Universidade Federal de Pernambuco

Centro de Informática

YAGO ZACARIAS GOMES COUTINHO RIBEIRO

VERIFICAÇÃO DE ASSINATURAS MANUSCRITAS BASEADA EM PARÂMETROS DEPENDENTES DO ESCRITOR

Recife

2016

YAGO ZACARIAS GOMES COUTINHO RIBEIRO

VERIFICAÇÃO DE ASSINATURAS MANUSCRITAS BASEADA EM PARÂMETROS DEPENDENTES DO ESCRITOR

Trabalho de Graduação apresentado no Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para a obtenção do grau do Bacharel em Engenharia da Computação.

ORIENTADOR:CLEBER ZANCHETTIN

Recife

2016

AGRADECIMENTOS

Agradeço a todos que contribuíram no decorrer dessa jornada, especialmente:

Ao meu pai Eliomar Ribeiro e à minha mãe Josana Ribeiro pelos seus esforços e pelo apoio.

Ao meu irmão Diego que sempre esteve ao meu lado.

À minha namorada Janielli que sempre me incentivou e entendeu as minhas ausências.

Ao meu orientador Cleber Zanchettin que sempre esteve disponível.

Aos meus amigos de turma pelos momentos de alegria, aprendizado e companheirismo.

RESUMO

Com o avanço da tecnologia, vários métodos de identificação baseados em biometria foram criados como a impressão digital, as veias da mão e o reconhecimento da face. Porém, um dos métodos mais antigos, a assinatura manuscrita, ainda é um dos mais utilizados, seja por corporações ou por indivíduos. Esse fato se dá por ser um processo simples, de baixo custo e com validade jurídica. Entretanto, essa simplicidade, potencializa o maior problema dos métodos de identificação, a fraude. Esse trabalho tem como objetivo avaliar um sistema de verificação de assinaturas utilizando técnicas de processamento de imagens e de aprendizagem de máquina. Para a extração de características das imagens, foi utilizada a transformada *Curvelet*. Para validar as características das imagens, foi utilizado o método *One-Class Support Vector Machine* que é treinada apenas com assinaturas genuínas. O banco de dados GPDS (Grupo de Processamento Digital de Sinais) foi utilizado para avaliar a eficiência do método.

Palavras-Chaves: Curvelet, One-Class Support Vector Machine, aprendizagem de máquina, biometria, extração de características.

LISTA DE FIGURAS

LISTA DE TABELAS

Sumário

1. Introdução

Cada ser humano possui características singulares, físicas ou comportamentais, que permitem diferenciá-los um dos outros. As técnicas de identificação baseadas em biometria tentam utilizar essas peculiaridades na identificação pessoal. Esses métodos podem ser classificados em duas categorias: fisiológicos e os comportamentais. Os métodos fisiológicos são baseados em características corporais inatas como íris, impressão digital, palma da mão e reconhecimento facial. Já os comportamentais são analisados a partir de ações realizadas pelos indivíduos, como a forma de falar e a assinatura manuscrita.

A ciência que estuda esses métodos é a biometria, o que faz com que os métodos de identificação sejam chamados de métodos biométricos. Antes da existência da tecnologia biométrica automatizada, possível com o advento da computação, vários outros métodos não automatizados de biometria foram usados. Todavia, a computação permitiu o desenvolvimento de vários métodos precisos como reconhecimento da íris, reconhecimento da retina, reconhecimento da impressão digital [2] e reconhecimento da palma da mão [29].

Assinatura manuscrita ocupa um lugar muito especial no amplo campo da biometria. Isso se dá principalmente pelo fato da assinatura manuscrita ter se estabelecido como o método mais usado para identificação pessoal [1].

Mesmo com o avanço da tecnologia e da criação de vários métodos de identificação, pela sua simplicidade, a assinatura manuscrita não perdeu o posto de um dos métodos de identificação mais usado. A assinatura é primordial para validação de documentos, em pagamentos de cheques e/ou cartões de crédito, em processos judiciais e em compromissos de negócios de todos os tipos. Além de ser uma das biometrias menos intrusivas, sua captura é simples e a mesma possui validade jurídica. Um documento com uma assinatura atesta o conhecimento e concordância do autor da assinatura com o conteúdo do mesmo.

A verificação da autenticidade da assinatura é, na maioria dos casos, e dependendo da qualidade do fraudador, uma atividade relativamente simples.

Contudo, o tempo necessário para reconhecer um número significativo de assinaturas é muito grande. Além disso, o especialista humano é sujeito a falhas que, em alguns casos críticos, são inadmissíveis. A dificuldade em analisar assinaturas se deve pelo fato de existirem três tipos de falsificação: simples, habilidosa e randômica. A falsificação simples é quando a assinatura falsa possui o mesmo contorno da assinatura verdadeira. A falsificação habilidosa é quando a assinatura falsa é muito similar a genuína. Por último, a falsificação randômica é quando a assinatura falsa pertence a outro individuo sem similaridade visual com a assinatura original. As Figuras 1a, 1b, 1c, e 1d exemplificam, respectivamente, uma assinatura genuína, uma falsificação randômica, uma falsificação simples e uma falsificação habilidosa. Todos esses fatos, em relação a verificação de assinaturas, evidenciam a necessidade de um sistema computacional que seja confiável e que seja capaz de processar várias assinaturas em um tempo aceitável.

