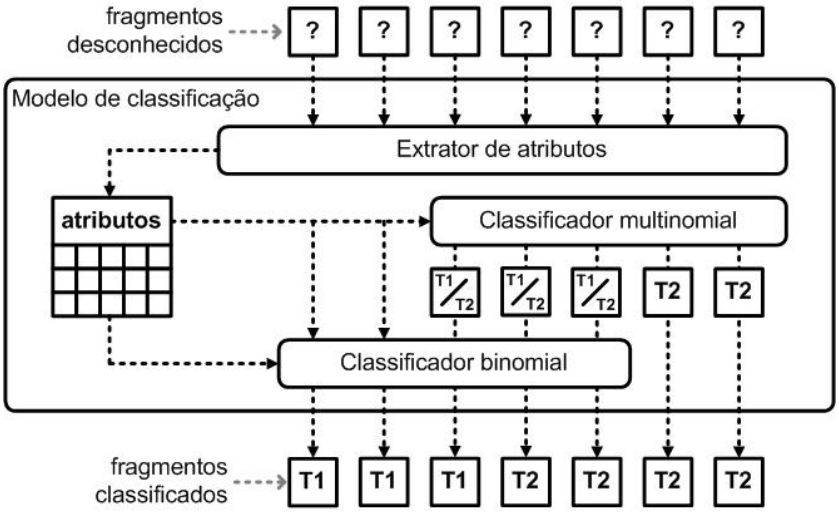

Figura 1. Aplicação dos modelos de classificadores binomiais e multinomiais.

Comumente os classificadores binomiais e multinomiais são utilizados de forma isolada. Um cenário possível é a utilização combinada dos dois modelos de classificadores, como ilustrado na Figura 1, na qual um classificador mais especializado é aplicado sempre que se há dúvidas a respeito do tipo de um fragmento previamente processado por um classificado mais genérico. A seção seguinte descreve o método de trabalho para se obter os classificadores binomiais e multinomiais.

4. Método de trabalho

O método de trabalho consiste em 4 fases principais, quais sejam: (1) seleção dos arquivos, (2) extração dos fragmentos dos arquivos, (3) extração dos atributos dos fragmentos e (4) treinamento e validação do classificador.

No fluxograma apresentado na Figura 2 são representadas as fases (1) e (2). Nela pode-se observar como o conjunto inicial de arquivos é filtrado, quando são retirados os arquivos idênticos, menores que 10 *Kbytes* e maiores que 100 *Mbytes*.

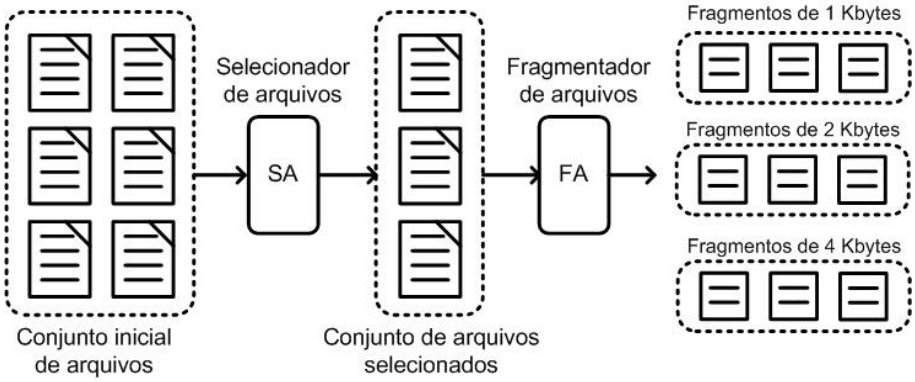

Figura 2. Visão geral das fases seleção e fragmentação de arquivos.

Já no fluxograma apresentado na Figura 3 é representada a fase (3), quando são extraídos um conjunto de atributos dos fragmentos dos arquivos. Os símbolos `SA`, `FA`, `EA` (`EA1`, `EA2`, ..., `EAn`) e `CD` são módulos do protótipo desenvolvido para automatizar as atividades das fases (1), (2), (3) e (4).

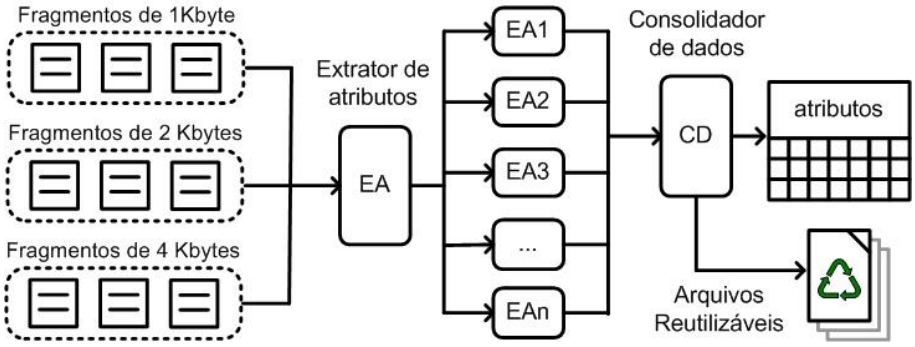

Figura 3. Visão geral da fase de extração dos atributos dos fragmentos.

