MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

GABRIEL DILLY VIEIRA FURTUOSO
RAFAEL FILIPE DOS SANTOS

APLICAÇÃO DE MACHINE LEARNING NA IDENTIFICAÇÃO DE SISTEMAS CRIPTOGRÁFICOS A PARTIR DE CRIPTOGRAMAS

RIO DE JANEIRO
2020

GABRIEL DILLY VIEIRA FURTUOSO
RAFAEL FILIPE DOS SANTOS

APLICAÇÃO DE MACHINE LEARNING NA IDENTIFICAÇÃO DE SISTEMAS CRIPTOGRÁFICOS A PARTIR DE CRIPTOGRAMAS

Projeto de Final de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Bacharel em Engenharia da Computação.

Orientador(es): José Antonio Moreira Xexéo, D. Sc. do IME
Leandro de Mattos Ferreira, M.Sc do IME

Rio de Janeiro

2020

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 – Praia Vermelha
Rio de Janeiro – RJ CEP: 22290-270

Dilly Vieira Furtuoso, Gabriel; Filipe dos Santos, Rafael.
Aplicação de Machine Learning na Identificação de Sistemas Criptográficos a partir de Criptogramas / Gabriel Dilly Vieira Furtuoso e Rafael Filipe dos Santos. – Rio de Janeiro, 2020.
63 f.

Orientador(es): José Antonio Moreira Xexéo e Leandro de Mattos Ferreira.

Projeto de Final de Curso (graduação) – Instituto Militar de Engenharia, Engenharia da Computação, 2020.

1. Criptografia. 2. DES. 3. RSA. 4. ElGamal. 5. Aprendizado de Máquina. 6. Classificação Binária. 7. SVM. i. Antonio Moreira Xexéo, José (orient.) ii. de Mattos Ferreira, Leandro (orient.) iii. Título

GABRIEL DILLY VIEIRA FURTUOSO
RAFAEL FILIPE DOS SANTOS

Aplicação de Machine Learning na Identificação de Sistemas Criptográficos a partir de Criptogramas

Projeto de Final de Curso apresentado ao Curso de Graduação em Engenharia da Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Bacharel em Engenharia da Computação.

Orientador(es): José Antonio Moreira Xexéo e Leandro de Mattos Ferreira.

Aprovado em Rio de Janeiro, 27 de outubro de 2020, pela seguinte banca examinadora:

Prof. **José Antonio Moreira Xexéo** - D.Sc. do IME - Presidente

---

Prof. **Leandro de Mattos Ferreira** - M.Sc. do IME

Prof. **Julio Cesar Duarte** - D.Sc. do IME

Prof. **Ronaldo Ribeiro Goldschmidt** - D.Sc. do IME

Rio de Janeiro
2020

Para nossos pais e professores, porque devemos respeitar nossas origens;
e para todos os estudantes interessados, porque também devemos cultivar o futuro.

AGRADECIMENTOS

Primeiramente a Deus, pois dEle deriva todo o resto.

Às nossas famílias, por todo apoio ao longo de nossas vidas, que nos ensinou primeiro, incentivando-nos sempre a persistir em nossos objetivos.

Aos nossos professores do Instituto Militar de Engenharia, pela excelente formação ao longo dos últimos cinco anos.

Em Especial aos professores TC José Antonio Moreira Xexéo e Maj Leandro de Mattos Ferreira, que tornaram este trabalho possível, orientando-nos sempre nos momentos de dificuldade.

Por último a todos os nossos amigos, que sempre nos dão força e apoio.

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito.
(Bíblia Sagrada, Romanos 12, 2)

RESUMO

O presente trabalho apresenta técnicas de inteligência artificial aplicadas na identificação de algoritmos de encriptação com base em textos cifrados, problema relevante dentro do processo de criptoanálise. A partir do conjunto de artigos Reuters-21578, construiu-se três bases cifradas para os algoritmos criptográficos DES, RSA e ElGamal e empregou-se o modelo de Aprendizado de Máquina SVM para a elaboração de um sistema de classificação de algoritmos criptográficos composto por três classificadores binários capazes de avaliar se um texto foi cifrado ou não utilizando um determinado algoritmo. Como entradas para os modelos de Aprendizado de Máquina, calculou-se medidas de similaridade e dissimilaridade entre os documentos. Ao final, foi realizada a análise de alguns experimentos realizados com o sistema de classificação, a fim de avaliar a acurácia do sistema, bem como analisar o impacto de alguns fatores nos resultados, como quais medidas escolher ou qual o tamanho da base de treino que deve ser utilizado.

Palavras-chave: Criptografia. DES. RSA. ElGamal. Aprendizado de Máquina. Classificação Binária. SVM.

ABSTRACT

The present work presents artificial intelligence techniques applied in the identification of encryption algorithms based on ciphertext, a relevant problem within the cryptanalysis process. From the set of articles Reuters-21578, three encrypted bases were constructed for the DES, RSA and ElGamal cryptographic algorithms and the SVM Machine Learning model was used to develop a cryptographic algorithm classification system composed of three binary classifiers capable of evaluating whether a text has been encrypted or not using a certain algorithm. As inputs for the Machine Learning models, measures of similarity and dissimilarity between the documents were calculated. At the end, an analysis of some experiments carried out with the classification system was carried out, in order to assess the accuracy of the system, as well as to analyze the impact of some factors on the results, such as which measures to choose or which the size of the training base that should be used.

Keywords: Cryptography. DES. RSA. ElGamal. Machine Learning. Binary Classification. SVM.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DES	Data Encryption Standard
RSA	Rivest–Shamir–Adleman
ECB	Eletronic codebook
CBC	Cipher block chaining
IP	Permutação Inicial
VI	Vetor de Inicialização
ML	Machine Learning
k-NN	K Nearest Neighbor
GAM	Generalized Additive Model
SVM	Support Vector Machine
LDA	Linear Discriminant Analysis

LISTA DE SÍMBOLOS

ϕ	Função Phi de Euler
$\oplus$	Ou Exclusivo
$ord_n(a)$	Ordem de a módulo n
ϵ	Erro
$(\text{mod } n)$	a congruente a b módulo n
$\equiv$	Congruência Modular
$P(A\|B)$	Probabilidade de o evento A ocorrer dada a ocorrência do evento B
log	Logaritmo
v^T	Vetor v transposto

SUMÁRIO

1 INTRODUÇÃO

A expressão *Machine Learning*, Aprendizado de Máquina em Português, foi popularizada pelo engenheiro Arthur Samuel, com seu trabalho *"Some Studies in Machine Learning Using the Game of Checkers"*, publicado no IBM Journal, em Julho de 1959.

Em seu trabalho, Samuel se utiliza do jogo de damas para verificar que um computador é capaz de ser programado de modo a aprender a jogar uma partida de damas melhor do que quem o programou, baseando-se essencialmente nas regras do jogo, e isso tudo num período de tempo relativamente pequeno (5).

No mesmo ano, Arthur Samuel definiu Aprendizado de Máquina como uma área de estudos que fornece a um computador a habilidade de aprender a partir de informações fornecidas previamente, sem ter sido explicitamente programado para isso. Dessa forma, Aprendizado de Máquina é um tipo de inteligência artificial.

Com o desenvolvimento dos computadores e o surgimento da internet, os estudos em Aprendizado de Máquina foram crescendo e surgiram diversos ramos de pesquisa na área, possibilitando a criação de novas técnicas e modelos. Há aplicações em diversas áreas para as técnicas comumente utilizadas e uma delas é no contexto de Segurança da Informação, que será o foco deste trabalho.

Uma das áreas da Segurança da Informação, a Criptoanálise, é a área que estuda meios de recuperar uma mensagem criptografada por meio da detecção de falhas ou padrões em sistemas criptográficos. Dessa forma, é importante o responsável por essas análises conhecer o algoritmo pelo qual a mensagem está criptografada.

Em situações reais, nem sempre se conhece por qual algoritmo um texto criptografado, também chamado criptograma, está criptografado. Nesse sentido, há diversas pesquisas que buscam construir mecanismos para resolver o problema da classificação de criptogramas de acordo com o algoritmo de criptografia utilizado. Tal problema está esquematizado na Figura 1. Porém, nem sempre a classificação direta é facilmente obtida, já que há parâmetros que dificultam o processo, como a utilização de chaves diferentes para criptografar textos com um mesmo algoritmo.

Um primeiro modo de lidar com este problema é por meio de um processo de agrupamento, ou seja, a separação de criptogramas de acordo com características semelhantes (3). Sua esquematização está exemplificada na Figura 2. Solucionando este problema, pode-se conseguir agrupar todos os criptogramas criptografados por um mesmo algoritmo, mas não necessariamente se conhece qual o algoritmo que criptografou as mensagens de um mesmo grupo.

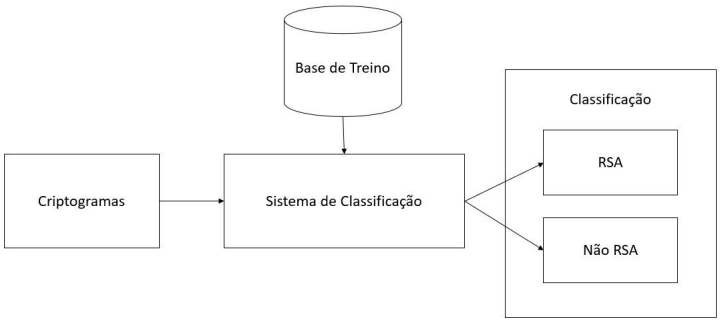

Figura 1 – Esquema de um sistema de classificação para o algoritmo RSA.

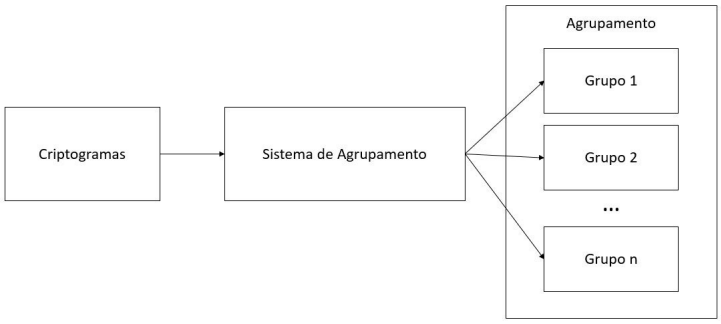

Figura 2 – Esquema de um sistema de agrupamento.

O Aprendizado de Máquina, principalmente por meio de modelos não supervisionados, se apresenta como uma forte ferramenta de apoio na solução destes problemas, possibilitando em alguns casos a resolução do problema de classificação, que é mais complexo que o problema de agrupamento, e trabalhos recentes na área têm comprovado a eficácia da utilização, como será comentado na próxima seção.

1.1 Motivação

Um dos objetivos de estudo da Segurança da Informação é a identificação de algoritmos de encriptação sendo fornecido apenas o texto cifrado. Esta busca pelo algoritmo utilizado constitui uma das etapas da criptoanálise (6).

Existem diferentes técnicas para atacar este problema. Uma primeira abordagem é inicialmente procurar agrupar criptogramas de acordo com certa similaridade, conforme comentado anteriormente. Como exemplos de abordagem para este problema, Carvalho(1) utilizou técnicas de agrupamento e recuperação de informação, separando criptogramas de determinados sistemas criptográficos de acordo com características semelhantes detectadas.

Souza(3), dando continuidade aos estudos de Carvalho(1), apresenta uma ideia semelhante com novas versões de sistemas criptográficos e inclui a utilização de técnicas de inteligência artificial em sua análise, por meio da utilização de redes neurais, para realizar o agrupamento. Além disso, ele faz uma análise de métricas de similaridade e distância

normalmente utilizadas nesses processos.

Apesar de o agrupamento de acordo com a similaridade entre os criptogramas ter sido bem sucedido, os trabalhos acima não solucionaram diretamente o problema da classificação de criptogramas.

No mesmo contexto, em trabalho mais recentemente publicado (7), há a utilização de técnicas de Aprendizado de Máquina para realizar a classificação dos criptogramas. A barreira que ainda não foi vencida aqui é a grande necessidade de processamento e memória.

Ferreira(8) procurou complementar as tentativas anteriores, mas realizando outra abordagem, por meio da utilização de transformadas *Wavelets* combinadas com técnicas de recuperação de informação. Além disso, ele também utilizou Aprendizado de Máquina em seu trabalho. Como um dos resultados obtidos, ele desenvolveu um classificador binário para um dos sistemas criptográficos analisados por ele.

Nota-se então que há pesquisas acadêmicas a respeito do problema de identificação de algoritmos de encriptação com base no texto cifrado. Além disso, a evolução e desenvolvimento de novas técnicas de Inteligência Artificial e Aprendizado de Máquina permitem maior variedade de testes e estudos a respeito do tema.

1.2 Objetivos

O objetivo principal deste projeto é desenvolver um sistema, aplicando-se técnicas de aprendizado de máquinas, com o intuito de obter avanços na solução do problema da criptografia relacionado à classificação de criptogramas de acordo com seus algoritmos. Serão abordados três sistemas criptográficos conhecidos: RSA, DES e ElGamal.

O sistema é composto principalmente por três classificadores binários capazes de dizer se um criptograma foi ou não encriptado por determinado algoritmo, cada um respondendo de acordo com um algoritmo específico. Eles são construídos com base em técnicas de Aprendizado de Máquina estudadas ao longo do projeto.

Foram geradas bases de exemplos de criptogramas de cada sistema criptográfico e, comparando os criptogramas de um mesmo sistema, foi possível extrair informações de similaridade ou dissimilaridade, as quais serviram para avaliar se um criptograma qualquer na entrada do sistema está ou não criptografado com determinado algoritmo. A construção da base, bem como o cálculo das medidas se baseou principalmente em algumas ideias presentes no trabalho de Ferreira(8).

1.3 Justificativa

No contexto do Instituto Militar de Engenharia, responsável por grande parte das pesquisas institucionais na área de Computação, a Segurança da Informação aparece como uma das mais fundamentais para o Exército Brasileiro, já que é de extrema importância para a proteção das informações e dados institucionais.

O tema deste projeto está diretamente alinhado aos objetivos do Setor Cibernético e do Plano Nacional de Defesa, pois une Defesa Cibernética e Criptografia, duas grandes áreas de interesse das Forças Armadas. Os diversos trabalhos e pesquisas relacionados ao tema, como os diversos trabalhos de pós-graduação do Instituto, já apresentados neste projeto, evidenciam o interesse em novos avanços no assunto em questão.

No contexto civil, com os avanços tecnológicos e a rápida evolução da Inteligência Artificial, a qual engloba o Aprendizado de Máquina, diversas empresas estão desenvolvendo áreas especializadas em lidar com problemas relacionados a Aprendizado de Máquina. Um exemplo disso são as áreas de Ciência de Dados.

Porém, o número de profissionais ainda é escasso nesse meio, e o contato desde cedo com este tipo de ferramenta propicia um grande aprendizado e amadurecimento, o qual fará bastante diferença no mercado de trabalho e no desenvolvimento de projetos futuros.

1.4 Metodologia

O projeto se desenvolveu nas seguintes etapas. A primeira etapa do projeto foi a fundamentação teórica: estudou-se os conceitos necessários para o desenvolvimento do projeto, relacionados a Aprendizado de Máquina e Criptografia. Também foi realizada a análise de trabalhos relacionados ao tema proposto.

A segunda etapa foi a modelagem dos Classificadores Binários: definiu-se as etapas da construção dos classificadores binários no contexto em questão, e determinou-se como seria a abordagem para sua implementação, bem como qual o modelo de Aprendizado de Máquina a ser utilizado.

Após a modelagem, analisou-se como seria a construção dos parâmetros a serem utilizados no modelo de Aprendizado de Máquina, optando-se pelas utilização de medidas de similaridade e dissimilaridade para isso.