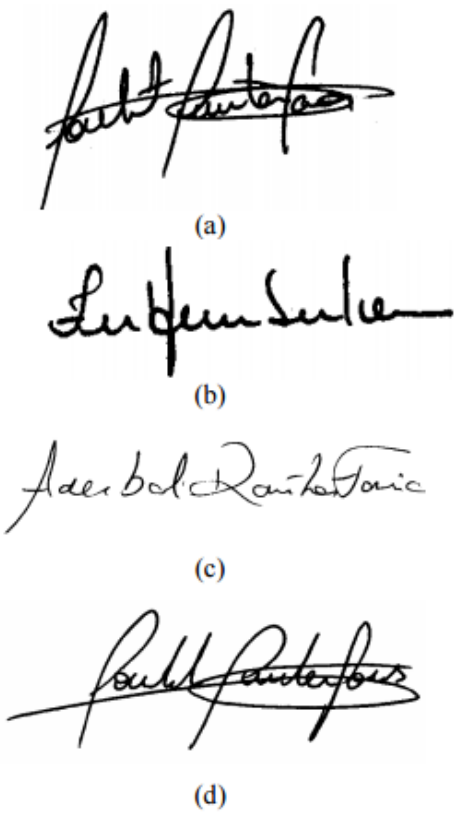


Figura 1 – Tipos de falsificação: a) Assinatura genuína; b) Falsificação randômica; c) Falsificação simples; d) Falsificação habilidosa [30]

Para atender a essa necessidade de um verificador de padrões para auxiliar na análise de assinaturas, ao longo do tempo foram desenvolvidos vários métodos na área de aprendizagem de máquina e inteligência artificial. Basicamente os sistemas que foram desenvolvidos para reconhecimento de assinaturas são divididos em dois grupos: *online* e *off-line*. As técnicas *online* são capazes de extrair dinamicamente, durante o processo de escrita, as características do assinante (trajetórias, pressão, velocidade, etc.). A aquisição das informações das assinaturas é possível através dos dispositivos de digitalização, como *tablet,* canetas especiais ou *touch pad*. O desenvolvimento de um sistema de verificação de assinaturas manuscritas baseado no modo *off-line* é mais difícil, comparado ao método *online,* pelo fato de que várias características desejáveis, como velocidade, pressão, entre outros, não estão disponíveis durante o processo de aquisição da imagem. O sistema de verificação depende apenas das características selecionadas das imagens das assinaturas [1].

Este trabalho de graduação será focado no desenvolvimento de sistemas de verificação de assinaturas *off-line*. Apesar de ser mais difícil e não tão preciso quanto o sistema *online*, a versão *off-line* é a mais difundida e presente nas operações comerciais por não haver a necessidade de hardware especifico e o processo poder ser realizado após a captura da assinatura sem o conhecimento do assinante. O desenvolvimento de um sistema *off-line* pode ser dividido em dois tipos: dependente do escritor ou assinante e independente do assinante.

O conjunto dependente do assinante é um grupo de sistemas que consiste em selecionar os parâmetros de cada escritor individualmente durante a construção do modelo. Esse modelo produz um sistema com menos erros, porém é muito custoso quando existem muitos assinantes, pois é necessário calcular os seus parâmetros para cada usuário inserido. Porém, permite evitar treinamentos com toda a base histórica de indivíduos quando os parâmetros precisam ser atualizados e permitem com que erros em amostras de alguns indivíduos não influenciem negativamente o sistema como um todo.

Já o grupo independente do assinante é justamente o oposto, possui uma taxa de acerto mais baixa quando dispõe de pouco dados na fase de treinamento, mas tem a vantagem de que quando um novo escritor é inserido,

não é necessário calcular os parâmetros do classificador. Para construir um sistema independente do usuário, inicialmente é utilizado um conjunto de assinantes na fase de treinamento, validação e testes. Quando um novo usuário é inserido ao sistema, os parâmetros dos classificadores encontrados, na fase de geração dos modelos, são atribuídos a esse novo usuário. Apesar dessa vantagem, a necessidade de assinantes como exemplo é um problema quando o sistema é implantado. Além disso, em alguns casos são necessários muitos exemplos de assinaturas falsificadas para treinar os modelos.

2. Motivação

A fraude em assinaturas manuscritas é um crime muito praticado no Brasil e gera prejuízos milionários a pessoas e instituições fraudadas. Em 2009, a empresa KPMG fez uma pesquisa com o intuito de investigar e avaliar o cenário de fraudes organizacionais de uma forma geral no país e mostrou que, na época, 68% das empresas entrevistadas sofreram fraudes, o de maior incidência detectado (29%) foi a falsificação de cheques e documentos, do qual faz parte a falsificação de assinaturas manuscritas [4].

A importância da assinatura para a validação de documentos e cheques bancários, aliados com a facilidade de fraude que esse método permite, foram os principais motivos para o desenvolvimento desse trabalho. Nele é proposto um avanço no método automatizado para verificação de assinaturas verdadeiras baseado no artigo de Y. Guerbai et al [1]. O principal mérito desse projeto é tentar simular o dia a dia dos usuários e das instituições, utilizando uma quantidade pequena de amostras, de quatro a dez, para treinar os classificadores de verificação de assinaturas manuscritas.