Ainda na Figura 3, pode-se observar que foram implementados módulos específicos para extrair cada atributo (são os subitens denominados `EA1`, `EA2`, ..., `EAn`). Ao final dessa fase, os dados são consolidados e salvos em um conjunto de arquivos.

A fase (4) do método de trabalho consiste em transformar o conjunto de dados dos atributos em conjuntos binomiais e multinomiais, ambos para o aprendizado supervisionado do algoritmo de classificação baseado em árvores de decisão. Por fim, o resultado é um conjunto reutilizável de modelos de classificadores binomiais e multinomiais. Na Figura 4 é ilustrada essa fase.

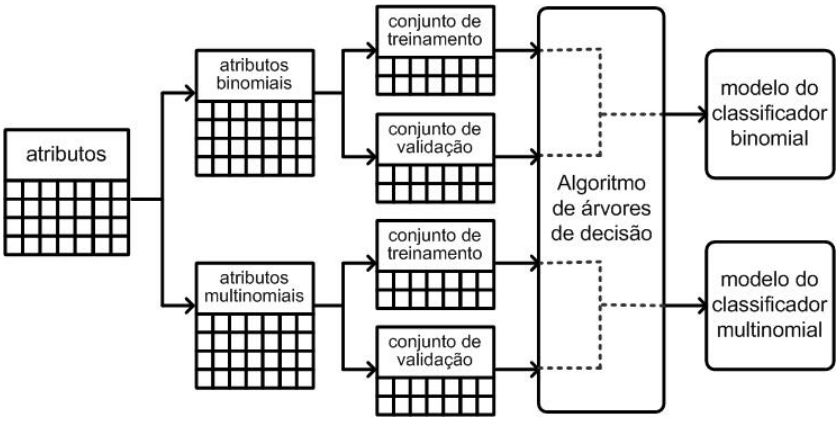

Figura 4. Visão geral da fase de treinamento e validação dos classificadores.

5. Experimentos

Utilizando-se do método de trabalho apresentado anteriormente, foram realizados vários experimentos com dados de fragmentações de 1, 2 e 4 *Kbytes*. Todo o processo, que envolve a seleção de arquivos, a extração de fragmentos e a extração de atributos do fragmento, foi automatizado através de um protótipo desenvolvido em linguagem *Java*. Os dados coletados foram exportados para a ferramenta *Weka* [Waikato 2016a], na qual foi aplicado o algoritmo de classificação baseado em árvores de decisão. Nas seções seguintes é descrita cada fase do processo.

5.1. Desenvolvimento do protótipo

Um protótipo em linguagem *Java* foi desenvolvido como prova de conceito. A implementação automatiza todas as fases do método de trabalho, ou seja, realiza a seleção dos arquivos, a extração dos fragmentos, a extração dos atributos e a consolidação final dos dados. A automatização das atividades se mostrou essencial para a realização do experimento, dada a grande quantidade de informações e de passos para processar e analisar os dados.

Para cada atributo foi desenvolvida uma classe específica que o extrai (ressalte-se que o protótipo pode ser facilmente adaptado para extrair novos atributos). Dessa forma, foram implementados algoritmos específicos para extrair os atributos descritos na Tabela 4 da seção 5.4. Ainda, a fim de criar a compatibilidade com o projeto *Weka*, o protótipo realiza a conversão automática dos dados dos atributos extraídos para o formato `ARFF` [Waikato 2016b].

5.2. Seleção do algoritmo de classificação

Foi realizado um experimento com um conjunto reduzido de 21.000 registros de dados de fragmentos de 1, 2 e 4 *Kbytes* (ou seja, cerca de 2% da base total utilizada no experimento principal). Sobre esses registros, foram executados os algoritmos de classificação baseados em árvores de decisão *Decision Stump*, *J48*, *REPTree*, *Random Forest* e *Random Tree* [Waikato 2016c], a fim de avaliar o percentual de acertos, o tempo de execução e o tamanho físico do modelo do classificador. O resultado é apresentado na Figura 5, onde a média dos valores de fragmentos de 1, 2 e 4 *Kbytes* dos parâmetros avaliados são postos em uma escala de 0 a 1.