A próxima etapa foi a implementação do projeto: desenvolveu-se os Classificadores Binários para cada um dos sistemas criptográficos abordados, incluindo a interface com o usuário, os cálculos das medidas utilizadas e a implementação do modelo supervisionado.

Por fim, realizou-se os testes do projeto para analisar os resultados. Muitos testes

também foram realizados em paralelo com a implementação para acompanhar o funcionamento correto das diversas partes do projeto. Para os testes, foi construída uma base de criptogramas, a qual foi usada em todas as etapas do projeto para verificar o correto funcionamento de cada parte.

1.5 Estrutura do texto

O capítulo 2 aborda a fundamentação teórica necessária para o projeto, dividida em dois grandes blocos: Criptografia e Aprendizado de Máquina.

Na parte de Criptografia, há o detalhamento dos conceitos de cifra de blocos e modo de operação, bem como a explicação dos três algoritmos abordados neste projeto: DES, RSA e ElGamal.

Já na parte de Aprendizado de Máquina, aborda-se os conceitos de regressão e classificação no contexto em questão, com destaque para dois modelos de aprendizado de máquina: regressão logística e SVM, o qual será utilizado na construção dos classificadores binários.

Ainda no capítulo 2, aborda-se também algumas medidas de similaridade e dissimilaridade, as quais serão utilizadas nos modelos de Aprendizado de Máquina.

No capítulo 3, descreve-se a proposta para o sistema de classificação a ser desenvolvido, detalhando a funcionalidade de cada parte: a construção do espaço de vetores, o cálculo das medidas e o modelo de Aprendizado de Máquina. Além disso, explica-se como tais partes se conectam umas às outras.

No capítulo 4, é abordada a implementação do sistema, explicando as decisões de projeto tomadas ao longo do trabalho, destacando-se as ferramentas e linguagens utilizadas e as dificuldades encontradas ao longo do desenvolvimento.

No capítulo 5, há a apresentação dos resultados alcançados, bem como uma discussão a respeito deles, com o intuito de analisar a eficácia do sistema como um todo.

Por fim, no capítulo 6, aborda-se as conclusões finais do projeto, retomando os resultados alcançados e apresentando algumas propostas para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são abordados os principais tópicos abordados no trabalho, que se dividem em duas grandes partes: Criptografia, relacionada à Segurança da Informação, e Machine Learning, em português Aprendizado de Máquina.

2.1 Criptografia

A criptografia é a área de estudo dos diversos processos de encriptação e decriptação de mensagens. Os primeiros registros de seu uso são relativos ao período do Império Romano.

De acordo com Stallings(6), entende-se por encriptação o processo de conversão de uma mensagem, a qual é usualmente chamada texto em claro, em uma mensagem codificada, normalmente chamada criptograma. Decriptação é o processo inverso, no qual se restaura o texto em claro a partir do criptograma.

Existem diferentes diferentes modos de realizar os processos de encriptação e decriptação. Cada um desses modos são chamados sistemas criptográficos ou cifras.

Há diversas técnicas para decifrar textos, mesmo que se desconheça o sistema utilizado, e o conjunto de tais técnicas constitui a criptoanálise.

Nas próximas seções, grande parte da teoria foi retirada de Stallings(6), e demais bibliografias consultadas estão reportadas ao longo do texto.

2.1.1 Modelo de Cifra Simétrica

Um esquema de cifra simétrica é caracterizado por cinco partes essenciais:

- Texto em claro: mensagem ou dados originais. É uma das entradas do sistema de encriptação;
- Algoritmo de encriptação: baseado em uma função $E(K, X)$ que produz transformações no texto em claro. Tal função depende do texto em claro e da chave secreta K;
- Chave secreta: é outra entrada do sistema. Independe do texto em claro, mas influencia no processo de encriptação, pois as transformações propiciadas pelo algoritmo dependem dela, de modo que a saída muda de acordo com a chave escolhida;

- Texto cifrado: resultado gerado após o texto em claro passar pelo algoritmo de encriptação;
- Algoritmo de decriptação: baseado na função $D(K, Y)$, inversa do algoritmo de encriptação. Tal função depende da chave secreta K e do texto cifrado.

Nas funções de encriptação e decriptação, as duas principais ações que estão envolvidas na ideia básica de todos os sistemas criptográficos são a substituição e a transposição de símbolos.

É importante que em um processo de compartilhamento de mensagens criptografadas por uma cifra simétrica tanto o emissor como o receptor possuam cópias da chave secreta, de modo que seja possível criptografar e decriptografar mensagens. Esse compartilhamento de chave deve ser feito de modo seguro, para evitar que a comunicação seja lida por um terceiro. Um esquema geral da comunicação baseada em um modelo de cifra simétrica está representado na Figura 3.

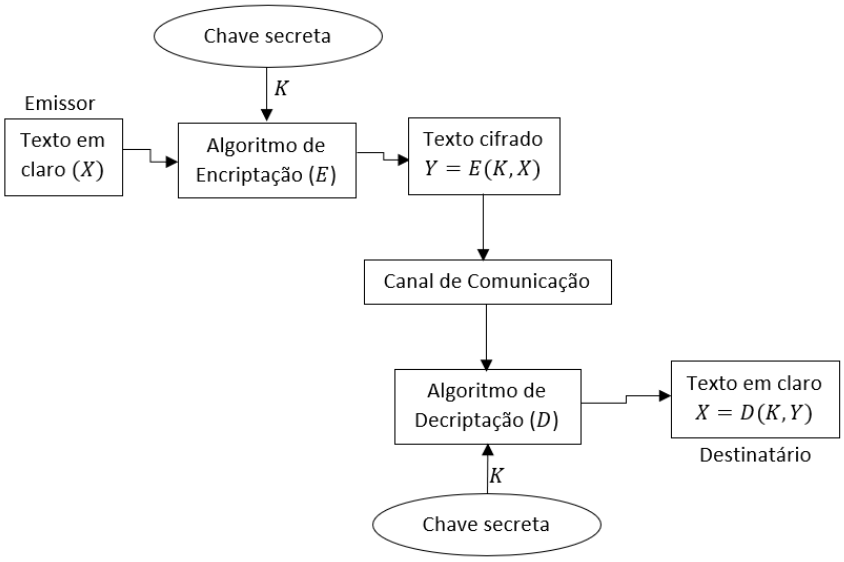

Figura 3 – Modelo simplificado de encriptação simétrica.

Um princípio fundamental de um algoritmo de encriptação é que ele não precisa ficar secreto, mesmo que seja uma informação de grande auxílio em um processo de criptoanálise.

Para ilustrar o esquema acima, o exemplo mais simples e mais antigo de cifra que se conhece é a Cifra de César. Ela é baseada na substituição de cada letra da mensagem pela letra que fica três posições depois, ou seja, A vira D, B vira E e assim sucessivamente.

Associando a cada letra X o número relativo a sua posição no alfabeto, pode-se caracterizar matematicamente a função de encriptação como

$$C = E(3, X) = (X + 3) \pmod{26}$$

A decriptação é dada de modo semelhante:

$$X = D(3, C) = (C - 3) \pmod{26}$$

Pode-se generalizar a Cifra para qualquer deslocamento, trocando o 3 por qualquer k inteiro.

Nota-se que, conhecida uma mensagem criptografada com a Cifra de César, com uma simples criptoanálise por força bruta pode-se encontrar as possíveis mensagens, mesmo sem saber precisamente o deslocamento, testando as 25 possibilidades para o deslocamento k.

Pode-se melhorar a Cifra de César se utilizando de cifras chamadas cifras monoalfabéticas. Nessas cifras, em vez de se deslocar cada letra de uma certa distância, associa-se ao alfabeto $(A, B, C, ..., Z)$ uma permutação qualquer dele $(A_1, A_2, ..., A_{26})$ e então o A_1 corresponde a A, A_2 corresponde a B, e assim sucessivamente.

Dessa forma, ao invés de apenas 25 tentativas para concluir o método de força bruta para decifrar sem saber a chave, seriam necessárias 26! tentativas, o que torna o processo bem mais complexo. Porém, conhecendo-se a língua, uma análise de frequência das letras poderia facilitar a criptoanálise.

As cifras que surgem para impedir a análise de frequência são as cifras poli alfabéticas, que são essencialmente cifras monoalfabéticas aplicadas em sequência. A chave nesse caso define qual regra é escolhida a cada momento na sequência, ou seja, a chave é na realidade uma sequência de chaves.

Os exemplos apresentados servem para ilustrar de modo simples o funcionamento de um modelo de cifra simétrica, mas tais sistemas não foram utilizados no desenvolvimento do projeto. Existem modelos mais sofisticados de cifras simétricas, assim como outros exemplos mais modernos de cifras, alguns dos quais estão comentados nas próximas seções e abordados no desenvolvimento do projeto.

2.1.1.1 Cifras de Blocos

Cifras de bloco são aquelas em que um bloco de texto funciona como uma única estrutura, a qual é utilizada para gerar um texto cifrado com mesmo tamanho. São cifras simétricas, de modo que emissor e receptor compartilham a chave secreta de encriptação simétrica. Usualmente usa-se blocos de 64, 128 ou 256 bits, de modo que normalmente utiliza-se palavras com o alfabeto binário.

O primeiro exemplo de cifra de bloco, que é bastante útil para o entendimento do Data Encryption Standard (DES), é a Cifra de Feistel.

Este modelo foi proposto por Feistel em 1973 e é baseado na execução de duas ou mais cifras simples em sequência de modo a fortalecer a segurança do modelo.

As etapas da cifra são a realização alternada de duas operações: substituição e permutação. Na substituição, há uma função que transforma os elementos do texto de entrada em um outro elemento ou em um grupo de elementos, a depender do tipo de substituição. Já na permutação, uma parte do texto de entrada é trocada por uma permutação dessa parte.

De acordo com Menezes, Oorschot e Vanstone(2), define-se a Cifra de Feistel como uma um mapeamento de um texto em claro de $2n$ bits, divididos em dois blocos de tamanho n (L_0, R_0), em um texto cifrado de $2n$ bits (R_r, L_r), por meio de um processo de r iterações, $r \geq 1$. E para cada $1 \leq i \leq r$, i-ésima iteração (L_{i-1}, R_{i-1}) para (L_i, R_i) da seguinte forma:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i),$$

em que K_i é uma subchave derivada da chave K, f é uma função, que pode também ser uma cifra de produto e $\oplus$ é a operação "ou exclusivo".

Na Figura 4, adaptada de Menezes, Oorschot e Vanstone(2), está ilustrado o processo iterativo da Cifra de Feistel:

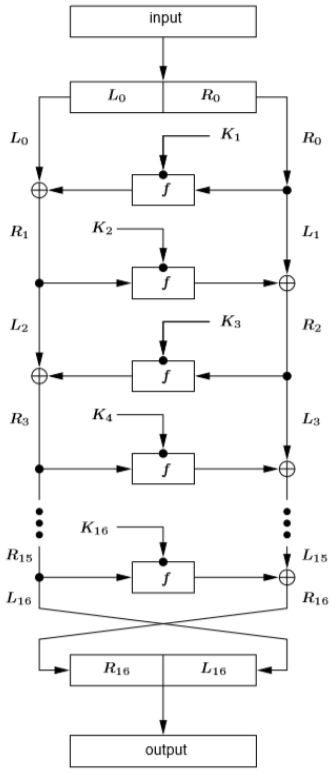

Figura 4 – Estrutura da cifra de Feistel.

Vale ressaltar que f não precisa ser invertível. Para realizar a decriptação basta seguir a estrutura na ordem inversa.

A seguir será discutido o Data Encryption Standard (DES), a principal cifra de bloco que utiliza a estrutura da Cifra de Feistel. O DES foi desenvolvido há cerca de 50 anos e permanece até hoje como uma cifra de bloco bastante utilizada.

Nas seções que seguem, os conceitos foram adaptados de Menezes, Oorschot e Vanstone(2) e Carvalho(1).

2.1.1.2 O Algoritmo Data Encryption Standard (DES)

O Data Encrypton Standard, geralmente referenciado como DES, é uma Cifra de Feistel em que $n = 64$ bits e o tamanho efetivo da chave secreta é de 56 bits. Normalmente são realizadas 16 iterações ($r = 16$).

O DES adota como estrutura base a Cifra de Feistel com uma função f específica e duas permutações, inicial e final, sendo a final a inversa da inicial (IP e IP^{-1}). A função do DES é dada por:

$$f(R_{i-1}, K_i) = P(S(E(R_{i-1}) \oplus K_i)),$$

em que

- $E(X)$ é uma função de expansão, a qual converte a entrada X, que possui 32 bits, em um 48 bits;
- $S(X)$ é uma função de substituições, as quais se fazem através de 8 tabelas, denominadas $S - boxes$, que transformam os 48 bits da entrada X em uma saída de 32 bits;
- $P(X)$ é uma função de permutação, a qual permuta os 32 bits de X.

Precisamente, a chave do DES possui 64 bits, mas os bits 8, 16, 24, ..., 64 são deixados para serem usados como bits de paridade. Porém, para a geração das subchaves K_i, há um algoritmo de geração de chave para transformar a chave de 56 bits em subchaves de 48 bits.

As funções acima baseiam-se em tabelas preestabelecidas. Maiores detalhes sobre as funções, algoritmo de geração de chave e tabelas, podem ser vistos com mais detalhes na bibliografia padrão do DES do National Institute of Standards and Technology(9). Nas Figuras 5 e 6 estão ilustrados a estrutura do DES e a função utilizada, respectivamente.

2.1.2 Criptografia de Chave Pública

O surgimento da criptografia de chave pública é um dos maiores acontecimentos no ramo da Criptografia. Ela surge na tentativa de solucionar dois grandes problemas das cifras simétricas: o problema da distribuição de chaves, já que era necessário uma chave

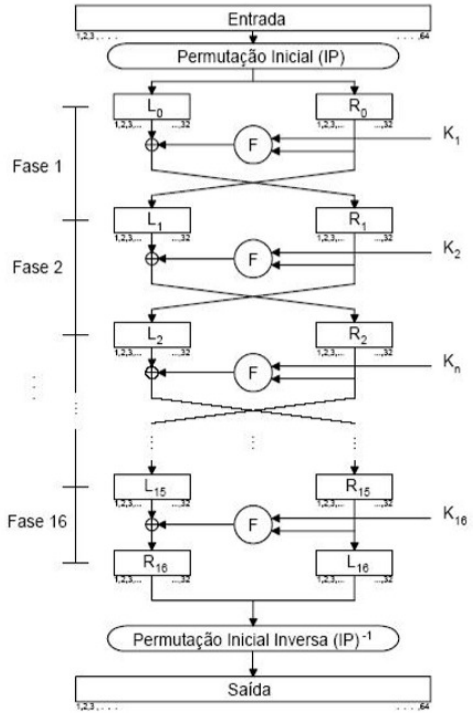

Figura 5 – Estrutura do DES. Retirada de Carvalho(1).

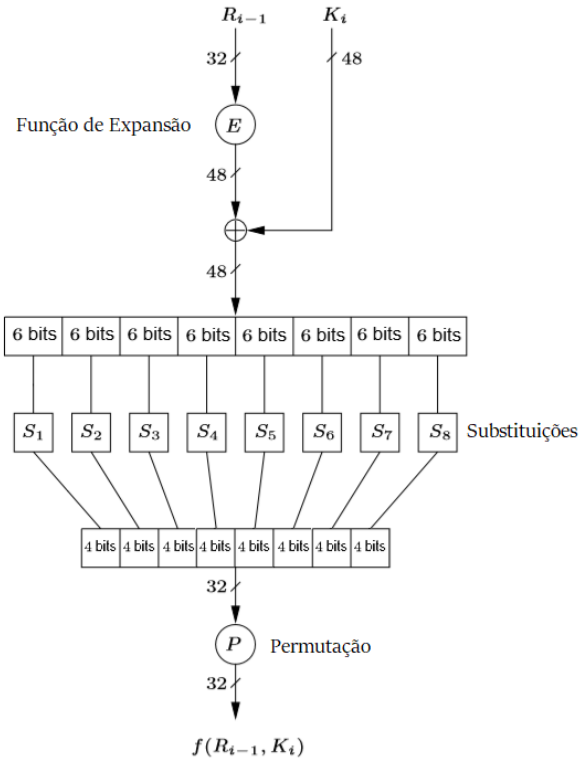

Figura 6 – Função do DES. Adaptado de Menezes, Oorschot e Vanstone(2).

para cada destinatário, e a questão das assinaturas digitais, já que, com o crescimento do uso da criptografia tornava-se cada vez mais necessário que as mensagens trocadas fossem assinadas de alguma forma, assim como se fazia com documentos de papel.