O número de amostras necessárias em qualquer abordagem relacionada ao reconhecimento de manuscritos que envolve treinamento de classificadores tornou-se um problema quando transposto para sistemas práticos, aplicações bancárias ou comerciais, que dispõem em geral de um número reduzido de amostras de cada classe, autor ou indivíduo. É inviável, então, solicitar diversas vezes aos usuários a produção de assinaturas para o sistema empregado realizar seu treinamento (manual ou automatizado) [3]. Desta forma, uma abordagem alternativa faz-se necessário, para um método ser aceito cientificamente e também para a sua utilização prática.

3. Objetivo

Este trabalho tem como objetivo desenvolver um verificador de assinaturas baseado no artigo de Guerbai et al. [1], tentando diminuir a sua taxa de erro na comparação entre assinaturas manuscritas. O artigo de Guerbai et al. propôs um verificador de padrões independente do escritor para diferenciar assinaturas fraudadas de assinaturas verdadeiras. Para isso, Guerbai et al. utilizaram a transformada Curvelet [6] para a extração das características da assinatura e o classificador *One-Class Support Vector Machine* (OC-SVM) [7] para construir a máquina de aprendizagem.

A proposta deste trabalho de graduação é modificar o método de extração de características e a forma de decisão do sistema (*threshold*), criando um sistema *off-line* e dependente do escritor, além de avaliar esta nova abordagem utilizando as mesmas métricas utilizadas por Guerbai et al [1].

4. Fundamentação teórica

Neste capítulo será apresentada a literatura e o background necessário para compreensão do sistema proposto. O capítulo foi organizado da seguinte forma: Métodos de avaliação, Trabalhos relacionados, *Support Vector Machines*, Pré-processamento das imagens e Transformada Curvelet.

4.1 Métodos de avaliação

Vários são os métodos de avaliar sistemas de classificação, na verificação de assinaturas manuscritas existe uma relação importante entre a quantidade de assinaturas falsas validadas e de assinaturas verdadeiras rejeitadas. Uma das métricas mais tradicionais na área envolve a análise da matriz de confusão gerada para o problema. Desta forma, na maioria dos trabalhos da literatura é avaliada a relação entre **FP** (False Positive): números de exemplos negativos classificados como positivo e **FN** (False Negative): número de exemplos positivos classificados como falso. Estas métricas são definidas na literatura como *False Acceptance Rate* (FAR) e *False Rejection Rate* (FRR). Além dessas, também foi utilizada a taxa *Average Error Rate* (AER), que permite verificar a relação ou ponto de equilíbrio entre estas duas taxas que normalmente são relacionadas. Essas taxas são definidas pelas fórmulas:

- **FRR** (*False Rejection Rate*): Números de exemplos positivos classificados como falso sobre o conjunto de assinaturas positivas.

$$FRR = \frac{FN}{nº\ de\ elementos\ positivos} \quad (1)$$

- **FAR** (*False Acceptance Rate*): Números de exemplos falsos classificados como positivo sobre o conjunto de assinaturas falsas.

$$FAR = \frac{FP}{nº\ de\ elementos\ falsos} \quad (2)$$

- **AER** (*Average Error Rate*): Média aritmética das taxas FRR e FAR.

$$AER = \frac{FAR+FRR}{2} \quad (3)$$

Essas métricas são muito utilizadas na literatura, como nos sistemas propostos por Kumar et al. [28], Batista et al. [11] e Guerbai et al. [1]. Para comparar os resultados do sistema proposto neste trabalho com a literatura, também foram utilizadas as mesmas métricas de avaliação.

4.2 Trabalhos relacionados

O trabalho desenvolvido por Srihari et al. [8] faz uma comparação dos métodos *off-line* dependente do escritor e independente do escritor. Para o método dependente, foram utilizados os classificadores *Distance Statistics* [31] e *Naive Bayes*. Já para o grupo independente do escritor, foram utilizados os classificadores *Distance Statistics*, *Naive Bayes* e *SVM*. Nos experimentos com os sistemas independentes, foram utilizadas dezesseis assinaturas na fase de treinamento, o artigo conseguiu uma taxa de FRR 21.3%, FAR 22.1% e AER 21.7% para o *Distance Statistics*, já o experimento utilizando o *Naive Bayes* alcançou um FRR de 22.9%, FAR 24.1% e AER 23.5%. Já para os métodos dependentes, a taxa de erro obtida ficou na média em relação ao que Guerbai et al e Batista et al. propuseram. Srihari et al conseguiram FRR de 17.6%, FAR de

20.7% e AER de 19.2% para o *Distance Statistics*. Para o *Naive Bayes* foi alcançado FRR de 9.95%, FAR de 13.0% e AER de 11.45%. A melhor taxa de erro ficou com a SVM que conseguiu FRR de 8.5%, FAR de 10.1% e AER de 9.3%. Como esperado, por causa das características de cada um, pode-se analisar que os métodos dependentes obtiveram um melhor resultado em relação os métodos independentes.