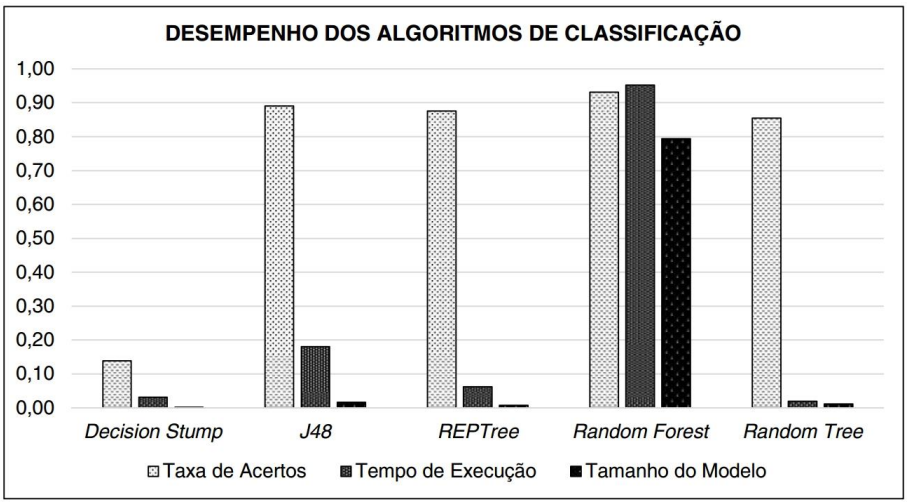

Figura 5. Avaliação de desempenho dos algoritmos de classificação.

Dessa forma, o algoritmo *J48* foi selecionado para a realização do experimento principal, por apresentar taxa de acertos acima da média e bons tempo de execução e tamanho físico do modelo de classificação – em detrimento, por exemplo, do algoritmo *Random Forest*, que apesar de apresentar a melhor taxa de acertos, teve o pior tempo de processamento e originou um modelo com tamanhos físicos extremamente grandes.

5.2. Seleção dos arquivos

Foram selecionados cerca de 500 relatórios de evidências periciais de casos reais, cujas investigações ocorreram nos anos de 2014, 2015 e 2016. Dos relatórios, foram extraídos os arquivos de evidências vinculados a eventos de natureza criminal e, dentre esses arquivos, foram selecionados 12.153 arquivos de 21 tipos diferentes. Na Tabela 2 são descritos os tipos selecionados para o experimento.

Tabela 2. Descrição dos tipos de arquivos selecionados para o experimento.

TIPO	DESCRIÇÃO DO TIPO
AVI	Arquivos de vídeos AVI (*Audio Video Interleave*).
BMP	Arquivos de imagens *Bitmap*.
DLL	Arquivo de bibliotecas dinâmicas DLL (*Dynamic-Link Library*).
DOC	Arquivos de documentos do programa *Microsoft Word*.
DOCX	Arquivos de documentos do programa *Microsoft Word* com formato XML.
DWG	Arquivos do programa *Autocad*.
EML	Arquivos de mensagens de *e-mails* com codificação *Base64*.
EXE	Arquivos de programas executáveis.
HTML	Páginas *web* no formato HTML (*Hiper Text Markup Language*).
JAR	Arquivos de programas ou bibliotecas em *Java*.
JPG	Arquivos de imagem JPEG (*Joint Photographic Experts Group*).
MP3	Arquivos de áudio MP3 (*MPEG-1/2 Audio Layer 3*).
MP4	Arquivos de vídeos MP4 (*MPEG-4 Part 14*).
PDF	Arquivos de documentos do programa *Adobe Acrobat Reader*.
RTF	Arquivos de documentos RTF (*Rich Text Format*).
SQLL	Arquivos de banco de dados *SQLite*.
TIF	Arquivos de imagens TIFF (*Tagged Image File Format*).
TXT	Arquivos de texto simples.
WMV	Arquivos de vídeos WMV (*Windows Media Player*).
XML	Arquivos de documentos XML (*Extensible Markup Language*).
ZIP	Arquivos de documentos compactado no formato *Winzip*.

Os tipos selecionados foram aqueles mais bem representados dentro do conjunto total dos arquivos. Dessa forma, foram selecionados os tipos que possuíam a maior quantidade de arquivos e a maior quantidade de espaço ocupado em disco. O primeiro critério visa obter a maior variedade possível de arquivos para o experimento; já o segundo visa obter uma quantidade expressiva de fragmentos.

A medida que os arquivos eram selecionados, também eram calculados os valores do *hash* MD5 dos arquivos, a fim de evitar que a amostra possuísse arquivos idênticos. Portanto, os arquivos que possuíam o mesmo valor de *hash* de um arquivo já selecionado eram excluídos.

Foram excluídos, ainda, os arquivos menores que 10 *Kbytes*, os quais não gerariam fragmentos de 4 *Kbytes* úteis para o experimento, uma vez que os fragmentos inicial e final dos arquivos (que podem conter dados de assinaturas do tipo de arquivo) são descartados; e também foram excluídos os arquivos maiores que 100 *Mbytes*, os quais produziriam uma grande quantidade não desejada de fragmentos do mesmo arquivo. O resultado final do processo de seleção de arquivos é apresentado na Tabela 3.

Tabela 3. Quantitativos dos arquivos selecionados para o experimento.