Diferentemente das funções de substituição e permutação usualmente usadas nos sistemas criptográficos simétricos, os algoritmos de chave pública baseiam-se em funções matemáticas.

Em relação às chaves, a criptografia de chave pública utiliza na realidade um par de chaves, cada uma com sua funcionalidade no processo, sendo portanto este tipo de criptografia considerado assimétrico. A utilização de duas chaves se dá pela tentativa de resolver dois problemas associados à encriptação simétrica: a distribuição de chaves e a autenticação do sistema (6).

Nas próximas seções, são analisadas a estrutura e características de dois algoritmos de chave pública, os quais são abordados neste projeto: RSA e ElGamal.

2.1.2.1 O Algoritmo *Rivest–Shamir–Adleman* (RSA)

O algoritmo *Rivest–Shamir–Adleman*, usualmente denotado por sua sigla RSA, é um dos sistemas criptográficos de chave pública mais usados atualmente. O seu nome é devido aos seus criadores Ron Rivest, Adi Shamir e Leonard Adleman.

Suponha que um emissor A deseja enviar uma mensagem a um receptor B. Então B publica um inteiro N, o qual é o produto de dois primos grandes p e q. Eles devem ser grandes de modo que se visto por um terceiro, este conheça N mas não sua fatoração pq. Além de N, o receptor pública um número e, primo com $\phi(N) = (p-1)(q-1)$, sendo $\phi(N)$ a Função Phi de Euler (quantidade de números inteiros positivos primos com N e menores do que N). Então, o par (N, e) é a chave pública do receptor.

A chave privada de B será um inteiro d, inverso multiplicativo de e módulo $\phi(N)$. Este inverso pode ser calculado pelo Algoritmo de Euclides (vale lembrar que B conhece a fatoração de N).

Define-se uma mensagem como um natural $m < N$. Dessa forma, para A criptografar m, basta calcular

$$C \equiv m^e \pmod{N},$$

e então C é a mensagem cifrada.

O receptor recebe C e recupera m com sua chave privada d fazendo:

$$C^d \equiv m \pmod{N}.$$

Para verificar que de fato vale a igualdade acima, basta notar que como d e e são pares de inversos multiplicativos módulo $\phi(N)$, então $ed = \phi(N)k + 1 = (p-1)(q-1)k + 1$,

para algum inteiro k. Logo:

$$C^d \equiv (m^e)^d \equiv m^{ed} \equiv m^{(p-1)(q-1)k+1} \equiv (m^{p-1})^{(q-1)k} m \equiv m \pmod{p}.$$

em que na última passagem utilizamos o Teorema de Fermat: sendo m um inteiro e p um primo tais que $\text{mdc}(m, p) = 1$, então $m^{p-1} \equiv 1 \pmod{p}$.

Observe que se p divide m, não teríamos $\text{mdc}(m, p) = 1$ e então não poderíamos usar o teorema, mas veja que a igualdade continua válida, já que ficaria 0 em ambos os lados. Podemos então usar o mesmo raciocínio para q e então verificamos que de fato vale a igualdade $C^d \equiv m \pmod{N}$.

A explicação do algoritmo foi adaptada de Martinez et al.(10) e mais detalhes sobre a pré-codificação (transformar o texto em claro em número) pode ser visto em Coutinho(11).

A força do RSA se dá na dificuldade de se fatorar números naturais muito grandes. Existem algoritmos polinomiais para testar primalidade, assim como para as demais tarefas necessárias para decriptografar.

Porém, ainda não se conhecem algoritmos polinomiais para se fatorar números muito grandes, de modo que a criptoanálise do RSA se torna bastante complexa. Existem diversas pesquisas em matemática para descobrir se existe ou não tal algoritmo. Este problema inclusive tem relação direta com um dos mais importantes problemas em aberto da matemática $P \neq NP$ (10).

2.1.2.2 Sistema Criptográfico ElGamal

Taher Elgamal, em 1984, desenvolveu um sistema criptográfico baseado na dificuldade de resolução do problema de logaritmo discretos, ou seja, na dificuldade em encontrar um inteiro x tal que, dados a, b, n inteiros, valha:

$$a^x \equiv b \pmod{n}.$$

Para entender este algoritmo é necessário entender o conceito de ordem e raiz primitiva. Os conceitos matemáticos abordados aqui são retirados de Martinez et al.(10).

Define-se a ordem de um inteiro a módulo n como o menor inteiro positivo t tal que $a^t \equiv 1 \pmod{n}$. Normalmente denota-se $t = ord_n(a)$.

O Teorema de Euler-Fermat diz que se a e m são inteiros tais que $\text{mdc}(a, m) = 1$, então $a^{\phi(m)} \equiv 1 \pmod{m}$, em que $\phi(m)$ é a função phi de Euler, já citada anteriormente. Nota-se que é uma extensão do Teorema de Fermat, já que aquele se reduz a este quando m é primo.

Quando se tem $ord_n(a) = \phi(n)$, diz-se que a é uma raiz primitiva módulo n. O caso trabalhado aqui será n primo e, neste caso, um fato importante é que a raiz primitiva

é um gerador do conjunto $\{1, 2, 3, ..., p-1\}$. Este fato é importante para a caracterização do Algoritmo de ElGamal.

De acordo com Stallings(6), para construir o algoritmo escolhe-se inicialmente um número primo q e uma raiz primitiva α módulo q. Em seguida, para gerar as chaves públicas e privadas, realiza-se os seguintes passos:

- Um indivíduo A escolhe um inteiro aleatório x_A, com $1 < x_A < q - 1$;
- Define-se $y_A = \alpha^{x_A} \pmod{q}$;
- A chave privada de A é x_A e a chave pública é a tripla $[q, \alpha, y_A]$.

Para se encriptar uma mensagem, qualquer usuário com as chaves de A pode encriptar uma mensagem e enviá-lo. Realiza-se os seguintes passos:

- Representa-se a mensagem por um inteiro m, com $0 \leq m \leq q - 1$. Caso seja maior, divide-se a mensagem em bloco em blocos;
- Em seguida, escolhe-se um inteiro aleatório k, com $1 \leq k \leq q - 1$.
- Calcula-se $K = y_A^k \pmod{q}$, que é também tratada como uma chave.
- A mensagem é então encriptada como o par (C_1, C_2) definidos como:

$$C_1 = \alpha^k \pmod{q} \quad \text{e} \quad C_2 = Km \pmod{q}.$$

Para decriptografar, o indivíduo A recupera a mensagem do seguinte modo:

- Recupera a chave K calculando-se $K = (C_1)^{x_A} \pmod{q}$;
- Recupera-se m com $m \equiv C_2 K^{-1} \pmod{q}$, sendo K^{-1} o inverso multiplicativo de K (mod q).

Um indivíduo que intercepta uma mensagem m, para decriptografá-la, precisaria encontrar x_A. Como ele conhece α e y_A, isso equivale a procurar a solução x de $\alpha^x \equiv y_A \pmod{q}$, que é justamente o problema do logaritmo discreto citado anteriormente.

Rigorosamente falando, α não precisa ser uma raiz primitive por alguma particularidade do método. A razão é puramente relacionada à segurança do sistema. Se α não for raiz primitiva, o período de $\alpha, \alpha^2, \alpha^3, ... \pmod{q}$ pode ser bem pequeno, diminuindo o número de tentativas para solucionar o problema do logaritmo discreto associado. Por outro lado, se α for raiz primitiva, é garantido que o período será $p - 1$, ou seja, para completar a criptoanálise será necessário percorrer todas as possibilidades para x_A, dificultando o processo.

2.1.3 Modos de Operação

As definições para os conceitos abordados nas seções de modo de operação foram retirados de Stallings(6).

Quando se utiliza uma cifra de bloco, os tamanhos da entrada e da saída são iguais a um mesmo valor n. Caso deseja-se transmitir uma mensagem com mais de n bits, uma solução é quebrar o texto em diversos blocos de tamanhos menores ou iguais a n e continuar utilizando a cifra.

Porém, quando se criptografa diversos textos seguidos com um mesmo algoritmo e mesma chave, a preocupação com a segurança aumenta, já que se torna mais fácil de determinar as características do algoritmo. Para solucionar essa questão, definem-se os modos de operação, que são técnicas que permitem confidencialidade a criptografia de textos arbitrariamente grandes.

Existem diversos modos de operação, cada um atendendo certas aplicações das cifras de blocos. A análise nesta seção se restringirá a dois modos de operação específicos: o modo *eletronic codebook* (ECB) e o modo *cipher block chaining* (CBC).

2.1.3.1 Modo *Eletronic Codebook* (ECB)

É o mais simples dos modos de operação e utiliza a ideia citada no início da seção anterior. Neste modo o texto em claro é dividido em blocos e cada bloco de tamanho n bits é criptografado individualmente, utilizando-se a mesma chave para todos os blocos. Caso o último bloco tenha menos do que n bits, é necessário expandi-lo.

Dessa forma, textos em claro idênticos fornecem textos cifrados idênticos. O nome deste modo vem justamente deste fato, já que seria possível fazer um *codebook*, em que cada linha relaciona um possível texto em claro de tamanho n com seu código correspondente.

Este método é normalmente utilizado com quantidade pequena de dados, não sendo recomendado para utilização em aplicações mais complexas. A Figura 7 mostra o diagrama do funcionamento da encriptação no modo ECB.

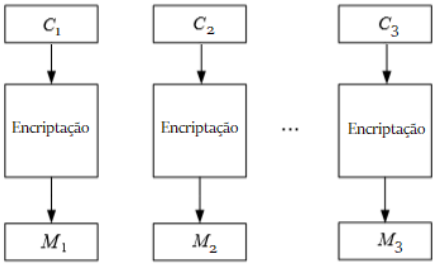

Figura 7 – Estrutura da encriptação no modo ECB - adaptado de Souza(3).

2.1.3.2 Modo *Cipher Block Chaining* (CBC)

O ECB possui algumas fragilidades, já que o fato de o texto em claro e o criptograma possuírem o mesmo tamanho facilita a determinação de alguns padrões. Para solucionar tais problemas do ECB, uma possibilidade é que blocos com mesmo texto em claro sejam criptografados com textos cifrados distintos.

Uma solução encontrada foi por meio do modo de encadeamento de bloco de cifra, do original em inglês *cipher block chaining*. A sua estrutura é caracterizada pelo encadeamento da sequência de blocos de n bits, de modo que o texto cifrado relativo ao bloco anterior é operado com o bloco de entrada atual pela operação "ou exclusivo". A chave é a mesma em todas as etapas e, caso o último bloco termine com menos de n bits, assim como no modo anterior, é necessário expandi-lo. Para o primeiro bloco, normalmente se utiliza um vetor de inicialização (VI), que deve ser secreto, conhecido apenas pelo emissor e receptor.

Fazendo dessa forma, blocos iguais não geram mais criptogramas iguais, aumentando assim a segurança do sistema.

Um questionamento interessante é como fazer para decriptografar no modo CBC. Para resolver, basta passar cada bloco pelo algoritmo de encriptação e então operar o resultado com o texto cifrado relativo ao bloco anterior. A Figura 8 mostra o diagrama da estrutura do modo CBC.

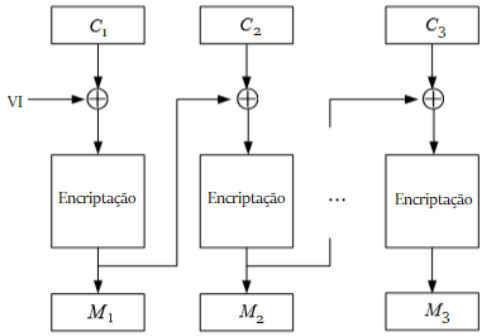

Figura 8 – Estrutura da encriptação no modo CBC - adaptado de Souza(3).

2.2 Machine Learning

De acordo com Bishop(12), os problemas de Aprendizado de Máquina, ou ainda *Machine Learning*, podem ser classificados em algumas categorias, das quais destacam-se as seguintes:

- Aprendizado supervisionado: o computador recebe entradas e saídas de um processo e seu objetivo é mapear as entradas às saídas. Para isso, utiliza-se um conjunto de

exemplos para treinamento;

- Aprendizado não supervisionado: o computador recebe apenas entradas e seu objetivo é clusterizá-las de acordo com características observadas;
- Aprendizado por reforço: o computador é submetido a um ambiente dinâmico, no qual o programa realiza determinado trabalho e, de acordo com erros e acertos, ele é recompensado ou punido. O objetivo do computador é maximizar as recompensas.

No presente trabalho, será utilizado o aprendizado supervisionado, que se baseia na construção de modelos a partir de entradas e saídas conhecidas de um conjunto de dados de treinamento. Após a obtenção dos modelos utilizando diversos algoritmos de ML, é possível predizer ou estimar uma saída com base em uma ou mais entradas de um novo conjunto de dados.

Com base na abordagem de James et al.(4), pode-se denotar as variáveis de entrada de um problema qualquer pelo símbolo X, sendo cada variável expressa por X_i. Elas também podem ser chamadas de preditores, variáveis independentes, *features* ou simplesmente variáveis. Para a variável de saída, também chamada de resposta ou variável dependente, utiliza-se o símbolo Y.

Cada variável pode ser caracterizada como quantitativa ou qualitativa. Variáveis quantitativas assumem valores numéricos como por exemplo a idade de uma pessoa, sua renda ou uma distância. Já as variáveis qualitativas podem assumir um valor dentre um conjunto de K possibilidades, chamadas de classes ou categorias, como por exemplo o gênero de uma pessoa (masculino e feminino), a cor dos olhos de um pessoa (azul, marrom ou verde) ou, no caso do presente trabalho, se um criptograma foi gerado a partir de um determinado sistema criptográfico (sim ou não).

No geral, pode-se dividir os problemas de aprendizado supervisionado em problemas de regressão, que possuem respostas quantitativas (como a regressão linear) e em problemas de classificação, que possuem respostas qualitativas (como a regressão logística).

2.2.1 Regressão

Os conteúdos desta e das próximas duas subseções foram adaptados de James et al.(4).

Para a observação de uma resposta quantitativa Y com p diferentes preditores, X_1, X_2,..., X_p, pode-se assumir a existência de uma relação da forma:

$$Y = f(X) + \epsilon$$

Sendo f uma função fixa e desconhecida de $X = (X_1, X_2, ..., X_p)$, e ϵ um erro independente de X com média zero.

Tipicamente, busca-se predizer Y a partir de uma função $\hat{f}$, aproximação de f, obtendo predições da forma $\hat{Y}$.

$$\hat{Y} = \hat{f}(X)$$

2.2.2 Classificação

Os problemas de classificação dentro do Aprendizado de Máquina são resolvidos por modelos cujo objetivo é predizer respostas qualitativas, ou seja, classificar uma observação em um conjunto de categorias ou classes. Geralmente os métodos utilizados, nesse caso, primeiro determinam a probabilidade de cada uma das categorias da variável qualitativa para depois classificar a observação.

Dentre as técnicas de classificação ou classificadores, alguns exemplos são a regressão logística, a análise discriminante linear (LDA), o classificador de Bayes e o classificador k-NN (*K Nearest Neighbor*), este último utilizado por Carvalho(1) e Ferreira(8).