Ferrer et al. [10] propuseram um método de extração de características denominado características geométricas para verificação automática de assinaturas. Esse método foi testado em três classificadores diferentes: *Hidden Markov Models* (HMM), SVM e classificador *Euclidean Distance*. Para validá-lo, foram usadas doze assinaturas genuínas na fase de treinamento. Para o caso de teste, também foram utilizadas doze assinaturas genuínas, além de trinta assinaturas falsas. O conjunto das assinaturas falsas utilizado possui dois tipos falsificação: randômica e simples. Para as assinaturas randômicas, o classificador HMM obteve 2.2% de FRR e 3.3% FAR. Já a *Euclidean Distance* obteve 5.56% de FRR e 5.13% FAR. Por sua vez, a SVM com kernel RBF obteve 3.23% de FRR e 2.65% de FAR. Já nas assinaturas falsas do tipo simples, o HMM obteve 14.1% de FRR e 12.6% de FAR. O *Euclidean Distance* obteve 16.21% de FRR e 15.66% de FAR. Por último, a SVM alcançou 15.41% de FRR e 15.64% de FAR. Podemos verificar que para as assinaturas falsas do tipo randômica, o sistema obteve bons resultados, porém deixou a desejar nas assinaturas falsas do tipo simples.

Batista et al. [11] criaram uma seleção dinâmica dos modelos HMM e SVM, selecionando alguns escritores do banco dados GPDS de assinaturas manuscritas para construir um sistema de reconhecimento de padrões. O outros dos escritores do banco foram utilizados para testar o sistema. Eles testaram se a quantidade de assinaturas na fase de treinamento era diretamente proporcional ao resultados. Foram feitos três testes utilizando quatro, oito e doze assinaturas genuínas na fase de treinamento. Para o conjunto de quatro assinaturas, eles obtiveram 20.75% de FRR, 20.31% de FAR e 20.53% de AER. Para o conjunto de oito assinaturas, a taxa de FRR foi de 16.69, a de FAR foi de 17.38% e a de AER foi de 17.03%. Por sua vez, o conjunto com doze assinaturas genuínas obteve 16.81% de FRR, 16.88% de FAR e 16.84% de AER. Com esses

dados é possível verificar que, apesar da técnica utilizada ter sido a independente do escritor, a quantidade de assinaturas na fase de treinamento do modelo foi um fator importante para o resultado obtido.

Guerbai et al.[1] propuseram um verificador de assinaturas baseado na técnica *off-line* independente do assinante. Para isso eles utilizaram o método da transformada *Curvelet* na extração de características e *One-Class* SVM para construção do classificador. O banco GPDS utilizado no sistema possui trezentos assinantes com vinte e quatro assinaturas verdadeiras e trinta assinaturas falsas por assinante. Para construir o modelo, foram utilizados cento e sessenta escritores na fase de treinamento. Já para a fase de teste, foram utilizados os cento e quarenta restantes. Na fase de treinamento e de validação foram utilizadas apenas assinaturas genuínas. Foram utilizados três tamanhos diferentes na fase de treinamento: quatro, oito e doze assinaturas genuínas. As demais assinaturas genuínas foram utilizadas na fase de validação para encontrar o valor do *threshold* utilizado na fase de teste. O *threshold* encontrado na etapa de construção do modelo foi atribuído a todos os cento e quarenta assinantes utilizados na fase de teste. O sistema proposto conseguiu uma taxa de AER de 16.92%, 15.95% e 15.07% para os conjuntos de treinamento possuindo quatro, oito e doze assinaturas genuínas, respectivamente. Maiores detalhes do sistema serão fornecidos na Seção 5.

4.3 Pré-processamento das imagens

O principal método de pré-processamento de imagens que passam por um processo de classificação é a binarização, que permite separar a área de interesse da imagem do seu background. Uma técnica muito usada neste contexto é o algoritmo de *threshold* ou limiar iterativo desenvolvido por Ridler e Calvard [23]. Essa técnica foi utilizada por Guerbai et al. [1] e por isso ela também

foi utilizada neste projeto, com o objetivo de remover possíveis ruídos presentes nas imagens.

O algoritmo utiliza um *threshold* inicial, que geralmente é a média de todos os pixels da imagem. Com esse *threshold* inicial, o algoritmo transforma os pixels abaixo desse valor em background, já os pixels que possuem um valor maior, são classificados como objeto. Após isso, ele utiliza a seguinte fórmula para achar o novo valor do *threshold*:

$$Threshold = \frac{(média\ do\ background\ +\ média\ do\ objeto)}{2} \tag{4}$$

Esse processo é repetido até que não haja mudanças muito grandes no valor do *threshold*. Após isso, a imagem é binarizada seguindo a regra abaixo:

$$\begin{cases} g(x,y) = 1\ se\ f(x,y) \geq T \\ g(x,y) = 0\ caso\ contrário \end{cases} \tag{5}$$

onde x e y são as posições dos pixels, g representa a imagem binarizada, f representa a imagem original e T o *threshold*.

4.4 Transformada Curvelet

A Transformada Curvelet [6] foi desenvolvida para superar as limitações dos filtros Wavelet [37] e Gabor [38]. Embora a transformada Wavelet tenha sido amplamente explorada em vários ramos do processamento de imagem, ela falha em representar objetos contendo bordas e curvas orientadas aleatoriamente, uma vez que não é boa em representar singularidades de linha. Filtros Gabor foram desenvolvidos para obter desempenhos melhores do que a transformada Wavelet em representar texturas e recuperar imagens devido a sua abordagem

de orientação múltipla. No entanto, devido a perda de informação espectral dos filtros Gabor, eles não podem efetivamente representar imagens [17].