TIPO	TAMANHO (KB)	% DO TOTAL	QUANTIDADE	% DE ARQUIVOS
AVI	195.233	4,76%	12	0,10%
BMP	195.311	4,76%	199	1,64%
DLL	195.313	4,76%	300	2,47%
DOC	195.312	4,76%	901	7,41%
DOCX	195.307	4,76%	716	5,89%
DWG	195.309	4,76%	269	2,21%
EML	195.310	4,76%	198	1,63%
EXE	195.306	4,76%	72	0,59%
HTML	195.312	4,76%	2.424	19,95%
JAR	195.312	4,76%	275	2,26%
JPG	195.193	4,76%	265	2,18%
MP3	195.306	4,76%	136	1,12%
MP4	195.211	4,76%	88	0,72%
PDF	195.214	4,76%	358	2,95%
RTF	195.298	4,76%	2.020	16,62%
SQLL	195.312	4,76%	401	3,30%
TIF	194.737	4,75%	33	0,27%
TXT	195.310	4,76%	1.320	10,86%
WMV	195.308	4,76%	29	0,24%
XML	195.312	4,76%	1.886	15,52%
ZIP	195.308	4,76%	251	2,07%

5.3. Extração dos fragmentos dos arquivos

A partir dos 12.153 arquivos selecionados foram extraídos os fragmentos de 3 tamanhos distintos: 1, 2 e 4 *Kbytes*. Os fragmentos que faziam parte do cabeçalho ou do final dos arquivos foram excluídos, a fim de eliminar qualquer vestígio da assinatura dos arquivos. O resultado final foi a seleção de 2.830.472 fragmentos, divididos da seguinte forma: 945.000 fragmentos de 1 *Kbytes*, 945.000 fragmentos de 2 *Kbytes* e 940.472 fragmentos de 4 *Kbytes*.

Os fragmentos foram selecionados de forma homogênea entre os tipos de arquivos. Para a maioria dos tipos, foram selecionados 45.000 fragmentos por tipo. A exceção ocorreu para os tipos HTML, RTF e XML, para os quais foram selecionados 42.578, 43.783 e 44.111 fragmentos de 4 *Kbytes*, respectivamente.

5.4. Extração dos atributos dos fragmentos

A partir dos fragmentos selecionados, foram extraídos os principais atributos utilizados nos experimentos realizados por Veenman (2007), Calhoun e Coles (2008), Axelsson (2010), Conti et al. (2010), Li et al. (2010), Fitzgerald et al. (2012). Foram extraídos, ainda, atributos de frequência de caracteres *Base85*, *Monte Carlo Pi*, correlação serial, frequência de *bytes* `x00`, `xFF` e `xF8`, entropia de *Shannon* de 3-gramas, histograma de 3-gramas, identificador de língua, identificador de linguagem de programação e identificador de delimitadores de *tags*. Na Tabela 4 são descritos, brevemente, os principais atributos extraídos de cada fragmento do experimento.

Tabela 4. Descrição dos atributos extraídos de cada fragmento.

NOME DADO AO ATRIBUTO	DESCRIÇÃO DO ATRIBUTO
`ascii-percentual`	Percentual de caracteres `ASCII` imprimíveis.
`ascii-freq`	Frequência de *bytes* no fragmento com o valor *v*, onde `x00` ≤ *v* < `x20`.
`ascii-low-freq`	Frequência de *bytes* no fragmento com o valor *v*, onde `x20` ≤ *v* ≤ `x7E`.
`ascii-high-freq`	Frequência de *bytes* no fragmento com o valor *v*, onde `x80` ≤ *v*.
`base64-percentual`	Percentual de caracteres com codificação *Base64*.
`base85-percentual`	Percentual de caracteres com codificação *Base85*.
`byte-mean`	Média dos valores dos *bytes* do fragmento.
`byte-sd`	Desvio padrão dos valores dos *bytes* do fragmento.
`byte-cornext`	Correlação dos valores dos *bytes* nas posições *n* e *n+1*.
`chi-square`	Calcula o grau de aleatoriedade dos *bytes* do fragmento.
`npl-dist-mean`	Calculo média da distância entre os valores de dois *bytes* consecutivos.
`npl-long-seq`	Cálculo da sequência mais longa de *bytes* repetidos.
`monte-carlo-pi`	Cálculo do valor *Monte Carlo para Pi* utilizando-se cada sequência sucessiva de 6 *bytes* como coordenadas *X* e *Y*.
`serial-correlation`	Calcula o grau de dependência de cada *byte* com o *byte* anterior.
`pattern-[FF,F8] -[1,2,3]gram`	Contagem do número de *bytes* `xFF` e `xF8`, inclusive sequencias de 1, 2 e 3 gramas (ou seja, sequências de 1, 2 e 3 *bytes* iguais a `xFF` e `xF8`).
`hamming-weight`	Calcula o total de *bits* 1dividido pelo total de *bits* do fragmento.
`histo-[1,2,3]gram-modes`	Soma dos 4 maiores valores do histograma de combinações de 1, 2 e 3 *bytes*.
`histo-[1,2,3]gram-sd`	Desvio padrão dos valores do histograma de combinações de 1, 2 e 3 *bytes*.
`histo-[1,2,3]gram-cornext`	Correlação da frequência dos valore *m* e *m+1* do histograma de combinações de 1, 2 e 3 *bytes*.
`lang-detect`	Detecção da língua utilizada no texto (ex: português, inglês, espanhol, japonês, etc.)
`program-lang`	Detecção da linguagem de programação ou de marcação utilizada. (ex: Java, C, Python, HTML, etc).
`entropy-[1,2,3]gram`	Cálculo do grau de entropia de 1, 2 e 3 gramas (ou seja, combinações de 1, 2 e 3 *bytes*).
`tag-score`	Contagem do número de *bytes* com valores `x28`, `x29`, `x3C`, `x3E`, `x5B`, `x5D`, `x7B`, `x7D`.
`zero-[1,2]gram-percentual`	Percentual de *bytes* `x00` no fragmento, inclusive sequencias de 1 e 2 gramas.
`zip-compression-score`	Taxa de compressão do fragmento.