Existem também outros métodos para classificação que são computacionalmente intensivos como modelos aditivos generalizados (GAMs), árvores de decisão, florestas aleatórias e máquinas de vetores de suporte (SVMs). Este último foi o modelo escolhido para ser utilizado neste projeto.

2.2.3 O Modelo *Support Vector Machine* (SVM)

O *Support Vector Machine*, comumente chamado pela sigla SVM, é um modelo de aprendizado de máquina supervisionado. O objetivo principal deste modelo é determinar fronteiras que separem o espaço analisado em regiões, com base em certas medidas de similaridades. Dessa forma, dado um conjunto de dados que se deseja classificar, tais medidas determinam as semelhanças entre eles, gerando regiões do espaço em que os pontos possuem características similares, permitindo assim uma classificação do conjunto.

Existem algumas abordagens para a utilização do SVM. A mais usual é em classificação binária, na qual se deseja dividir o espaço em duas regiões, como mostra a Figura 9. A maior dificuldade em todas elas é encontrar as funções, também chamadas de *kernels*, ideais que geram as fronteiras entre as regiões. Algumas ideias são apresentadas na literatura, como a utilização de um hiperplano optimal, uma estrutura polinomial ou até mesmo uma esfera. Todas elas são respostas para algum problema de otimização, o qual geralmente pode ser modelado analiticamente. Neste projeto nos restringiremos à modelagem utilizando um hiperplano.

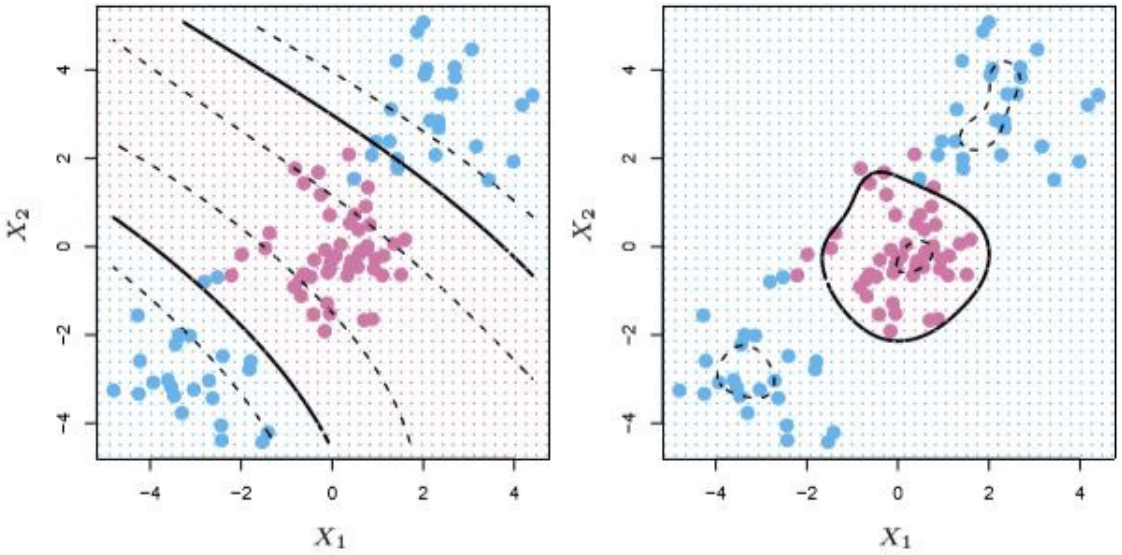

Figura 9 – Resultados gerados por um modelo de SVM - retirado de James et al.(4).

2.2.3.1 Separação Utilizando um Hiperplano

Esta é a abordagem mais simplificada do SVM. Sendo o espaço $(p-1)$-dimensional, um hiperplano é definido por:

$$\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p = 0.$$

Um problema de classificação pode ser definido como segue. Considera-se um conjunto $P_1, P_2, ..., P_n$ de pontos de treinamento, cada um pertencente a uma dentre duas classes. Diremos que as variáveis $y_1, y_2, ..., y_n \in \{-1, 1\}$ representam a qual classe (-1 ou 1) os pontos pertencem. Dado um ponto de teste P^*, deseja-se classificá-lo de acordo com uma das duas classes.

No projeto em questão, a variável y_i pode indicar se um certo criptograma está ou não criptografado com determinado algoritmo, por exemplo.

Para determinar um hiperplano que divide $P_1, P_2, ..., P_n$, sendo $P_i = (x_{i1}, x_{i2}, ..., x_{in})^T$, pode-se estudar soluções $\beta_0, \beta_1, ..., \beta_p$ tais que:

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) > 0,$$

para $i = 1, 2, ..., n$.

Caso haja soluções, existe um hiperplano que divide da forma desejada. Porém, deve-se buscar uma solução máxima, que mantém os pontos de classes diferentes o mais distantes possíveis. Isso aumenta as chances de um novo ponto de teste ser corretamente classificado.

Para isso, o objetivo deve ser resolver o seguinte problema de otimização:

$$\text{Maximizar} M,$$

$$\text{sujeito a } \sum_{j=1}^{p} \beta_j^2 = 1, \text{ sendo}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M, \text{ para } i = 1, 2, ..., n.$$

A solução deste problema é aquela que maximiza a distância do ponto mais próximo ao hiperplano. Para exemplificar, considerando a Figura 10, o hiperplano ótimo é o que maximiza a menor das distâncias destacadas.

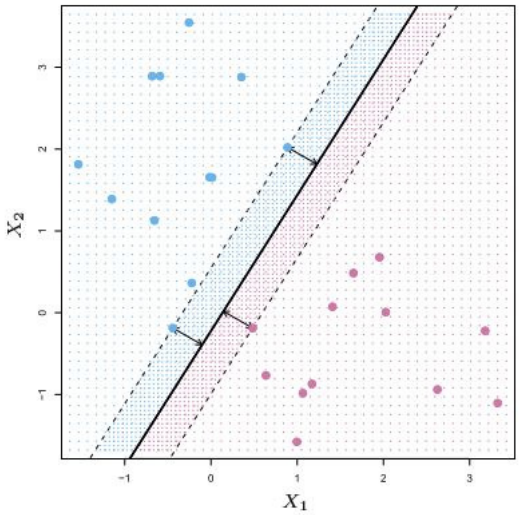

Figura 10 – Exemplo de hiperplano que classifica um conjunto - retirado de James et al.(4).

Uma dificuldade é que nem sempre existe um hiperplano. Outro problema é quando a adição de um simples ponto de treinamento causa uma variação brusca no hiperplano optimal. Dessa forma pode ser necessário fazer adaptações para melhorar os resultados. Uma delas é construir um hiperplano, normalmente chamado *soft margin classifier* que permite que alguns dentre $P_1, P_2, ..., P_n$ fiquem do lado errado da classificação. Ele é solução de uma leve variação do problema de otimização acima e pode ser visto com mais detalhes em James et al.(4).

2.3 Medidas de Similaridade e Dissimilaridade

Para construir um Classificador Binário, é necessário calcular algumas medidas entre os criptogramas analisados, as quais funcionarão como parâmetros nos modelos de aprendizado de máquina utilizados neste projeto.

De acordo com Souza(3), a similaridade é o valor que indica o quão associados dois objetos estão. Analogamente, a dissimilaridade, também chamada de distância, indica o quanto dois objetos são divergentes, de acordo com Teknomo(13). Dessa forma, torna-se interessante adotar medidas baseadas em similaridade ou dissimilaridade para distinguir um objeto de outro.

Nas próximas seções são comentadas algumas medidas de similaridades ou dissimilaridade, as quais foram utilizadas pelos classificadores projetados. As medidas de similaridade e dissimilaridade foram adaptadas de Manning e Schütze(14) e Souza(3).

Souza(3), em seu trabalho, propõe um método estatístico para avaliar como as medidas influenciam nos resultados dos processos de agrupamento, concluindo que de fato algumas medidas podem gerar resultados melhores. Em todo caso, neste projeto, estão implementadas todas as medidas apresentadas, para que haja a possibilidade de escolha pelo o usuário, possibilitando diversos tipos de experimentos.

Como está detalhado mais a frente, os dados analisados são representados por vetores de números inteiros. Dessa forma, considere dois objetos e sejam

$$X = (x_1, x_2, \ldots, x_n) \text{ e } Y = (y_1, y_2, \ldots, y_n)$$

os vetores que os representam, sendo $x_i, y_i \in \mathbb{Z}$.

2.3.1 Medida do Cosseno entre Dois Vetores

O cosseno entre X e Y, denotado por $\cos(X, Y)$, é uma medida de similaridade que assume, nesse trabalho, valores no intervalo $[0, 1]$ e é definida como:

$$\cos(X, Y) = \frac{\sum_{i=1}^{n} x_i \times y_i}{\sqrt{\left(\sum_{i=1}^{n} x_i^2\right)\left(\sum_{i=1}^{n} y_i^2\right)}}.$$

2.3.2 Coeficiente Simple-Matching

O coeficiente Simple-Matching, que denotaremos por $C_{SM}(X, Y)$ e que está no intervalo $[0, \infty[$, é uma medida de similaridade dada pelo produto escalar entre os vetores X e Y, ou seja, é definido por:

$$\mathrm{C}_{SM}(X, Y) = \sum_{i=1}^{n} x_i \times y_i.$$

2.3.3 Coeficiente Dice

O Coeficiente Dice entre X e Y, que denotaremos por $C_D(X, Y)$ e que está no intervalo $[0, 1]$, é uma medida de similaridade dada pela seguinte razão:

$$\mathrm{C}_D(X, Y) = \frac{2\sum_{i=1}^{n} x_i \times y_i}{\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} y_i^2}.$$

2.3.4 Coeficiente Jaccard

O Coeficiente Jaccard entre X e Y, que denotaremos por $C_J(X, Y)$ e que está no intervalo $[0, 1]$, é uma medida de similaridade definida pela razão abaixo:

$$\mathrm{C}_J(X, Y) = \frac{\sum_{i=1}^{n} x_i \times y_i}{\sum_{i=1}^{n} x_i^2 + \sum_{i=1}^{n} y_i^2 - \sum_{i=1}^{n} x_i \times y_i}.$$

2.3.5 Distância Euclidiana

A Distância Euclidiana entre X e Y, que denotaremos por $D_E(X, Y)$, é uma medida de dissimilaridade que está no intervalo $[0, \infty[$ dada por:

$$\mathrm{D}_E(X, Y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

2.3.6 Distância Manhattan

A Distância Manhattan entre X e Y, que denotaremos por $D_M(X, Y)$, é uma medida de dissimilaridade que está no intervalo $[0, \infty[$ dada por:

$$\mathrm{D}_M(X, Y) = \sum_{i=1}^{n} |x_i - y_i|.$$

3 PROPOSTA PARA O SISTEMA DE CLASSIFICAÇÃO

Nesta seção será abordada a proposta geral do sistema de classificação desenvolvido, destacando as partes pelas quais ele é composto, bem como o funcionamento de cada uma delas.

3.1 Caracterização do Espaço de Palavras

Inicialmente, o usuário do sistema utiliza como entrada uma base de criptogramas, como por exemplo uma base de documentos criptografados, alguns das quais ele deseja determinar com qual algoritmo está criptografado. Esta base é dividida em dois grupos: base de teste e base de treinamento, cada uma delas usada em determinada etapa da classificação.

A primeira etapa do sistema é definida pela leitura de cada documento e posterior transformação de cada um deles em estruturas que facilitem a análise dos criptogramas de acordo com as medidas que serão necessárias calcular.

Para tal, a estrutura utilizada é a representação vetorial para construir os objetos correspondentes a cada documento, gerando um espaço de vetores. Optou-se por esta modelagem devido a sua utilização nos trabalhos de Souza(3) e Ferreira(8), que alcançaram bons resultados.

No que segue, entende-se por *palavra* uma sequência de bits específica, sendo todas as palavras com um mesmo tamanho. Esse tamanho também é um parâmetro importante na construção dos experimentos. A sua escolha baseou-se nos estudos de Ferreira(8) e Souza(3), que conseguiram realizar experimentos com palavras de diferentes tamanhos (8, 16, 32 ou 64 bits). Uma das conclusões é que, dependendo da abordagem utilizada, a diminuição das palavras pode possibilitar uma melhora nos resultados, a depender de fatores como o tamanho das cifras e o número de repetições das palavras.

O principal motivo pelo qual a utilização desta abordagem com a utilização de palavras é interessante é o fato de que a linguagem natural em qualquer idioma possui certas redundâncias, as quais sobrevivem ao processo criptográfico, conforme explica Shannon(15).

Cada documento é então caracterizado como um vetor de tamanho n, em que n é o tamanho do domínio de palavras existentes ao se analisar todos os documentos, ou seja, cada coordenada do vetor corresponde a uma palavra existente em algum dos arquivos. O valor de uma coordenada é definido como a quantidade de vezes em que a palavra

correspondente a esta coordenada aparece no documento.

Para determinar o valor de n e os valores das coordenadas, pode-se percorrer os documentos e, para cada um deles, gerar uma espécie de histograma, que contabiliza todas as palavras que vão aparecendo conforme o documento é lido. No caso de palavras menores, uma outra maneira é adotar n como o total de palavras existentes, ou seja, 2^t, sendo t o tamanho da palavra, de modo que às palavras que não apareceram atribui-se o valor 0 na coordenada correspondente. A Figura 11 destaca as etapas da construção da representação vetorial.

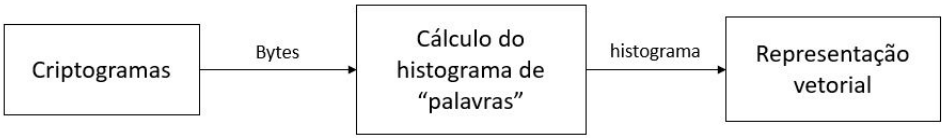

Figura 11 – Construção da representação vetorial.

Para exemplificar, suponha que pretende-se utilizar palavras de 3 bits e três documentos estejam criptografados com as seguintes sequências de bits:

Documento 1: 001 111 110 101 101 011

Documento 2: 011 110 111 001 110 000

Documento 3: 001 110 111 001 111 110

Dessa forma, nota-se que n vale 6, pois aparecem as palavras 000, 001, 011, 101, 110 e 111. Portanto, os vetores correspondentes a cada documento são (as coordenadas seguem a ordem crescente das palavras):

Documento 1: $(0, 1, 1, 2, 1, 1)$

Documento 2: $(1, 1, 1, 0, 2, 1)$

Documento 3: $(0, 2, 0, 0, 2, 2)$

Considerando a segunda maneira, que pode se adotada já que 3 é pequeno, tem-se $n = 2^3$ e apenas acrescenta-se 0 nas coordenadas correspondentes a 010 e 100, que são as palavras que não aparecem. Nesse caso, a i-ésima coordenada corresponde à palavra equivalente a representação de i em binário.

Documento 1: $(0, 1, 0, 1, 0, 2, 1, 1)$

Documento 2: $(1, 1, 0, 1, 0, 0, 2, 1)$

Documento 3: $(0, 2, 0, 0, 0, 0, 2, 2)$

3.2 Cálculo das Medidas de Similaridade ou Dissimilaridade

As medidas de similaridade ou dissimilaridade são calculadas entre cada um dos possíveis pares de documentos que estão na base de treinamento, já representados como vetores. Após todos os cálculos, é gerada uma matriz para cada medida, sendo cada entrada a_{ij} dessa matriz definida como o valor da medida entre o documento i e o documento j. Nota-se que ela é uma matriz simétrica. Então usualmente ela é representa apenas por sua parte triangular superior. Um exemplo de matriz é apresentado na Tabela 1 e a Figura 12 mostra as etapas do cálculo das medidas.

Serão implementadas as medidas de similaridade e dissimilaridade que constam na fundamentação teórica e será oferecido ao usuário a possibilidade de escolha a respeito de qual medida utilizar.