Devido a essas dificuldades de analisar imagens, Candes e Donoho desenvolveram a Transformada Curvelet [18]. A transformada possui um dicionário altamente redundante que pode fornecer a representação esparsa de sinais que possuem bordas ao longo da curva regular. A construção inicial da Curvelet foi redesenhada posteriormente e foi reintroduzida como *Fast Digital Curvelet Transform* (FDCT) [18]. Esta segunda geração da Curvelet foi desenvolvida para ser mais simples de entender e de usar. Também é mais rápida e menos redundante em comparação com a sua primeira versão [19].

A segunda geração da Curvelet foi feita utilizando duas implementações: *Unequally Spaced Fast Fourier Transform* (USFFT) [18] e *Wrapping Based Fast Curvelet Transform* (WBFCT) [18]. A segunda técnica é mais rápida e mais robusta que a técnica da primeira geração e a USFFT. Para esse projeto foi utilizado a Transformada Curvelet WBFCT.

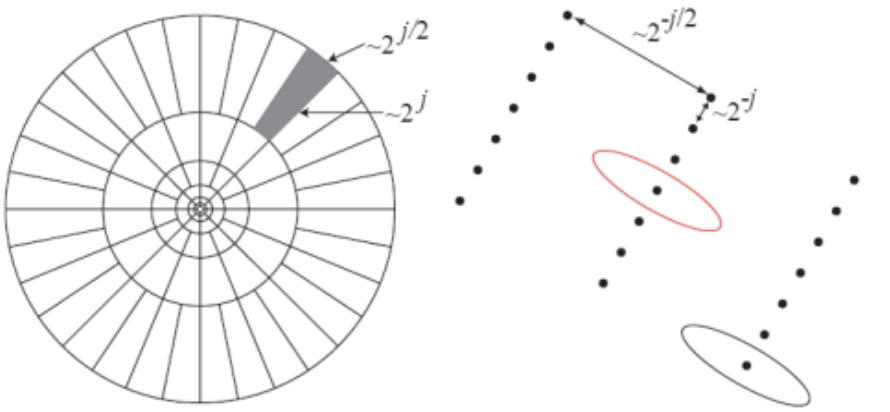


Figura 2 – Curvelet na frequência de Fourier (esquerda) e no domínio espacial (direita) [19]

Para implementar a transformada Curvelet, é tomada a primeira Transformada Rápida de Fourier (FFT) da imagem. Em seguida o plano de frequência Fourier 2D é dividido em fatias (como a região sombreada na Figura

2). A forma parabólica das fatias é o resultado da divisão do plano de Fourier em círculos radiais (círculos concêntricos) e divisões angulares. Os círculos concêntricos são responsáveis pela decomposição de uma imagem em escalas múltiplas (usadas para passar a imagem em escalas diferentes) e as divisões angulares particionam a imagem em diferentes ângulos ou orientações. Assim, se quisermos lidar com fatias particulares, precisamos definir sua escala j e ângulo l. Podemos, por exemplo, analisar o domínio espacial (Figura 2 a direita). Cada uma das fatias aqui corresponde a uma curva particular (mostrada como elipses) a uma dada escala e ângulo. Isto indica que a FFT inversa de uma fatia particular determinará os coeficientes da curva para essa escala e ângulo. Esta é a ideia principal por trás da implementação da transformada Curvelet [19].

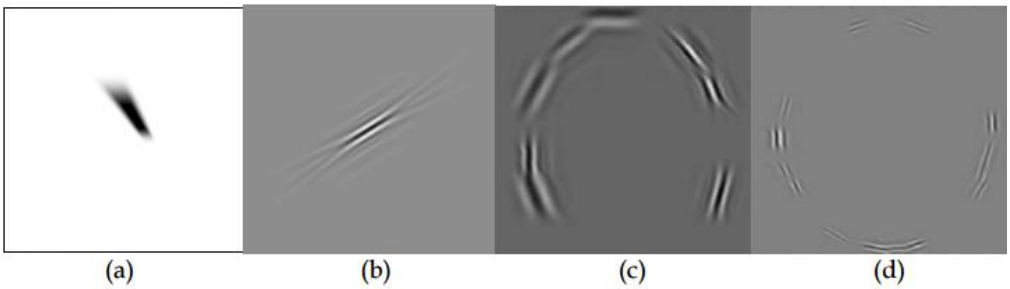


Figura 3 – (a) Curva no domínio da frequência, (b) Curvelet no domínio espacial, (c) Curvelets alinhadas a uma curva em uma escala particular, (d) Curvelet em escala mais fina [19]

Embora se mostre que as curvas constroem a forma de uma elipse, olhando para a Figura 3 (b-d), se pode verificar que na verdade ele se parece mais com agulhas alongadas. Isso decorre da lei de escalonamento parabólico ($Comprimento \approx largura^2$) a qual as curvas obedecem. Os valores dos coeficientes de curva são determinados pela forma como estão alinhados na imagem real. Quanto mais precisamente uma curva estiver alinhada com uma dada curva numa imagem, maior será o valor do coeficiente [19]. Como por exemplo na Figura 4, a Curvelet rotulada de ‘c’ está praticamente alinhada com a curva, fazendo com que o seu coeficiente seja alto. Já as Curvelets ‘a’ e ‘b’ terão um coeficiente com valores aproximadamente zero pois estão desalinhadas em relação a curva.