A partir dessa fase do experimento, cada fragmento passa a ser representado pelo seu conjunto de atributos.

5.5. Treinamento e validação do classificador

O conjunto de atributos foram consolidados e, em seguida, convertidos para o formato `ARFF`, o qual é o formato de arquivo utilizado pelo projeto *Weka*. Por meio da ferramenta *Weka* foi aplicado o algoritmo de árvores de decisão *J48*.

O algoritmo *J48*, cuja implementação é baseada no algoritmo *C4.5* [Quinlan 1993], percorre todos os atributos de modo a identificar aquele apresenta o maior ganho de informação (ou seja, maior contribuição para o resultado na árvore). Após identificar esse atributo, ele é definido como um nó na árvore (ou a raiz, caso seja a primeira iteração) [Bell 2014].

Neste trabalho, foi realizada a classificação binomial na forma `TYPE` vs. `NOT-TYPE` e, dessa forma, o algoritmo de classificação foi executado 21 vezes, uma vez para cada tipo, na qual a cada execução o fragmento é classificado como de um tipo específico ou como diferente do tipo específico. Também foi realizada a classificação multinomial, na qual o classificador resultante será capaz de distinguir qualquer um dos 21 tipos.

Na Figura 6 é apresentada, como exemplo, a árvore de decisão extraída após o processamento do algoritmo *J48* para a classificação binomial dos fragmentos de 1 *Kbyte* do tipo `EML`. Cada linha segue o formato `[regra-de-decisão]: [classificação]([total-de-instâncias]/[erro-de-classificação])`. No final da figura pode-se observar o percentual de classificações corretas e incorretas e a matriz de confusão do classificador.

```
base64-percentual <= 0.998047
|   ascii-percentual <= 0.999023: NOT-EML (774069.0)
|   ascii-percentual > 0.999023
|   |   histo-2gram-sd <= 0.071029
|   |   |   ascii-freq <= 0.987305
|   |   |   |   histo-2gram-modes <= 0.042198
|   |   |   |   |   ascii-freq <= 0.979492
|   |   |   |   |   |   base64-percentual <= 0.843137
|   |   |   |   |   |   |   tag-score-3C <= 0.055172: EML (22.0/2.0)
|   |   |   |   |   |   |   tag-score-3C > 0.055172:  NOT-EML  (2.0)
|   |   |   |   |   |   base64-percentual > 0.843137: NOT-EML (23.0)
|   |   |   |   |   ascii-freq > 0.979492: EML (35.0)
|   |   |   |   histo-2gram-modes > 0.042198
|   |   |   |   |   tag-score-3E <= 0.042471
|   |   |   |   |   |   histo-3gram-modes <= 0.038348
|   |   |   |   |   |   |   entropy-2gram <= 0.824984
|   |   |   |   |   |   |   |   lang-detect = ??: NOT-EML (0.0)

... linhas removidas ...

Correctly Classified Instances      944817      99.9806 %
Incorrectly Classified Instances       183      0.0194 %

... linhas removidas ...

Confusion Matrix:
      a      b   <-- classified as
  44880    120 |     a = EML
     63 899937 |     b = NOT-EML
```

Figura 6. Árvore de decisão produzida pelo algoritmo *J48* sobre fragmentos do tipo `EML`.

5.6. Resultados

Ao final do processamento do algoritmo *J48*, foram obtidos 63 classificadores binomiais (específicos para cada um dos 21 tipos e para cada tamanho de fragmento de 1, 2 e 4 *Kbytes*) e 3 classificadores multinomiais (específicos para cada um dos tamanhos de 1, 2 e 4 *Kbytes*). As taxas de precisão foram consolidadas e são apresentadas nas Tabelas 5 e 6.

Tabela 5. Resultado da classificação binomial do algoritmo *J48*.

TAXA DE CLASSIFICAÇÃO CORRETA CLASSIFICAÇÃO BINOMIAL			
TIPO	**FRAG. DE 1 KBYTES**	**FRAG. DE 2 KBYTES**	**FRAG. DE 4 KBYTES**
AVI	98,73%	98,88%	99,12%
BMP	99,72%	99,83%	99,88%
DLL	99,22%	99,20%	98,87%
DOC	97,58%	97,31%	97,19%
DOCX	99,70%	99,60%	99,44%
DWG	99,79%	99,56%	99,60%
EML	99,98%	99,99%	99,93%
EXE	96,14%	97,46%	97,93%
HTM	99,90%	99,94%	99,95%
JAR	97,69%	98,39%	98,67%
JPG	96,68%	96,84%	97,40%
MP3	99,14%	99,59%	99,76%
MP4	96,64%	97,75%	98,11%
PDF	96,33%	97,63%	98,40%
RTF	99,79%	99,85%	99,92%
SQLITE	99,67%	99,69%	99,68%
TIF	99,41%	99,71%	99,56%
TXT	99,56%	99,69%	99,84%
WMV	98,54%	98,89%	98,15%
XML	99,89%	99,94%	99,86%
ZIP	95,88%	95,70%	96,26%
média	98,57%	98,83%	98,93%

Tabela 6. Resultado da classificação multinomial do algoritmo *J48*.