Documentos	1	2	3
1	1	0.75	0.34
2	0.75	1	0.58
3	0.34	0.58	1

Tabela 1 – Exemplo de matriz de similaridade ou dissimilaridade.

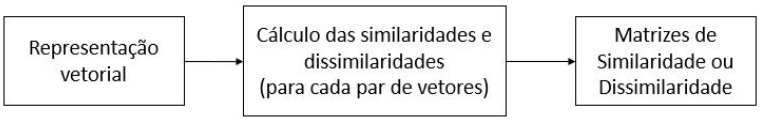

Figura 12 – Geração das matrizes de similaridade e dissimilaridade.

3.3 Aplicação do Modelo de Aprendizado de Máquina

O modelo proposto é o SVM, já descrito no Capítulo 2, com *kernel* linear, ou seja, um hiperplano. A base de treinamento funcionará para a determinação dos dados de treino necessários para a implementação do SVM. Estes dados de treinamento estão todos com os respectivos algoritmos de criptografia identificados e cada um deles caracterizará um ponto no plano, e o hiperplano separador será então definido com base nestes pontos. A base de teste é composta pelos documentos que de fato se quer identificar, e portanto estes não estão rotulados.

As medidas de similaridade ou dissimilaridade serão utilizadas como parâmetros no SVM. O usuário escolhe as medidas que deseja utilizar no modelo e então cada documento passará a ser representado no espaço do SVM por um ponto, em que cada coordenada representa uma medida. Para cada documento, cada coordenada é calculada pela média entre os valores obtidos pelo cálculo da medida entre este documento e os demais documentos da base de treino criptografados com algoritmo que o classificador

determina, ou seja, no caso do classificador RSA, a média será feita com os documentos criptografados com RSA.

De modo mais claro, se o classificador é do algoritmo X, escolhe-se um certo conjunto de medidas e, para cada medida m e para cada documento de treino d, calcula-se a média aritmética dos valores da medida m entre d e cada documento de treino criptografado com o algoritmo X. Variando m sobre as medidas escolhidas, cada documento d será interpretado pelo classificador através de um vetor de médias gerado por esses cálculos.

Para exemplificar, na Tabela **??** mostra-se um exemplo com 3 documentos de teste ($d = 1, 2, 3$) para um classificador do algoritmo RSA utilizando o coeficiente Dice e a Distância Euclidiana como medidas de similaridade e dissimilaridade. Esse classificador possui 4 documentos criptografados com RSA como base de treino. Assim, é possível calcular 12 valores de similaridade entre eles e os documentos de teste para cada medida. Note que para um mesmo documento de teste, os valores obtidos são bem próximos, podendo ser resumidos como uma única variável de entrada para o modelo SVM a partir de uma média aritmética simples.

	Coeficiente Dice					Distância Euclidiana				
	1	2	3	4	Média	1	2	3	4	Média
1	0.9938	0.9945	0.9941	0.9941	0.9941	1740.7	1646.6	1707.3	1709.4	1701.0
2	0.9957	0.9960	0.9958	0.9957	0.9958	1457.8	1392.9	1430.1	1444.5	1431.3
3	0.9985	0.9986	0.9986	0.9984	0.9985	856.5	816.6	816.5	883.9	843.4

Tabela 2 – Exemplo de cálculo das médias do coeficiente Dice e da distância euclidiana para palavras de 8 bits de um classificador RSA

Portanto, dado um documento de treino como entrada no sistema, calculam-se as médias como citado, obtendo representações dos documentos em vetores, como por exemplo nos 3 documentos de teste abaixo, com cada média correspondendo a uma dimensão. Estes vetores são utilizados para alimentar o SVM e treinar o modelo para predizer com qual algoritmo novas observações foram criptografadas:

$$\text{Documento de teste 1: } (0.9941, 1701.0)$$
$$\text{Documento de teste 2: } (0.9958, 1431.3)$$
$$\text{Documento de teste 3: } (0.9985,\ 843.4)$$

3.4 Esquema do Sistema de Classificação

Os classificadores binários são uma possível solução para o problema de classificação dentro do aprendizado supervisionado e são caracterizados pela presença de um resposta qualitativa do tipo binária, que pode assumir dois valores (sim e não). No caso deste projeto será um criptograma ter sido ou não criptografado com determinado algoritmo criptográfico.

Com base nas seções anteriores deste capítulo, nota-se que o Classificador Binário depende de algumas decisões de projeto importantes, como por exemplo qual será o tamanho de uma palavra a ser utilizada para definir o espaço de palavras ou quais as medidas de similaridade ou dissimilaridade a serem utilizadas.

A Figura 13 mostra o esquema geral proposto para a implementação do sistema de classificação. Como o esquema sugere, cada documento que se deseja determinar o algoritmo pelo qual foi criptografado passará pelos três classificadores para então se determinar a resposta do sistema. Um problema de ambiguidade pode surgir quando dois ou mais classificadores retornam "sim" como resposta, ou seja, mais de um classificador aponta o documento como criptografado pelo respectivo algoritmo.

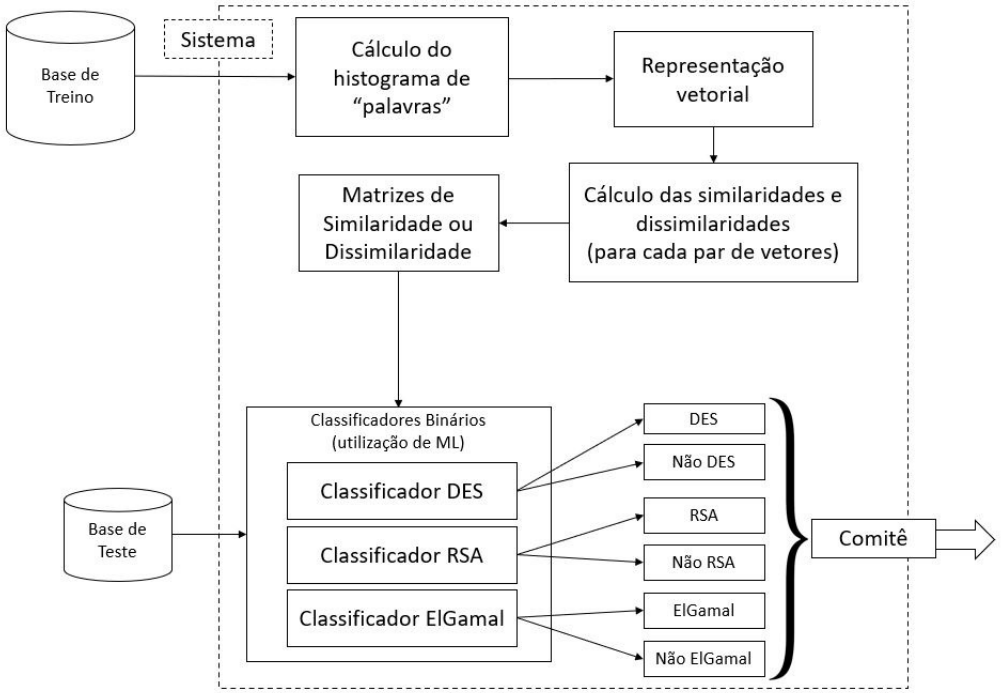

Figura 13 – Esquema geral do Sistema de Classificação

Portanto, torna-se necessária uma etapa extra para analisar qual será o resultado definitivo, já que é necessário que o classificador retorne um único resultado. Essa etapa é composta por um comitê, que decidirá qual dos resultados será aceito. Tal avaliação pode ser feita com base em alguma métrica específica, baseando-se por exemplo na métrica com erro mínimo ou com resultado de maior confiança. Por exemplo, pode-se escolher a distância euclidiana e adotar a classificação do ponto de maior proximidade, que é a proposta adotada neste projeto.

Logo, o comitê funciona do seguinte modo: para cada documento de teste no qual houve empate entre os classificadores, calcula-se a distância euclidiana dele com cada um dos documentos de um algoritmo que empatou e toma-se a média destes valores. Repete-se então esse procedimento com os demais algoritmos empatados. O sistema define, portanto,

como resposta o algoritmo correspondente a menor dentre as médias calculadas. Caso não haja empate, o comitê só reproduz a resposta obtida pelo modelo, assim como no caso de todas as respostas serem 'não', situação em que o sistema apontará como documento criptografado com algum outro algoritmo diferente de DES, ElGamal e RSA.

3.5 Funcionamento do Sistema de Classificação

Nesta seção está descrito o funcionamento do sistema em termos de sua interface com o usuário, ou seja, como o sistema é apresentado ao usuário.

Inicialmente, o sistema solicita dois parâmetros de entrada ao usuário. O primeiro é o caminho da pasta com a base de criptogramas para treino e teste, que devem estar juntos. Aqui os documentos já estão criptografados e recomenda-se que todos tenham o mesmo tamanho, em torno de 500 KB, e que haja mesma quantidade de documentos com cada algoritmo. Além disso, os documentos de treino devem estar rotulados com um nome padrão indicado pelo sistema, como *DES_doc_01.txt*, enquanto os de teste devem estar nomeados com por exemplo *test_doc_01.txt*. O segundo parâmetro é o tamanho da palavra a ser usado na geração do espaço de palavras.

Inseridos os dois parâmetros solicitados inicialmente, o sistema gera o espaço de palavras. Em seguida, ele solicita mais duas informações: o caminho de destino para guardar as matrizes geradas com os cálculos das medidas de similaridade e dissimilaridade, e quais as medidas de similaridade que serão utilizadas (o sistema apresentará um menu com as opções de medida e escolhe-se algumas dentre as apresentadas). O sistema salva as matrizes de similaridade em um caminho especificado para caso o usuário necessite para consultas futuras.

O sistema então gera o modelo de SVM para classificação em cada classificador e, após terminar, imprime na tela as predições de cada documento de teste e plota os gráficos correspondentes ao SVM.

Como forma de avaliar o modelo, o sistema possui um outro modo de funcionamento, em que o usuário pode inserir documentos de teste em que ele sabe o rótulo. Para isso, o sistema solicita um vetor que indique os reais algoritmos dos documentos de teste e, a partir dele, compara com o resultado predito e gera as matrizes de confusão que avaliam o modelo, além de calcular a acurácia dos classificadores de cada algoritmo e do sistema como um todo.

4 IMPLEMENTAÇÃO DO SISTEMA

Neste capítulo será comentada a implementação do sistema, incluindo algumas dificuldades e observações encontradas, bem como a descrição dos experimentos realizados.

4.1 Caracterização das Bases de Criptogramas

Inicialmente, escolhe-se uma base de dados, composta por diversos textos, os quais são posteriormente divididos em dois grupos: base de treino e base de teste. A base de dados escolhida foi o conjunto de dados Reuters-21578, composto por 21.578 artigos de notícias em inglês, totalizando cerca de 30MB de texto em claro (16). Escolheu-se esse conjunto devido ao seu amplo uso no meio científico, pois os dados são de fácil acesso, bem como pela quantidade de dados disponíveis. Mas é importante ressaltar que o sistema desenvolvido será independente da base, podendo este funcionar com outras bases. A escolha de uma base é apenas para viabilizar os testes das etapas e análise dos resultados do projeto.

Os documentos são unificados em um grande texto conforme a Figura 14. O grande texto é então submetido a cada um dos algoritmos criptográficos, gerando três criptogramas, os quais são particionados em documentos menores, todos com mesmo tamanho. A unificação inicial é justamente para maximizar o número de criptogramas, já que, ao se dividir cada documento da base original separadamente, poderia ocorrer de, em algum dos documentos, sobrar uma parte de texto menor que o tamanho utilizado para os criptogramas.

Estes documentos criptografados compõem a base de criptogramas. Dois parâmetros importantes são o tamanho destes documentos criptografados e a quantidade de documentos em cada base. É dada a liberdade ao usuário de escolher o tamanho de cada documento, bem como quantos documentos ele desejará utilizar em cada base (base de treino e base de teste), ressaltando que o recomendado é haver quantidades iguais de cada algoritmo nas bases, com o intuito de manter a avaliação balanceada.

Os textos foram criptografados no modo ECB e, portanto, é necessário definir o tamanho de bloco utilizado para cifrar os textos. Idealmente, deve-se utilizar os tamanhos usuais, visto que, no momento da classificação, não se conhece a opção adotada por quem criptografou a mensagem, sendo o tamanho usual o mais provável de estar sendo utilizado. Os tamanhos usuais são 64 bits para o DES, como exige o próprio algoritmo, e 86 bytes para o RSA. Já para ElGamal, não há um tamanho usual bem definido na literatura. Optou-se então por utilizar 64 bits.

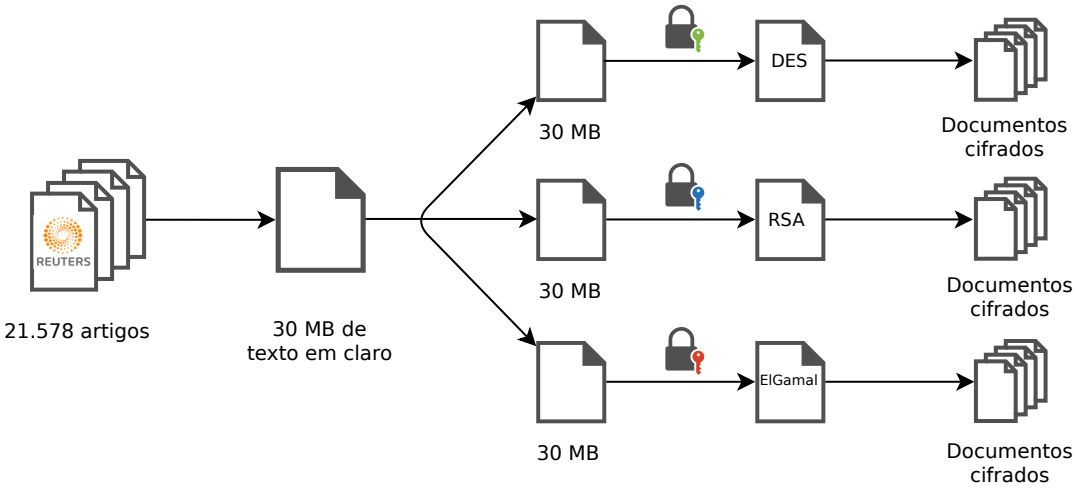

Figura 14 – Construção das bases de criptogramas.

A respeito de uma especificação do Algoritmo ElGamal, nota-se, pela fundamentação teórica, que ele gera como saída dois números. Considerou-se que os criptogramas são compostos apenas pelos valores de C_2, relacionados à mensagem M, conforme descrito na seção 2.1.2.2. Em termos práticos, a inclusão do C_1, que independe da mensagem, acrescentaria uma repetição muito grande ao criptograma, facilitando a análise do documento por alguém que o interceptou. Dessa forma, é mais recomendado que o transmissor e o receptor troquem o valor de C_1 por algum canal seguro.

Quanto à escolha das chaves, optou-se por gerar todos os criptogramas de um mesmo algoritmo com uma mesma chave, ou seja, escolheu-se uma única chave para cada algoritmo, totalizando três chaves. Essa decisão se deu principalmente pelo fato de que, ao utilizar chaves diferentes para um mesmo algoritmo, a aleatoriedade dos criptogramas é muito maior, tornando a análise bastante complexa, como mostram os resultados dos experimentos realizados por Ferreira(8).

Por fim, os tamanhos das chaves escolhidos foram de 64 bits para o DES, com o tamanho efetivo de 56 bits (8 bits são de paridade), que é uma exigência do próprio algoritmo; 1024 bits para o RSA, tamanho normalmente recomendado; e 256 bits para o ElGamal, principalmente devido ao tempo de processamento do ElGamal, que fica muito longo com chaves maiores.

4.2 Implementação das Bases de Criptogramas

A construção das bases iniciou-se a partir da unificação de todos os artigos presentes na Reuters-21578 em um único texto, que foi então dividido em três textos com tamanhos aproximadamente iguais, e cada um deles passou por um dos algoritmos analisados DES, RSA ou ElGamal, conforme já citado anteriormente.