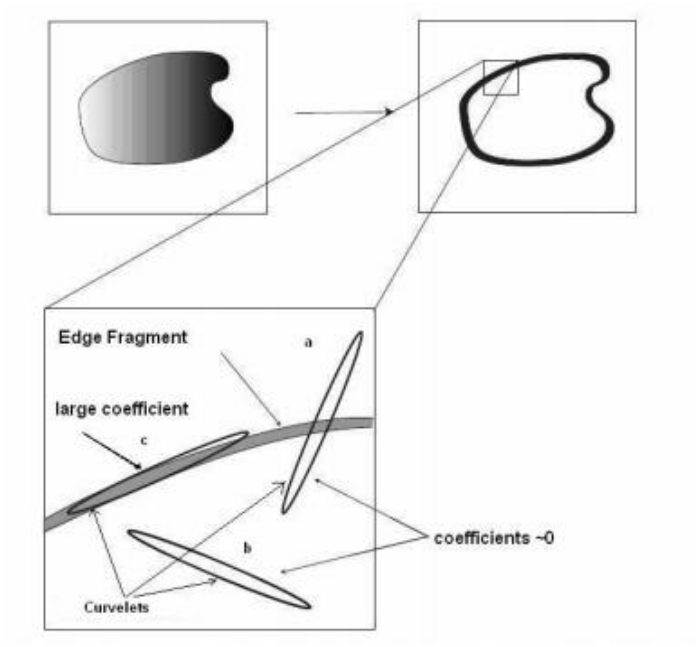


Figura 4 – Alinhamento das Curvelets em relação a curva [19]

Para calcular o valor dos coeficientes da Figura 4, a Transformada Curvelet recebe como entrada imagens em 2-D na forma Cartesiana $f[m,n]$ onde $0 \leq m \leq M, 0 \leq n \leq N$ e gera uma quantidade de coeficientes indexados pela escala j, pela orientação l e pela posição (k_1, k_2) da matriz que representa a imagem. Os coeficientes são calculados pela fórmula:

$$C^D(j,l,k_1,k_2) = \sum_{\substack{0 \leq m \leq M \\ 0 \leq n \leq N}} f[m,n]\, \psi^D_{j,l,k_1,k_2}[m,n] \tag{6}$$

onde cada $\psi^D_{j,l,k_1,k_2}[m,n]$ é a forma de onda digital da curvelet.

4.5 Support Vector Machines

Support Vector Machines (SVM) são máquinas de aprendizagem supervisionadas que são usadas tanto para classificação quanto para regressão. A SVM é uma técnica que tem como objetivo a separação ótima de duas classes

distintas. Por exemplo, a classificação SVM por ser binária, realiza uma separação automática entre duas classes que possuem características distintas. A partir de uma entrada de dados de treinamento, que possui duas classes, uma SVM responde ao usuário a qual das duas classes o indivíduo selecionado pertence [15].

Nos últimos anos, a SVM vem recebendo crescente atenção da comunidade de Aprendizagem de Máquina. Os resultados dessa técnica são comparáveis e muitas vezes superiores a máquinas muito utilizadas, como a Rede Neural [12]. Na literatura, existem vários exemplos de aplicações da SVM com sucesso na área de análise de imagens [1, 11, 34] e Bioinformática [35, 36].

SVM destaca-se por pelo menos duas características: possui sólida fundamentação teórica e pode alcançar alto desempenho em aplicações práticas. A teoria de aprendizagem pode identificar precisamente os fatores que devem ser considerados para a aprendizagem ser bem-sucedida e construir modelos que são bastante complexos [14].

4.5.1 SVM linear

A SVM tenta prever corretamente a classe à qual determinado dado de entrada pertence baseando-se nos seus próprios dados e nos dados do mesmo domínio em que o aprendizado ocorreu [15]. Logo, diante de uma amostragem de duas classes distintas, a SVM obtém uma função, que a partir dos parâmetros fornecidos, gera um hiperplano separando as mesmas.

O espaço que separa N classes distintas é denominado de região de decisão ou hiperplano de separação. Essa região é de suma importância no desenvolvimento da máquina. Ela será responsável pela generalização da SVM, quanto maior essa região, maior será a generalização. Logo a função de decisão mais adequada é aquela em que a distância entre os conjuntos das amostras de treinamento é maximizada.

A Figura 5 abaixo mostra dois separadores distintos para os mesmos dados. Como podemos ver através desse exemplo, é possível gerar diversas regiões de decisões com os mesmos dados. Por isso deve-se analisar qual é a função que faz a melhor distinção entre duas classes. Nessa figura, podemos concluir que a segunda região de decisão obteve um melhor resultado em relação a primeira, pois a região da segunda é mais larga o que gera uma melhor separação dos dados, implicando numa máquina que possui um desempenho melhor na fase de teste do sistema.