TAXA DE CLASSIFICAÇÃO CORRETA CLASSIFICAÇÃO MULTINOMIAL			
TIPO	**FRAG. DE 1 KBYTES**	**FRAG. DE 2 KBYTES**	**FRAG. DE 4 KBYTES**
AVI	87,10%	88,60%	91,40%
BMP	97,60%	98,20%	98,70%
DLL	90,60%	89,70%	86,10%
DOC	97,00%	96,00%	93,80%
DOCX	69,10%	68,70%	69,40%
DWG	98,10%	95,20%	95,60%
EML	99,80%	99,90%	99,20%
EXE	58,10%	78,00%	79,90%
HTM	98,90%	99,40%	99,60%
JAR	69,20%	79,30%	83,20%
JPG	64,70%	68,00%	73,50%
MP3	90,50%	95,00%	97,10%
MP4	58,10%	72,60%	79,30%
PDF	56,40%	70,40%	79,80%
RTF	97,10%	97,80%	98,80%
SQLITE	96,10%	96,40%	96,60%
TIF	92,30%	96,50%	95,30%
TXT	96,60%	98,10%	98,80%
WMV	81,80%	85,90%	79,60%
XML	98,90%	99,30%	98,80%
ZIP	50,30%	49,00%	56,70%
média	83,25%	86,76%	88,15%

Os classificadores binomiais apresentaram melhores percentuais de acertos (uma média de 98,78% de classificações corretas). Foi observado, ainda, que a medida que os fragmentos aumentam de tamanho, há um aumento na taxa de precisão.

Já os classificadores multinomiais, apresentaram uma queda em seus percentuais de classificação (uma média de 86,05% de classificações corretas). No caso específico do classificador multinomial, como era esperado, há uma redução no percentual de acertos do modelo, em troca da flexibilidade de se ter um único modelo capaz de classificar todos os tipos.

A seguir, na Tabela 7, é apresentada a matriz de confusão do classificador multinomial para fragmentos de 4 *Kbytes*. A diagonal da tabela indica os índices de classificação positiva, ou seja, um fragmento do tipo sendo classificado corretamente.

Dessa forma, os dados com a taxa de classificação correta para os fragmentos de 4 *Kbytes*, os quais foram inicialmente apresentados na Tabela 6, são posicionados na diagonal da matriz da Tabela 7. Ademais, para cada item da diagonal, na mesma linha ou coluna, são apresentados os percentuais de classificações incorretas de cada tipo.

Tabela 7. Matriz de confusão da classificação multinomial do algoritmo *J48* (4 *Kbytes*).