Para a cifragem com DES e RSA foi utilizada a biblioteca *pycryptodome* de Python.

Já para o ElGamal, construiu-se um código próprio do algoritmo. Todos os algoritmos geram documentos em hexadecimal ao final do processo de encriptação.

Os textos foram criptografados no modo ECB e por isso cada texto foi lido de blocos em blocos, de acordo com os tamanhos informados no capítulo anterior (64 bits para DES, 86 bytes para RSA e 64 bits para ElGamal). Uma particularidade do DES na biblioteca *pycryptodome* é que ele só lê blocos de tamanho 64, de modo que foi preciso cortar alguns caracteres para que o texto a ser criptografado com DES tivesse tamanho múltiplo de 64. Como neste projeto não há preocupação com o conteúdo da mensagem, essa perda não gera nenhum problema. Caso a informação fosse relevante, uma opção mais recomendada seria o acréscimo de informação até alcançar um tamanho possível.

Por fim, os textos criptografados foram então particionados em documentos com o tamanho desejado pelo usuário, conforme explicado no capítulo anterior, e distribuídos em quantidades já pré-estabelecidas pelo usuário entre as bases de treino e teste, ficando todos armazenados numa mesma pasta.

4.3 Implementação do Espaço de Palavras e das Medidas

Após a construção das bases de criptogramas a partir dos artigos do conjunto Reuters-21578 e dos três algoritmos criptográficos escolhidos (DES, RSA e ElGamal), iniciou-se de fato a codificação de um sistema capaz de ler os documentos cifrados e representá-los em vetores de tamanho n, ou seja, vetores que representam a quantidade de ocorrências das n palavras distintas do domínio analisado. Essa funcionalidade foi implementada a partir do cálculo do histograma de palavras dos documentos.

O sistema, desenvolvido na linguagem Python, apresenta quatro opções disponíveis de tamanho de palavras para a análise dos vetores: 8, 16, 32 e 64 bits, embora não tenham sido analisados resultados com 32 e 64 bits, por motivos a serem detalhados na próxima subseção. Como os documentos cifrados foram armazenados em arquivos de texto com caracteres em hexadecimal, cada palavra pode representar 2, 4, 8 ou 16 caracteres de 4 bits, respectivamente.

Considerando os dois casos a serem analisados (8 e 16 bits), nos quais o espaço de todas as possíveis palavras correspondem a 2^8 e a 2^{16} sequências distintas de bits, foi possível implementar o histograma de palavras no segundo formato citado na seção 3.1 por meio da inicialização de um vetor com zero em todas as posições, e somando-se um à coordenada correspondente a uma palavra sempre que ela aparecia no texto.

Após obter a representação vetorial de cada documento cifrado, implementou-se as medidas de similaridade citadas na fundamentação teórica, a partir das quais foi possível obter as matrizes de similaridade e dissimilaridade para cada tamanho de palavra. As

funções correspondentes a cada medida foram desenvolvidas com o apoio da biblioteca de Python *numpy*, que possui um módulo de álgebra linear.

Como citado anteriormente, embora não tenham sido analisados ow resultados, foi possível implementar os espaços de palavras com 32 e 64 bits. Porém, ao longo da implementação, encontrou-se uma limitação de *hardware* ao se tentar realizar de modo análogo ao que foi feito com 8 e 16 bits. Ocorreu um problema de falta de memória, o qual exigiu uma abordagem diferente para o cálculo do histograma de palavras. Dessa forma, realizou-se a implementação com base em dicionários, tipo de estrutura de dados baseado em chave e valor, que são mais flexíveis pois apenas as palavras presentes nos documentos eram de fato armazenadas.

Com o intuito de visualizar as medidas de similaridade e dissimilaridade obtidas nos cálculos entre vetores dois a dois, construiu-se as matrizes triangulares superiores relacionando apenas alguns dos documentos da base de treino. Na Tabela 3, há um exemplo para o coeficiente Dice utilizando palavras de 8 bits. Após remover a diagonal da matriz, foi utilizado um mapa de calor em que tons verdes representam valores mais próximos de 1, com maior similaridade, e tons vermelhos representam valores mais próximos de 0, com menor similaridade.

É possível perceber que o algoritmo DES se distingue melhor dos outros, enquanto que a comparação entre ElGamal e RSA é mais sutil. Porém, em termos numéricos, todas as medidas obtidas são bem próximas com diferenças a partir da 3ª casa decimal, devido ao fato de, para esse caso, haver apenas 256 palavras distintas possíveis.

		DES							
		1	2	3	4	5	6	7	8
DES	1		0.9979	0.9980	0.9981	0.9983	0.9980	0.9984	0.9982
	2			0.9985	0.9985	0.9985	0.9984	0.9989	0.9987
	3				0.9990	0.9989	0.9984	0.9988	0.9985
	4					0.9991	0.9987	0.9988	0.9986
	5						0.9988	0.9990	0.9986
	6							0.9988	0.9986
	7								0.9990
	8								
ElGamal	1								
	2								
	3								
	4								
	5								
	6								
	7								
	8								
RSA	1								
	2								
	3								
	4								
	5								
	6								
	7								
	8								

		ElGamal							
		1	2	3	4	5	6	7	8
DES	1	0.9936	0.9935	0.9935	0.9939	0.9942	0.9938	0.9938	0.9935
	2	0.9956	0.9956	0.9954	0.9954	0.9958	0.9957	0.9958	0.9954
	3	0.9935	0.9934	0.9932	0.9936	0.9937	0.9934	0.9937	0.9932
	4	0.9941	0.9941	0.9941	0.9943	0.9943	0.9940	0.9942	0.9940
	5	0.9941	0.9941	0.9939	0.9942	0.9944	0.9941	0.9942	0.9940
	6	0.9938	0.9937	0.9935	0.9938	0.9940	0.9938	0.9939	0.9935
	7	0.9950	0.9948	0.9947	0.9949	0.9953	0.9950	0.9952	0.9946
	8	0.9945	0.9944	0.9944	0.9945	0.9948	0.9945	0.9947	0.9942
ElGamal	1		0.9990	0.9989	0.9989	0.9990	0.9990	0.9990	0.9990
	2			0.9989	0.9987	0.9991	0.9989	0.9990	0.9989
	3				0.9988	0.9989	0.9989	0.9989	0.9989
	4					0.9989	0.9989	0.9989	0.9988
	5						0.9991	0.9991	0.9990
	6							0.9991	0.9990
	7								0.9990
	8								
RSA	1								
	2								
	3								
	4								
	5								
	6								
	7								
	8								

		RSA							
		1	2	3	4	5	6	7	8
DES	1	0.9938	0.9945	0.9941	0.9941	0.9942	0.9940	0.9940	0.9941
	2	0.9957	0.9960	0.9958	0.9957	0.9960	0.9955	0.9955	0.9959
	3	0.9935	0.9940	0.9939	0.9937	0.9938	0.9935	0.9936	0.9938
	4	0.9942	0.9945	0.9945	0.9943	0.9943	0.9942	0.9943	0.9944
	5	0.9941	0.9946	0.9944	0.9944	0.9943	0.9941	0.9942	0.9944
	6	0.9936	0.9942	0.9940	0.9941	0.9940	0.9936	0.9939	0.9943
	7	0.9948	0.9954	0.9952	0.9951	0.9953	0.9949	0.9949	0.9952
	8	0.9947	0.9950	0.9949	0.9949	0.9948	0.9946	0.9947	0.9949
ElGamal	1	0.9986	0.9987	0.9987	0.9986	0.9988	0.9987	0.9986	0.9988
	2	0.9989	0.9989	0.9988	0.9987	0.9988	0.9987	0.9987	0.9989
	3	0.9985	0.9986	0.9986	0.9984	0.9987	0.9985	0.9986	0.9987
	4	0.9985	0.9987	0.9985	0.9985	0.9986	0.9985	0.9986	0.9987
	5	0.9989	0.9989	0.9988	0.9988	0.9990	0.9988	0.9988	0.9990
	6	0.9988	0.9988	0.9987	0.9986	0.9989	0.9986	0.9986	0.9990
	7	0.9988	0.9989	0.9988	0.9988	0.9989	0.9989	0.9988	0.9990
	8	0.9987	0.9986	0.9986	0.9985	0.9987	0.9986	0.9985	0.9987
RSA	1		0.9991	0.9989	0.9990	0.9992	0.9991	0.9989	0.9990
	2			0.9989	0.9989	0.9991	0.9990	0.9989	0.9990
	3				0.9990	0.9989	0.9989	0.9990	0.9989
	4					0.9990	0.9990	0.9990	0.9989
	5						0.9990	0.9991	0.9991
	6							0.9989	0.9989
	7								0.9990
	8								

Tabela 3 – Matriz parcial de similaridade do coeficiente Dice para palavras de 8 bits

Já na Tabela 4, os cossenos entre vetores para alguns dos documentos de cada algoritmo são exibidos considerando palavras de 16 bits. Nesse caso, percebe-se uma distinção maior entre os valores, apesar de a diferença entre RSA e ElGamal ainda ocorrer em pequenas casas decimais.

		DES								ElGamal								RSA							
		1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8	1	2	3	4	5	6	7	8
DES	1		0.820	0.836	0.841	0.851	0.838	0.843	0.838	0.336	0.334	0.335	0.337	0.337	0.335	0.338	0.335	0.340	0.338	0.336	0.338	0.342	0.340	0.340	0.339
	2			0.871	0.860	0.856	0.846	0.857	0.843	0.390	0.393	0.394	0.394	0.393	0.394	0.396	0.393	0.398	0.396	0.398	0.395	0.400	0.397	0.396	0.396
	3				0.910	0.899	0.872	0.877	0.868	0.327	0.329	0.331	0.331	0.329	0.330	0.332	0.327	0.334	0.331	0.332	0.330	0.335	0.331	0.332	0.332
	4					0.910	0.874	0.873	0.870	0.332	0.334	0.333	0.336	0.333	0.334	0.333	0.331	0.340	0.336	0.335	0.335	0.340	0.339	0.336	0.336
	5						0.888	0.892	0.864	0.334	0.337	0.335	0.338	0.338	0.337	0.338	0.334	0.342	0.339	0.338	0.337	0.342	0.338	0.337	0.338
	6							0.894	0.866	0.340	0.339	0.338	0.343	0.343	0.340	0.343	0.339	0.347	0.344	0.343	0.342	0.346	0.344	0.343	0.344
	7								0.882	0.362	0.361	0.360	0.362	0.363	0.362	0.364	0.358	0.367	0.365	0.364	0.362	0.367	0.366	0.364	0.363
	8									0.344	0.345	0.344	0.347	0.345	0.344	0.346	0.342	0.349	0.348	0.346	0.348	0.352	0.350	0.348	0.346
ElGamal	1										0.656	0.663	0.660	0.657	0.655	0.657	0.662	0.646	0.646	0.646	0.646	0.648	0.647	0.647	0.647
	2											0.663	0.661	0.666	0.663	0.663	0.666	0.648	0.648	0.651	0.650	0.651	0.650	0.649	0.649
	3												0.661	0.664	0.658	0.666	0.667	0.649	0.648	0.650	0.650	0.651	0.651	0.649	0.649
	4													0.662	0.659	0.661	0.661	0.652	0.652	0.649	0.649	0.653	0.651	0.647	0.648
	5														0.662	0.660	0.661	0.650	0.650	0.648	0.650	0.651	0.649	0.648	0.648
	6															0.662	0.660	0.653	0.653	0.652	0.653	0.653	0.652	0.651	0.654
	7																0.661	0.653	0.655	0.652	0.651	0.653	0.654	0.652	0.652
	8																	0.650	0.648	0.647	0.648	0.650	0.648	0.646	0.646
RSA	1																		0.659	0.655	0.658	0.659	0.656	0.655	0.657
	2																			0.656	0.656	0.660	0.655	0.656	0.656
	3																				0.657	0.655	0.655	0.654	0.654
	4																					0.658	0.657	0.655	0.655
	5																						0.658	0.658	0.658
	6																							0.656	0.656
	7																								0.657
	8																								

Tabela 4 – Matriz parcial de similaridade do cosseno entre vetores para palavras de 16 bits

Assim, ao todo foram obtidas 12 matrizes, que relacionam os documentos da base de criptogramas, considerando as medidas de similaridade apresentadas na seção 2.3. Como são 6 medidas, foram geradas 6 matrizes para o caso de palavras de 8 bits e 6 matrizes para o caso de palavras de 16 bits, totalizando 12 matrizes.

4.3.1 Tempos de Execução dos Cálculos das Medidas

Para os espaços de palavras com 8 e 16 bits, o cálculo das medidas teve duração curta, o que facilitou bastante a análise. A Tabela 5 mostra os tempos de execução:

Medida	8 bits	16 bits
Ângulo Cosseno	6.170	352.16
Coeficiente Simple-Matching	4.692	314.27
Coeficiente Dice	7.430	353.62
Coeficiente Jaccard	6.426	354.34
Distância Euclidiana	6.017	333.47
Distância Manhattan	11.724	1710.01
Tempo Médio	7.076	569.647

Tabela 5 – Tempos de execução para os cálculos das medidas de similaridade ou dissimilaridade em segundos.

Já para os casos dos espaços de palavras com 32 e 64 bits, o tempo estava consideravelmente maior, levando horas e até dias, o que dificultou a análise. Dessa forma, por questões de viabilidade de projeto, optou-se por não analisar estes dois casos.

4.4 Implementação do Modelo de Aprendizado de Máquina

O modelo de Aprendizado de Máquina SVM (*support vector machine*) foi implementado com auxilio da biblioteca *sklearn* em Python. A classe utilizada é a SVC (*support vector classification*), que recebe como parâmetro principal o tipo de *kernel*, que é linear (hiperplano) nas análises deste trabalho. A escolha do *kernel* linear se deu com o intuito de averiguar se o modelo mais já poderia ser suficiente para gerar bons resultados, ficando a análise com outros tipos de kernels para trabalhos futuros.

As medidas entram como vetores de treinamento nos parâmetros pelo método *fit* da classe SVC. Estes vetores são as médias das medidas de similaridade ou dissimilaridade, calculadas como descrito na Seção 3.3.

Os gráficos dos experimentos e as matrizes de confusão geradas ao final de cada um deles também são construídos com base nesta biblioteca.

4.5 Descrição dos Experimentos

Conforme informado na seção 3.5, o sistema possui dois modos de funcionamento. Um deles é a nível de produção, para classificar de fato criptogramas, e o outro é para avaliar o sistema em si, com o intuito de analisar seu desempenho ao classificar os criptogramas. Dessa forma, foram realizados experimentos neste segundo modo.

Os experimentos analisados foram todos com espaços de palavras com tamanhos de 8 ou 16 bits. A base de criptogramas implementada para os experimentos ficou com 100 criptogramas de cada algoritmo, todos com aproximadamente 500 KB, totalizando 300 documentos. Escolheu-se aleatoriamente 60 deles, 20 de cada algoritmo, e retirou-se os rótulos dos algoritmos para que pudessem servir como entradas para o classificador, ficando 240 documentos de treino e 60 de teste. Porém, guardou-se os rótulos originais para avaliar o modelo quanto à sua acurácia ao final de cada experimento. A necessidade de retirar os rótulos é pelo fato de a entrada do sistema, devido a sua implementação, exigir que os documentos de teste tenham um nome em formato específico, conforme descrito na seção 3.5.

Inicialmente, gerou-se o modelo de SVM para cada uma das 6 medidas implementadas, ou seja, com apenas um parâmetro, gerando 6 experimentos. Em seguida, os modelos analisados passaram a ter um par de medidas de similaridade ou dissimilaridade como parâmetro. Como foram implementadas 6 medidas, ao final ficaram 15 experimentos com pares de medidas. Por último, realizou-se uma rodada extra de experimento com todas as medidas como parâmetros, totalizando 22 experimentos. Cada um desses 22 experimentos foi realizado tanto com 8 bits como com 16 bits.