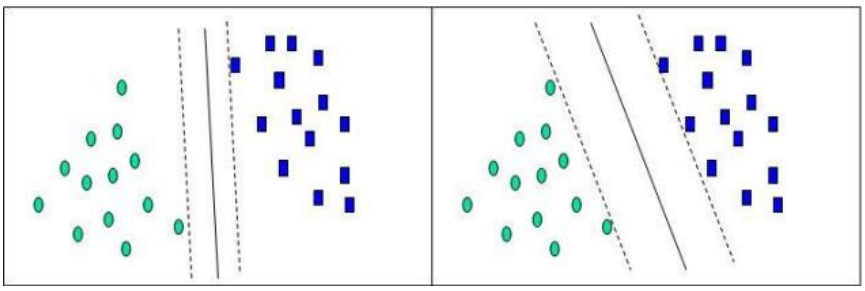


Figura 5 – Exemplos de duas regiões de decisão [13]

4.5.2 SVM não linear

O exemplo mostrado na figura acima é um exemplo muito simples e que, na maioria dos casos, não representa os dados coletados na vida real. Geralmente as amostras ficam distribuídas no espaço de forma não linear. Para resolver o problema da não linearidade dos dados, Hofmann et al. [16] propuseram uma projeção dos dados amostrais para um espaço de maior dimensão, através da função Kernel (Φ), com o intuito de encontrar uma dimensão em que seja possível traçar um hiperplano que consiga separar as classes de forma ótima. Após a aplicação da função Kernel, a técnica da SVM é aplicada nos dados a partir desse novo espaço. A Figura 6 mostra um exemplo da aplicação da função Kernel. Após a aplicação da função Kernel, na Figura 6,

os dados apresentaram o mesmo comportamento dos dados da Figura 5, sendo assim, possível traçar um hiperplano ótimo.

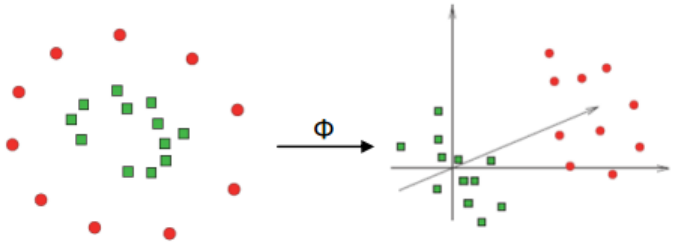


Figura 6 – Aplicação da função Kernel

Define-se a função Kernel como sendo uma função que recebe dois argumentos xi e xj, os dados de entrada, e calcula o produto escalar desses argumentos no espaço de características. A construção de Φ é feita implicitamente, pois se utiliza o Kernel sem conhecê-lo, tornando seu uso mais simplificado. As principais funções de Kernel que podem ser utilizadas no SVM são: linear, *Radial Basis Function* (RBF), polinomial e sigmoidal [13]. Nesse trabalho foi usada o Kernel RBF, o mesmo utilizado por Guerbai et al.

A função RBF possui a seguinte fórmula:

$$\Phi = \exp\Big(-\gamma * d\big(x_i * x_j\big)\Big) \tag{7}$$

onde γ é um parâmetro do Kernel RBF e d é a função de distância. A RBF tem a função euclidiana como função de distância padrão, entretanto muitas outras funções de distância podem ser utilizadas. Para esse projeto de graduação foi utilizado o Kernel RBF com cinco técnicas de distâncias distintas: Euclidiana, Correlação, Chebychev, *Spearman* e *Cityblock*.

4.5.3 One Class SVM

A One Class Support Vector Machine (One-Class SVM), desenvolvida por Schölkopf et al. [7], é uma SVM não linear que recebe, como conjunto de treinamento, apenas dados pertencentes a uma classe (genuína) e tem como objetivo verificar se os dados do conjunto de teste pertencem a essa classe. Para alcançar esse objetivo, a One-Class SVM cria uma região de fronteira que agrupa, de forma ótima, os dados do conjunto de treinamento.

Para criação do algoritmo, Schölkopf et al. consideraram que os dados próximos da origem pertencem a classe falsa, enquanto que os dados que não estão na origem são da classe genuína. A One-Class SVM pode ser vista como uma SVM binária, onde todos os dados da fase de treinamento pertencem a primeira classe (genuína), ao passo que a origem pertence a segunda classe (falsa).

Mais especificamente, a One-Class SVM tem como objetivo estimar uma função $f(x)$ que engloba a maioria das amostras da fase de treinamento dentro da região de fronteira. A função $f(x)$ é definida pela fórmula:

$$f(x) = sgn\left\{\sum_{i=1}^{m} \alpha_i K(x, x_i) - \rho\right\} \quad (8)$$

onde m é a quantidade de elementos da classe fornecida na fase de treinamento, ρ é a distância da região de fronteira em relação a origem e α_i são os multiplicadores de Lagrange computados pela otimização da seguinte fórmula:

$$min_{\alpha}\left\{\frac{1}{2}\ \alpha_i \alpha_j K(x_i, x_j)\right\} \quad (9)$$

respeitando as seguintes regras:

$$0 \leq \alpha_i \leq \frac{1}{vm} \quad (10)$$

$$\sum_{i}^{m} \alpha_i = 1 \quad (11)$$

onde v é a porcentagem de elementos considerados como *outliers* e K é a função Kernel [1]. A One-Class SVM irá aceitar um elemento quando a função $f(x) > 0$, caso contrário o elemento é rejeitado. A Figura 7 exemplifica a separação dos conjuntos, feita pela OC-SVM.