	classificado como																				
	AVI	BMP	DLL	DOCX	DOC	DWG	EML	EXE	HTML	JAR	JPG	MP3	MP4	PDF	RTF	SQLI	TIF	TXT	WMV	XML	ZIP
AVI	91,4	0,0	0,2	0,1	0,3	0,2	0,0	0,2	0,0	0,1	0,1	0,7	2,6	0,2	0,0	0,0	0,3	0,1	2,8	0,0	0,6
BMP	0,0	98,7	0,2	0,1	0,1	0,1	0,0	0,0	0,0	0,1	0,0	0,0	0,1	0,1	0,0	0,1	0,2	0,1	0,0	0,0	0,0
DLL	0,2	0,3	86,1	0,3	1,6	1,2	0,0	3,5	0,0	0,8	0,3	0,1	0,7	0,7	0,1	1,3	0,1	0,8	0,7	0,1	1,1
DOCX	0,1	0,1	0,3	93,8	1,9	0,1	0,0	0,1	0,0	0,2	0,7	0,1	0,2	0,9	0,0	0,4	0,0	0,0	0,2	0,0	1,0
DOC	0,2	0,1	1,5	1,6	69,4	0,3	0,0	0,7	0,0	0,9	14,9	0,1	0,7	2,3	0,1	0,6	0,3	0,1	1,1	0,0	4,9
DWG	0,2	0,1	0,9	0,2	0,3	95,6	0,0	0,2	0,0	0,1	0,1	0,0	0,2	0,3	0,0	0,4	0,1	0,2	0,7	0,0	0,2
EML	0,0	0,0	0,0	0,0	0,0	0,0	99,2	0,0	0,1	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,7	0,0
EXE	0,2	0,1	4,0	0,1	0,8	0,3	0,0	79,9	0,0	1,7	0,0	0,1	2,7	0,5	0,0	0,3	0,1	0,4	0,3	0,0	8,5
HTML	0,0	0,0	0,0	0,0	0,0	0,0	0,1	0,0	99,6	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,2	0,0
JAR	0,2	0,1	0,8	0,3	0,9	0,2	0,0	2,5	0,0	83,2	0,2	0,0	0,8	1,8	0,0	0,2	0,6	0,0	1,1	0,1	7,0
JPG	0,1	0,0	0,3	0,7	14,7	0,2	0,0	0,0	0,0	0,2	73,5	0,0	0,1	5,2	0,0	0,0	0,3	0,1	0,4	0,0	4,1
MP3	0,7	0,0	0,0	0,0	0,1	0,1	0,0	0,0	0,0	0,1	0,0	97,1	0,9	0,1	0,0	0,0	0,0	0,1	0,6	0,0	0,1
MP4	2,8	0,0	0,8	0,2	0,8	0,3	0,0	3,8	0,0	0,7	0,1	1,0	79,3	0,7	0,0	0,0	0,1	0,1	6,1	0,0	3,0
PDF	0,2	0,1	0,7	1,1	2,8	0,4	0,0	1,1	0,0	2,1	5,9	0,1	0,7	79,8	0,1	0,1	0,4	0,0	0,8	0,0	3,4
RTF	0,0	0,1	0,1	0,0	0,2	0,1	0,0	0,1	0,0	0,1	0,0	0,0	0,1	0,1	98,8	0,0	0,0	0,0	0,0	0,0	0,2
SQLI	0,1	0,1	1,0	0,5	0,5	0,4	0,0	0,2	0,0	0,1	0,1	0,0	0,1	0,1	0,0	96,6	0,0	0,0	0,2	0,0	0,1
TIF	0,4	0,1	0,2	0,0	0,4	0,1	0,0	0,1	0,0	0,6	0,2	0,0	0,2	0,3	0,0	0,0	95,3	0,0	1,2	0,0	0,9
TXT	0,0	0,0	0,2	0,0	0,1	0,1	0,0	0,0	0,0	0,0	0,0	0,0	0,2	0,0	0,0	0,1	0,0	98,8	0,1	0,1	0,0
WMV	3,2	0,0	0,7	0,2	1,3	0,7	0,0	0,4	0,0	1,1	0,4	0,6	6,5	0,7	0,1	0,2	1,4	0,1	79,6	0,0	3,0
XML	0,0	0,0	0,1	0,0	0,0	0,0	0,7	0,0	0,2	0,1	0,0	0,0	0,0	0,0	0,0	0,0	0,0	0,1	0,0	98,8	0,0
ZIP	0,9	0,1	1,3	1,2	5,6	0,3	0,0	10,9	0,0	6,9	4,9	0,1	3,4	3,5	0,1	0,1	1,0	0,0	2,9	0,0	56,7

Ainda na Tabela 7, foram destacados os maiores percentuais de erros de classificação dos fragmentos. Esses são casos que requerem maiores estudos para introdução de novos atributos que auxiliem nessa classificação. Por exemplo, no caso específico da classificação dos fragmentos dos tipos `JPG`, 14,7% dos fragmentos foram classificados como fragmentos do tipo `DOC` e 14,9% de fragmentos do tipo `DOC` foram considerados como fragmentos `JPG`. Esse erro de classificação pode ser atribuído a presença de imagens `JPG` nos fragmentos do tipo `DOC`.

Da mesma forma, observa-se que o tipo `ZIP` tem a pior taxa de acerto (56,7%). A matriz mostra que 10,9% dos fragmentos `ZIP` foram classificados como fragmentos do tipo `EXE`. No sentido oposto, 8,5% dos fragmentos `EXE` foram classificados como fragmentos do tipo `ZIP`. Isso decorre do fato de que vários outros tipos de arquivos utilizam compressão semelhante àquelas aplicadas nos arquivos `ZIP`, tais como `JAR`, `DOCX` e executáveis comprimidos com *packers* (`EXE`).

Os casos de falso-positivos e falso-negativos têm impactos diretos sobre o processo posterior de recuperação do arquivo. No primeiro, o componente de recuperação tentaria recuperar um fragmento inválido; já no segundo, o componente de recuperação poderia ignorar incorretamente o fragmento válido. Esse cenário indica a necessidade de classificadores específicos como os classificadores binários apresentados anteriormente. Note-se que vários classificadores binários podem ser aplicados a um mesmo fragmento. Se mais de um classificador indicar que aquele fragmento é do seu tipo, o componente de recuperação deverá testar as duas hipóteses. O importante, nesse caso, é que, pelo menos, um dos classificadores gere uma classificação positiva correta, apesar dos demais falsos-positivos.

6. Conclusão

Neste trabalho foi explorado o uso de técnicas de aprendizagem de máquina baseadas em árvores de decisão aplicadas no problema de classificação de fragmentos para posterior recuperação do arquivo. Em cenários nos quais inexistem informações sobre o sistema de arquivos e, ainda, inexistem o cabeçalho e o final do arquivo, a classificação e a recuperação de arquivos se tornam atividades desafiadoras.