Foram analisados os resultados de cada classificador separadamente, bem como

o resultado do sistema como um todo, após a passagem pelo comitê, que está descrito na seção 3.4, e, por fim, realizou-se um teste com o sistema de classificação com alguns dos experimentos, para analisar o impacto da quantidade de documentos de treino nos resultados, ou seja, começando com 240 documentos de treino, foi-se reduzindo a quantidade e analisando a acurácia do sistema em cada experimento.

5 RESULTADOS DOS EXPERIMENTOS

Nesta seção estão apresentados os resultados de cada experimento proposto. Alguns deles são descritos por meio de matrizes de confusão, as quais permitem analisar a quantidade de observações que foram preditas corretamente ou não após aplicar um determinado modelo de Aprendizado de Máquina. Padroniza-se a leitura da matriz de confusão de acordo com a Tabela 6.

Para avaliar a performance de cada classificador com suas variáveis de entrada, define-se a acurácia, que representa o percentual das observações que foram preditas na classe correta. Considerando a notação da Tabela 6, temos a seguinte fórmula:

$$\text{acurácia} = \frac{VN + VP}{VN + FN + FP + VP}$$

		Classe Predita	
		0	1
Classe Verdadeira	0	VN verdadeiro negativo	FP falso positivo
	1	FN falso negativo	VP verdadeiro positivo

Tabela 6 – Representação para uma matriz de confusão

5.1 Experimentos com palavras de 8 bits e uma Medida como Parâmetro

Os primeiros experimentos consistiram em 3 modelos SVM que possuem como variável de entrada a média dos valores de uma medida de similaridade ou dissimilaridade calculados em relação aos 80 documentos de treino para cada algoritmo criptográfico.

A Tabela 7 mostra as acurácias obtidas com cada modelo, bem como a acurácia após a passagem do comitê:

Nota-se a acurácia de 100% pelo classificador DES. Como o algoritmo DES se diferencia bastante dos outros dois, que são de chave pública, pode-se dizer que este desempenho é esperado.

Destaca-se também que, após a avaliação dos 60 casos de teste pelo comitê, obteve-se uma maior acurácia em relação a cada um dos 3 classificadores binários. Note, por exemplo, a matriz de confusão para o classificador RSA do modelo com o coeficiente

	Medidas Utilizadas						Acurácia			
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA	após o Comitê
8	X						100.0%	96.7%	98.3%	98.3%
8		X					100.0%	100.0%	88.3%	95.0%
8			X				100.0%	96.7%	98.3%	98.3%
8				X			100.0%	96.7%	98.3%	98.3%
8					X		100.0%	96.7%	100.0%	96.7%
8						X	100.0%	95.0%	100.0%	96.7%

Tabela 7 – Acurácias para os modelos SVM com uma medida de similaridade ou dissimilaridade com palavras de 8 bits

Simple-Matching na Tabela 8 em que obteve-se 3 falsos negativos e 4 falsos positivos: após o critério de desempate do comitê, foi possível reduzir os erros de predição para apenas os 3 falsos negativos, implicando no acerto de 57 documentos de 60 testes (acurácia de 95%). Isso mostra a eficiência da medida de distância euclidiana para avaliar a dissimilaridade entre documentos.

	Medidas Utilizadas						Matriz de Confusão		
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA
8	X						40 0 0 20	39 1 1 19	39 1 0 20
8		X					40 0 0 20	40 0 0 20	36 4 3 17
8			X				40 0 0 20	39 1 1 19	39 1 0 20
8				X			40 0 0 20	39 1 1 19	39 1 0 20
8					X		40 0 0 20	40 0 2 18	40 0 0 20
8						X	40 0 0 20	39 1 2 18	40 0 0 20

Tabela 8 – Matrizes de confusão para os modelos SVM com uma medida de similaridade/dissimilaridade com palavras de 8 bits

5.2 Experimentos com palavras de 8 bits e duas Medidas como Parâmetros

Repetiu-se os experimentos com os 3 modelos SVM mas desta vez com duas médias como variáveis de entrada, também calculadas com base nas medidas de similaridade ou dissimilaridade em relação aos 80 documentos de treino para cada algoritmo criptográfico.

A Tabela 9 mostra as acurácias obtidas com cada par de medidas pelos classificadores DES, ElGamal e RSA, e após a passagem pelo comitê:

Comparativamente com o experimento anterior, nota-se que o DES permaneceu facilmente identificável. Verifica-se também que o ElGamal permaneceu com muitos casos

	Medidas Utilizadas						Acurácia			
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA	após o Comitê
8	X	X					100.0%	100.0%	90.0%	95.0%
8	X		X				100.0%	96.7%	98.3%	98.3%
8	X			X			100.0%	96.7%	98.3%	98.3%
8	X				X		100.0%	96.7%	100.0%	96.7%
8	X					X	100.0%	95.0%	100.0%	95.0%
8		X	X				100.0%	100.0%	90.0%	95.0%
8		X		X			100.0%	100.0%	90.0%	95.0%
8		X			X		100.0%	100.0%	91.7%	96.7%
8		X				X	100.0%	100.0%	90.0%	95.0%
8			X	X			100.0%	96.7%	98.3%	98.3%
8			X		X		100.0%	96.7%	100.0%	96.7%
8			X			X	100.0%	95.0%	100.0%	95.0%
8				X	X		100.0%	96.7%	100.0%	96.7%
8				X		X	100.0%	95.0%	100.0%	95.0%
8					X	X	100.0%	95.0%	100.0%	95.0%

Tabela 9 – Acurácias para os modelos SVM com 2 medidas de similaridade ou dissimilaridade com palavras de 8 bits

sem acurácia total, porém ainda altos. Destaca-se também que em ambos os experimentos a medida Simple-Matching teve um desempenho consideravelmente inferior nos classificadores binários RSA. Na identificação do ElGamal, essa mesma medida obteve acurácia de 100% para todos os experimentos.

Nota-se também o sucesso do comitê ao conseguir desempatar corretamente alguns dos documentos utilizando a distância euclidiana. Além disso, um outro ponto notável é o sucesso das medidas de dissimilaridade (distâncias euclidiana e Manhattan) na identificação do RSA que, salvo dois experimentos (linhas 8 e 9), identificou os documentos criptografados com RSA em 100% dos casos.

As matrizes de confusão dos modelos de cada classificador deste experimento estão presentes na Tabela 10:

5.3 Experimentos com Palavras de 16 Bits e uma Medida como Parâmetro

Repetiu-se os experimentos com uma medida como parâmetro de entrada no modelo mas desta vez para palavras de 16 bits.

A Tabela 11 mostra as acurácias de cada um dos classificadores nestes experimentos com cada uma das medidas, bem como após a passagem pelo comitê:

Comparativamente ao mesmo experimento, mas com palavras de 8 bits, nota-se

	Medidas Utilizadas						Matriz de Confusão		
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente J accard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA
8	X	X					40 0 0 20	40 0 0 20	37 3 3 17
8	X		X				40 0 0 20	39 1 1 19	39 1 0 20
8	X			X			40 0 0 20	39 1 1 19	39 1 0 20
8	X				X		40 0 0 20	40 0 2 18	40 0 0 20
8	X					X	40 0 0 20	40 0 3 17	40 0 0 20
8		X	X				40 0 0 20	40 0 0 20	37 3 3 17
8		X		X			40 0 0 20	40 0 0 20	37 3 3 17
8		X			X		40 0 0 20	40 0 0 20	37 3 2 18
8		X				X	40 0 0 20	40 0 0 20	37 3 3 17
8			X	X			40 0 0 20	39 1 1 19	39 1 0 20
8			X		X		40 0 0 20	40 0 2 18	40 0 0 20
8			X			X	40 0 0 20	40 0 3 17	40 0 0 20
8				X	X		40 0 0 20	40 0 2 18	40 0 0 20
8				X		X	40 0 0 20	40 0 3 17	40 0 0 20
8					X	X	40 0 0 20	40 0 3 17	40 0 0 20

Tabela 10 – Matrizes de confusão para os modelos SVM com 2 medidas de similaridade ou dissimilaridade com palavras de 8 bits

	Medidas Utilizadas						Acurácia			
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente J accard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA	após o Comitê
16	X						100.0%	100.0%	95.0%	100.0%
16		X					100.0%	100.0%	66.7%	66.7%
16			X				100.0%	100.0%	95.0%	100.0%
16				X			100.0%	100.0%	95.0%	100.0%
16					X		100.0%	100.0%	95.0%	100.0%
16						X	100.0%	100.0%	93.3%	100.0%

Tabela 11 – Acurácias para os modelos SVM com uma medida de similaridade ou dissimilaridade com palavras de 16 bits

a melhoria na identificação dos documentos criptografados com ElGamal, nos cause a acurácia foi de 100%, com todas as medidas. Quanto aos documentos RSA, nota-se uma perda considerável na acurácia, com destaque negativo para a medida Simple-Matching, que teve acurácia de $66,7\%$, bem abaixo das acurácias obtidas nos demais experimentos. Nota-se também que novamente comitê foi bem sucedido nos desempates, com exceção do experimento com a medida Simple-Matching.

A Tabela 12 apresenta as matrizes de confusão dos modelos de cada classificador para este experimento:

	Medidas Utilizadas						Matriz de Confusão		
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA
16	X						40 0 0 20	40 0 0 20	37 3 0 20
16		X					40 0 0 20	40 0 0 20	40 0 20 0
16			X				40 0 0 20	40 0 0 20	37 3 0 20
16				X			40 0 0 20	40 0 0 20	37 3 0 20
16					X		40 0 0 20	40 0 0 20	37 3 0 20
16						X	40 0 0 20	40 0 0 20	36 4 0 20

Tabela 12 – Matrizes de confusão para os modelos SVM com uma medida de similaridade ou dissimilaridade com palavras de 16 bits

5.4 Experimentos com Palavras de 16 Bits e Duas Medidas como Parâmetros

A Tabela 13 apresenta as acurácias obtidas nos experimentos com duas medidas para o caso de palavras de 16 bits.

Assim como nos experimentos com palavras de 16 bits e uma medida como parâmetro, nota-se a facilidade para identificar tanto DES como ElGamal e a dificuldade para identificar RSA. Nota-se que novamente o coeficiente Simple-Matching foi o responsável pelas acurácias mais baixas no RSA.

Comparando com os experimentos com duas medidas para palavras de 8 bits, houve uma redução considerável nas acurácias do classificador RSA, porém, o classificador ElGamal identificou todos os documentos criptografados com ElGamal, mostrando uma melhora significativa.

Quanto ao comitê, novamente ele desempatou todos os documentos corretamente dos experimentos sem a medida Simple-Matching.

A Tabela 14 apresenta as matrizes de confusão dos modelos de cada classificador para este experimento.

5.5 Experimentos com Palavras de 8 e 16 Bits e Seis Medidas como Parâmetros

Estes dois últimos experimentos consistiram em utilizar todas as medidas implementadas como parâmetros para os três modelos.

A Tabela 15 mostra as acurácias de cada um dos três classificadores e após o comitê:

Percebe-se que o aumento da quantidade de medidas como parâmetros não influen-

	Medidas Utilizadas						Acurácia			
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA	após o Comitê
16	X	X					100.0%	100.0%	70.0%	90.0%
16	X		X				100.0%	100.0%	95.0%	100.0%
16	X			X			100.0%	100.0%	95.0%	100.0%
16	X				X		100.0%	100.0%	95.0%	100.0%
16	X					X	100.0%	100.0%	93.3%	100.0%
16		X	X				100.0%	100.0%	70.0%	90.0%
16		X		X			100.0%	100.0%	70.0%	90.0%
16		X			X		100.0%	100.0%	70.0%	90.0%
16		X				X	100.0%	100.0%	80.0%	95.0%
16			X	X			100.0%	100.0%	95.0%	100.0%
16			X		X		100.0%	100.0%	95.0%	100.0%
16			X			X	100.0%	100.0%	93.3%	100.0%
16				X	X		100.0%	100.0%	95.0%	100.0%
16				X		X	100.0%	100.0%	95.0%	100.0%
16					X	X	100.0%	100.0%	93.3%	100.0%

Tabela 13 – Acurácias para os modelos SVM com 2 medidas de similaridade ou dissimilaridade com palavras de 16 bits

	Medidas Utilizadas						Matriz de Confusão		
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA
16	X	X					40 0 0 20	40 0 0 20	28 12 6 14
16	X		X				40 0 0 20	40 0 0 20	37 3 0 20
16	X			X			40 0 0 20	40 0 0 20	37 3 0 20
16	X				X		40 0 0 20	40 0 0 20	37 3 0 20
16	X					X	40 0 0 20	40 0 0 20	36 4 0 20
16		X	X				40 0 0 20	40 0 0 20	28 12 6 14
16		X		X			40 0 0 20	40 0 0 20	28 12 6 14
16		X			X		40 0 0 20	40 0 0 20	28 12 6 14
16		X				X	40 0 0 20	40 0 0 20	31 9 3 17
16			X	X			40 0 0 20	40 0 0 20	37 3 0 20
16			X		X		40 0 0 20	40 0 0 20	37 3 0 20
16			X			X	40 0 0 20	40 0 0 20	36 4 0 20
16				X	X		40 0 0 20	40 0 0 20	37 3 0 20
16				X		X	40 0 0 20	40 0 0 20	37 3 0 20
16					X	X	40 0 0 20	40 0 0 20	36 4 0 20

Tabela 14 – Matrizes de confusão para os modelos SVM com 2 medidas de similaridade ou dissimilaridade com palavras de 16 bits

ciou na acurácia do classificador RSA. De fato, ao se fazer a acurácia média para os outros 21 experimentos com palavras de 8 bits para o RSA, obtém-se uma acurácia média de $96,65\%$ para o classificador RSA, que é bem próximo ao $96,7\%$ encontrado no experimento com palavras de 8 bits desta seção. De modo semelhante, ao se fazer a acurácia média nos

	Medidas Utilizadas						Acurácia			
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA	após o Comitê
8	X	X	X	X	X	X	100.0%	100.0%	96.7%	96.7%
16	X	X	X	X	X	X	100.0%	100.0%	88.3%	96.7%

Tabela 15 – Acurácias para os modelos SVM com 6 medidas de similaridade ou dissimilaridade com palavras de 8 e 16 bits

21 experimentos com palavras de 16 bits, obtém-se uma acurácia média de $87,85\%$ para o classificador RSA, que novamente é bem próximo aos $88,3\%$ obtidos no experimento para 16 bits desta seção.

A Tabela 16 mostra as matrizes de confusão dos modelos de cada classificador para estes dois últimos experimentos:

	Medidas Utilizadas						Matriz de Confusão		
Bits	Ângulo Cosseno	Simple Matching	Coeficiente Dice	Coeficiente Jaccard	Distância Euclidiana	Distância Manhattan	DES	ElGamal	RSA
8	X	X	X	X	X	X	40 0 0 20	40 0 0 20	40 0 2 18
16	X	X	X	X	X	X	40 0 0 20	40 0 0 20	35 5 2 18

Tabela 16 – Matrizes de confusão para os modelos SVM com 6 medidas de similaridade ou dissimilaridade com palavras de 8 e 16 bits

Com relação a todos os experimentos analisados, pode-se concluir que a utilização de palavras menores (8 bits) proporcionou uma ligeira melhora no classificador RSA, enquanto que no caso do classificador ElGamal, os resultados foram melhores com palavras maiores (16 bits). Para o classificador ElGamal com a medida de similaridade Simple Matching também foram obtidos bom resultados utilizando palavras de 8 bits. Assim, as configurações mais adequadas para um sistema em tempo real seriam utilizando palavras menores (8 bits) devido ao menor tempo no cálculo das medidas de similaridade e dissimilaridade conforme a Tabela 5.