Foi empregado o algoritmo de classificação *J48*, considerado um classificador estável e de amplo uso na área de aprendizagem de máquina, sobre um conjunto de fragmentos extraídos de arquivos de 21 tipos distintos. Como resultado, obteve-se um conjunto de classificadores binomiais especializados para 21 tipos de arquivos e um conjunto de classificadores multinomiais.

O resultado do experimento mostrou que mesmo quando inexistentes a informação do cabeçalho, do final do arquivo e do sistema de arquivos, é possível classificar um fragmento com percentuais de acertos de 98,78% (com classificadores binomiais) e de 86,05% (com classificadores multinomiais). Logo, considera-se que a utilização de tal técnica pode trazer grandes ganhos para o exame pericial ao complementar as demais formas de recuperação de dados.

Devido ao caráter experimental do trabalho, muito ainda precisa ser feito a respeito da (1) análise dos atributos utilizados no experimento, (2) proporção dos tipos de fragmentos do experimento e (3) tratamento de fragmentos de tipos desconhecidos.

Assim, deve-se analisar a necessidade de normalizar ou mesmo discretizar alguns atributos, assim como analisar o custo e o impacto de cada um. Para determinados modelos de classificação, pode-se eliminar alguns desses atributos sem que haja prejuízos significativos aos resultados do modelo.

Pode-se equilibrar a amostragem de registros para o modelo de classificação binária. O experimento atual foi realizado com uma proporção de 4,76% contra 95,24% dos fragmentos, de tal forma que algumas árvores de classificação binária se tornaram "especialistas" em identificar fragmentos que não pertencem ao tipo. Experimentos com uma quantidade reduzida de registros mostrou que, após o ajuste para uma proporção de 50% contra 50%, os modelos de classificação têm uma redução média de 5% na taxa de acerto do algoritmo.

Nesse experimento não é dado um tratamento para os fragmentos de tipos desconhecidos no classificador multinomial e, portanto, inexiste uma classe "outros". Então, se um fragmento de um tipo desconhecido (ex.: `GIF`) for apresentado, o classificador necessariamente vai indicar um dos 21 tipos esperados, gerando um falso-positivo para aquele tipo.

Agradecimentos. Os autores agradecem o apoio da Secretaria Nacional de Segurança Pública (SENASP), do Departamento de Engenharia Elétrica da Universidade de Brasília, da Fundação de Peritos em Criminalística Ilaraine Acácio Arce (FPCIAA), da Associação Brasiliense de Peritos Criminais (ABPC), da Diretoria Técnico-Científica da Polícia Federal e da FINEP (Convênio 01.12.0433.01, Projeto: Defesa Nacional e Segurança Pública) na realização deste trabalho.

Referências Bibliográficas

Axelsson, S. (2010) "The normalized compression distance as a file fragment classifier". Proceedings of the 2010 Digital Forensics Research Conference (DFRWS).

Bell, J. (2014) "Machine Learning: Hands-on for developers and technical professionals". John Wiley & Sons.

Calhoun, W., Coles, D. (2008) "Predicting the types of file fragments". Proceedings of the 2008 Digital Forensics Research Conference (DFRWS).

Cohen, M. I. (2007) "Advanced carving techniques". In Digital Investigation: The International Journal of Digital Forensics & Incident, volume 4, pages 119-128.

Conti, G., Bratus, S., Sangster, B., Ragsdale, R., et al. (2010) "Automated mapping of large binary objects using primitive fragment type classification". Proceedings of the 2010 Digital Forensics Research Conference (DFRWS).

Garfinkel, S. L. (2007) "Carving contiguous and fragmented files with fast object validation". Journal Digital Investigation, volume 4, pages 2-12.

Fitzgerald, S., Mathews, G., Morris, C. and Zhulyn, O. (2012) "Using NLP techniques for file fragment classification". Journal Digital Investigation, volume 9.

Foremost (2016). Disponível em: <http://foremost.sourceforge.net/>.

Li, Q., Ong, A., Suganthan, P., Thing, V. (2010) "A novel support vector machine approach to high entropy data fragment classification". Proceedings of the South African Information Security Multi-Conference.

Penrose, P., Macfarlane, R., and Buchanan, W. J. (2013) "Approaches to the classification of high entropy file fragments". Journal Digital Investigation, volume 10, issue 4, pages 372-384.

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

Roussev, V. and Quates, C. (2013) "File fragment encoding classification-An empirical approach". Journal Digital Investigation, volume 10, pages S69-S77.

Veenman, CJ. (2007) "Statistical disk cluster classification for file carving". In Proceedings of the IEEE 3rd international symposium on information assurance and security, IEEE Computer Society, pages 393–8.

Waikato (2016a) "Weka: Waikato Environment for Knowledge Analysis Tool". Disponível em: <http://www.cs.waikato.ac.nz/~ml/>.

Waikato (2016b) "Weka: Attribute-Relation File Format". Disponível em: <http://weka.wikispaces.com/ARFF>.

Waikato (2016c) "Weka: Decision Tree Algorithms". Disponível em: <http://weka.sourceforge.net/doc.stable/weka/classifiers/trees/package-summary.html>.