Alguns fatores que podem ter influenciado nessa diferenciação são algumas escolhas para parâmetros de pré-processamento, ou seja, da própria construção da base de teste. Um exemplo de parâmetro que pode ter influenciado é o tamanho de bloco para criptografar no modo ECB. Em todo caso, em ambos os casos os resultados do DES e do comitê, por meio da utilização da distância euclidiana para desempate, conseguiram identificar todos os criptogramas.

5.6 Experimentos com Bases de Treino Menores

Uma das formas de avaliar o comportamento dos modelos SVM foi através da variação da quantidade de documentos da base de treino, de 10 em 10 documentos até o

máximo de 80 documentos (base de treino completa) para cada algoritmo criptográfico.

Com base na Figura 15, nota-se um aumento da acurácia do comitê para modelos treinados com mais documentos rotulados, o que era intuitivamente esperado por formar um conjunto de documentos mais variado de forma a ser mais provável haver semelhanças com um documento desconhecido.

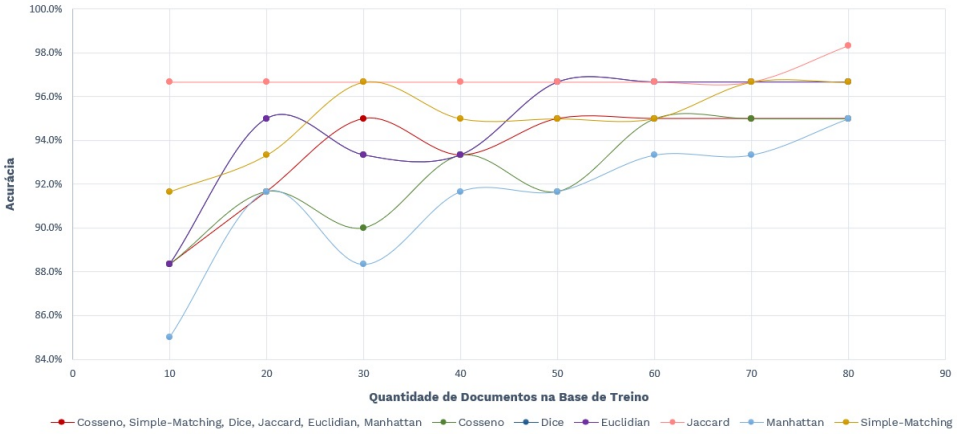

Figura 15 – Evolução da acurácia após o comitê para palavras de 8 bits com modelos de uma ou seis medidas de similaridade/dissimilaridade

Já para palavras de 16 bits, a quantidade de documentos da base de treino não apresentou impacto direto na acurácia dos modelos, conforme a Figura 16. Com exceção do coeficiente Simple-Matching, todas as outras medidas e a combinação das seis geraram resultados com acurácias entre $96,7\%$ e 100%.

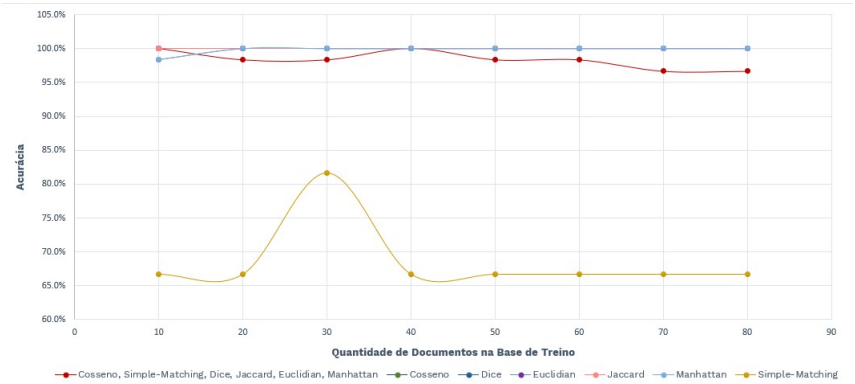

Figura 16 – Evolução da acurácia após o comitê para palavras de 16 bits com modelos de uma ou seis medidas de similaridade/dissimilaridade

5.7 Visualização dos Modelos de SVM

Com o intuito de ilustrar o modelo, as Figuras 17, 18, 19 e 20 apresentam os gráficos gerados pelo SVM com os pontos de treino e teste junto com o separador (kernel linear) aprendido durante o treinamento para alguns dos experimentos realizados. Dessa forma, é

possível visualizar os dados do problema trazendo maior interpretabilidade para o modelo de Aprendizado de Máquina, o que é possível após sintetizar as matrizes de similaridade e dissimilaridade em 2 variáveis de entrada a partir da utilização das médias já apresentadas.

Nota-se nos gráficos que alguns pontos possuem um certo alinhamento entre si enquanto outros ficam mais dispersos. De acordo com cada conjunto de medidas, foram obtidos resultados melhores para os gráficos com menor dispersão de pontos. É possível visualizar também quais pontos foram identificados de forma incorreta tanto durante a separação no treinamento quanto na classificação dos documentos de teste a partir das cores representadas.

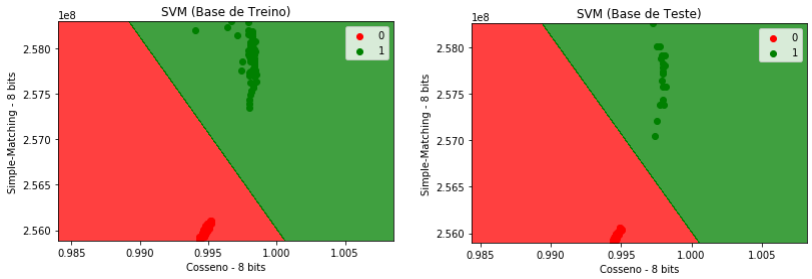

Figura 17 – Representação do plano SVM da Base de Treino e de Teste para o classificador DES com as medidas Cosseno e Simple Matching e palavras 8 bits

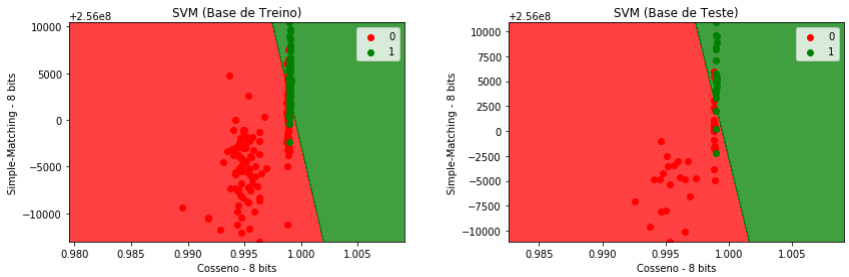

Figura 18 – Representação do plano SVM da Base de Treino e de Teste para o classificador RSA com as medidas Cosseno e Simple Matching e palavras 8 bits

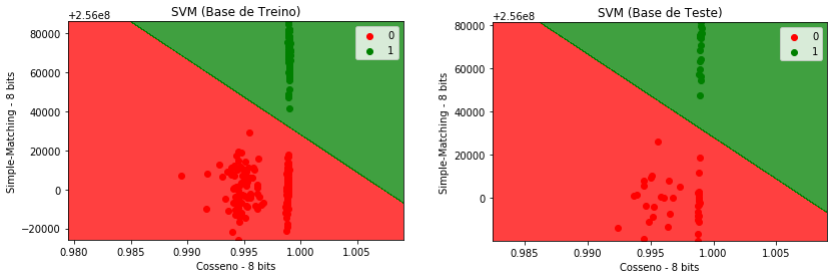

Figura 19 – Representação do plano SVM da Base de Treino e de Teste para o classificador ElGamal com as medidas Cosseno e Simple Matching e palavras 8 bits

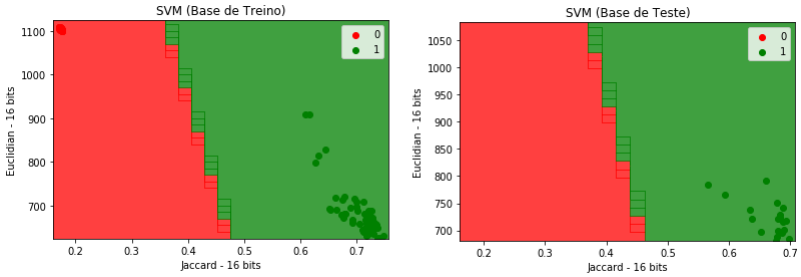

Figura 20 – Representação do plano SVM da Base de Treino e de Teste para o classificador DES com as medidas Coeficiente Jaccard e Distância Euclidiana e palavras 16 bits

6 CONCLUSÃO

O presente trabalho teve como objetivo desenvolver classificadores binários com base em técnicas de Aprendizado de Máquina amplamente utilizadas, as quais possuem inúmeras aplicações como por exemplo, na área de criptografia. Neste trabalho, desenvolveu-se um sistema de classificação com tais classificadores com o intuito de resolver o problema da identificação de criptogramas de acordo com o algoritmo utilizado para criptografá-lo.

A partir da utilização de diferentes medidas de similaridade e dissimilaridade, foi possível identificar a capacidade de classificação de criptogramas com base nas informações extraídas dos vetores de palavras para os documentos cifrados com os algoritmos criptográficos DES, RSA e ElGamal. O método apresentado permite sintetizar o conteúdo dos criptogramas em dados relevantes os quais puderam ser empregados no modelo de aprendizado de máquina SVM.

O sistema permitiu a realização de diversos experimentos, em cenários distintos, possibilitando não só a análise dos resultados das classificações em si, mas também um comparativo entre as medidas de similaridade. Os experimentos foram realizados em espaços com palavras de 8 ou 16 bits.

Em termos de resultados, os experimentos realizados mostraram que o sistema desenvolvido cumpriu com os objetivos. Apesar de isoladamente nem todos os três classificadores desenvolvidos tenham identificado os criptogramas corretamente, a implementação do comitê se mostrou bastante satisfatória, corrigindo na maioria dos experimentos as identificações erradas adotadas pelos classificadores. Nesse sentido, ressalta-se o bom funcionamento da medida de dissimilaridade de distância euclidiana para tal funcionalidade. Em todo caso, quanto aos resultados dos classificadores, o classificador DES acertou todos os testes, assim como o classificador ElGamal, porém apenas para palavras de 16 bits. Nota-se também que em geral não houve grande diferença entre as medidas de similaridade, salvo o Coeficiente Simple-Matching, que apresentou os piores resultados na identificação de documentos RSA.

Outro ponto a respeito dos resultados é que, embora as medidas à primeira vista parecerem estar bastante próximas ao se comparar os valores de RSA e ElGamal, já que ambos são algoritmos de chave pública, verifica-se que isso não se refletiu totalmente nos resultados, sendo consideravelmente mais simples de se identificar documentos ElGamal.

Dessa forma, com base nos resultados obtidos, conclui-se que, nas condições apresentadas para este projeto, o sistema cumpriu o objetivo previsto, ou seja, conseguiu classificar de modo eficiente criptogramas em uma base de documentos criptografados com DES, RSA ou ElGamal.

Como principal contribuição deste trabalho, há um sistema de classificação de criptogramas que possibilita diferentes valores de entrada para alguns parâmetros importantes para o sistema, como o tamanho de uma palavra, permitindo entradas de 8, 16, 32 ou 64 bits, e o número de medidas a se utilizar dentre as seis medidas implementadas. Como contribuições secundárias, há uma base de criptogramas criptografados com DES, ElGamal ou RSA, com diversas matrizes de similaridades calculadas sobre ela e um sistema de encriptação DES, ElGamal e RSA, que permite gerar criptogramas de mesmo tamanho, bem como na quantidade desejada, salvo por limitações da própria base de entrada. Além disso, há também uma análise do impacto de algumas medidas nas classificações, bem como a análise impacto do número de documentos de treino sobre a acurácia da classificação.

Como trabalhos futuros, sugere-se inicialmente a realização de experimentos com palavras de 32 e 64 bits, que não foram realizados nesse projeto. Sugere-se também outras formas de avaliação dos modelos desenvolvidos, a exemplo de Ferreira(8), que reduziu os tamanhos dos criptogramas, os quais neste projeto foram sempre fixos (500 KB). Quanto ao modelo de Aprendizado de Máquina, sugere-se também a análise do SVM com outros tipos de *kernels*. Além disso, sugere-se também a análise de outros experimentos, possivelmente com novas medidas de similaridade ou dissimilaridade, ou outros algoritmos criptográficos.

REFERÊNCIAS

1 CARVALHO, C. A. B. de. *O uso de Técnicas de Recuperação de Informação em Criptoanálise*. Rio de Janeiro: [s.n.], 2006. 79 p. (Mestrado em Sistemas e Computação). 16 maio de 2020. Disponível em: <http://www.comp.ime.eb.br/pos/arquivos/publicacoes/dissertacoes/2006/2006-CarlosCarvalho.pdf>.

2 MENEZES, A.; OORSCHOT, P. van; VANSTONE, S. *Handbook of Applied Cryptography*. 5. ed. Boca Raton: CRC Press, 2001. 816 p.

3 SOUZA, W. A. R. de. *Identificação de Padrões em Criptogramas Usando Técnicas de Classificação de Textos*. Rio de Janeiro: [s.n.], 2007. 252 p. (Mestrado em Sistemas e Computação). 16 maio de 2020. Disponível em: <http://www.comp.ime.eb.br/pos/arquivos/publicacoes/dissertacoes/2007/2007-William.pdf>.

4 JAMES, G.; WITTEN, D.; HASTIE, T.; TIBSHIRANI, R. *An introduction to statistical learning*. [S.l.]: Springer, 2013. v. 112.

5 SAMUEL, A. L. Some studies in machine learning using the game of checkers. *IBM Journal*, v. 3, n. 3, p. 211–229, 1959.

6 STALLINGS, W. *Criptografia e segurança de redes: princípios e práticas*. 6. ed. São Paulo: Pearson, 2015. 578 p.

7 MELLO, F. L. de; XEXéO, J. A. M. Cryptographic algorithm identification using machine learning and massive processing. *IEEE Latin America Transactions*, v. 14, n. 11, p. 4585–4590, 2016.

8 FERREIRA, L. de M. *A Aplicação de Wavelets no Reconhecimento de Padrões Criptográficos*. Rio de Janeiro: [s.n.], 2017. 138 p. (Mestrado em Sistemas e Computação). 17 maio de 2020. Disponível em: <http://www.comp.ime.eb.br/pos/arquivos/publicacoes/dissertacoes/2017/2017-LeandroFerreira.pdf>.

9 National Institute of Standards and Technology. *Data Encryption Standard (DES)*. 1999. FIPS Publication 46-3.

10 MARTINEZ, F. B.; MOREIRA, C. G.; SALDANHA, N.; TENGAN, E. *Teoria dos números: um passeio com primos e outros números familiares pelo mundo inteiro*. 2. ed. Rio de Janeiro: IMPA, 2013. 481 p.

11 COUTINHO, S. C. *Números Inteiros e Criptografia RSA*. 2. ed. Rio de Janeiro: IMPA, 2011. 226 p.

12 BISHOP, C. M. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 0387310738.

13 TEKNOMO, K. *Similarity Measurement*. 2006. 12 ago. de 2020. Disponível em: <https://people.revoledu.com/kardi/tutorial/Similarity>.

14 MANNING, C. D.; SCHüTZE, H. *Foundations of Statistical Natural Language Processing*. Cambridge, MA, USA: MIT Press, 1999. ISBN 0262133601.

15 SHANNON, C. E. A mathematical theory of communication. *Bell Syst. Tech. J.*, v. 27, n. 3, p. 379–423, 1948. Disponível em: <http://dblp.uni-trier.de/db/journals/bstj/bstj27.html#Shannon48>.

16 LEWIS, D. et al. Reuters-21578. *Test Collections*, v. 1, 1987. 19 mai. de 2020. Disponível em: <http://www.daviddlewis.com/resources/testcollections/reuters21578/>.