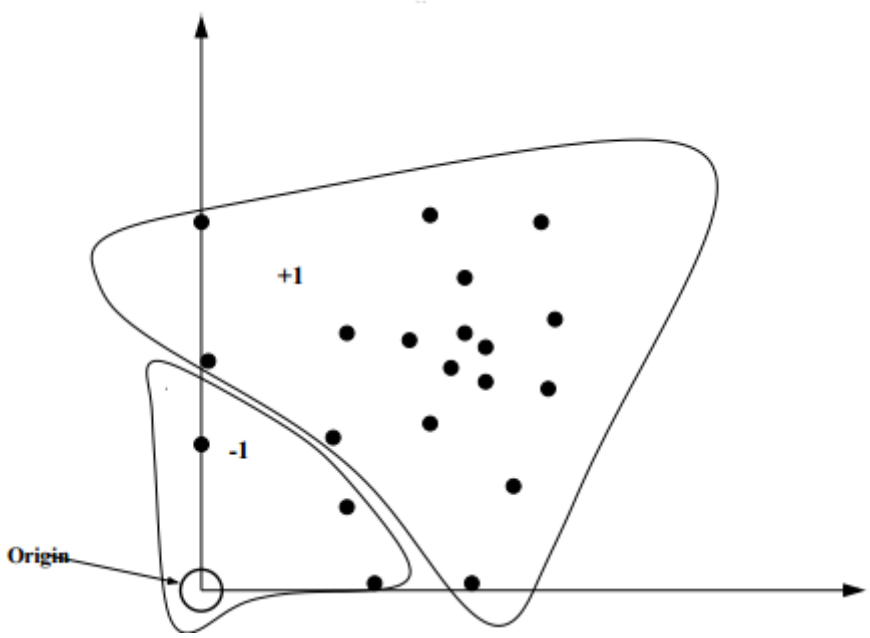


Figura 7 – Divisão dos conjuntos feita pela OC-SVM [40].

5. One-Class SVM para verificação de assinaturas baseada em parâmetros independente do escritor

Nesta seção será explanado o sistema desenvolvido por Guerbai et al. [1], o qual serviu de base para a desenvolvimento desse trabalho de graduação.

Guerbai et al. propuseram um sistema de verificação de assinatura manuscrita baseada na técnica *off-line* independente do escritor, seguindo os fluxogramas da Figura 8 e 9.

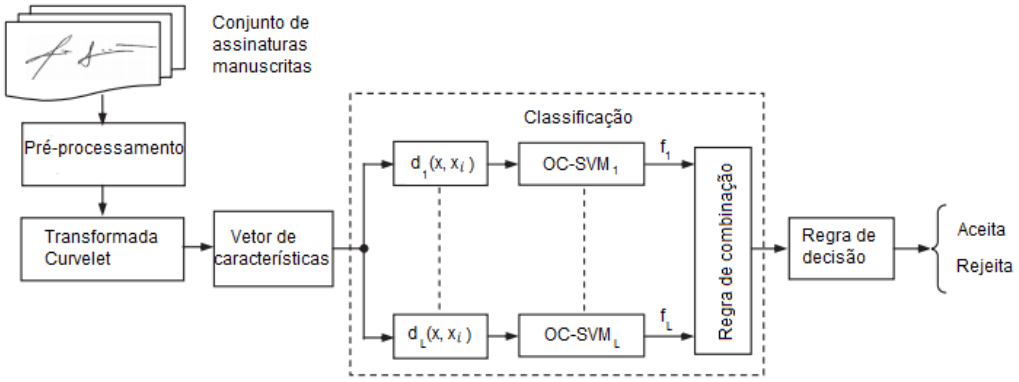


Figura 8 – Fluxograma do sistema proposto por Guerbai et al. [1] utilizado na fase de validação e teste.

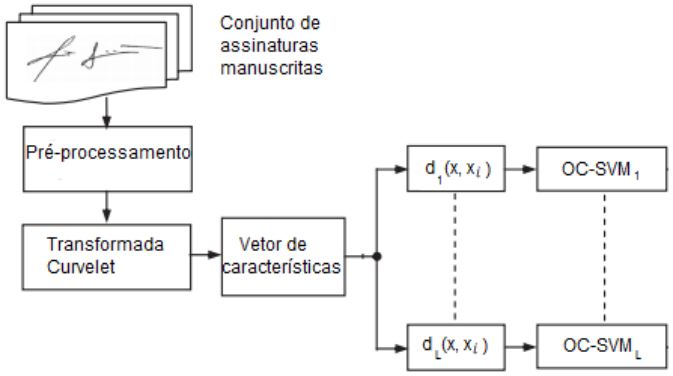


Figura 9 – Fluxograma do sistema proposto por Guerbai et al. [1] utilizado na fase de treinamento.

Nos tópicos seguintes será explanada cada etapa do fluxograma utilizado pelos autores.

5.1 Banco de dados

Para desenvolvimento do sistema, Guerbai et al. utilizaram dois bancos de dados, o primeiro foi fornecido pelo Grupo Procesado Digital de La Señal (GPDS) e o segundo pelo *Center of Excellence for Document Analysis and Recognition* (CEDAR). Esses dois bancos de dados foram utilizados com o intuito de avaliar o sistema proposto.

O GPDS, cuja a sede de pesquisa fica na Universidade de Las Palmas nas Ilhas Canárias, foi criado há dez anos por engenheiros de telecomunicação recém-formados, possui várias áreas de interesse sendo uma das quais a de sistemas biométricos. O banco fornecido por eles é composto por assinaturas reais e manuscritas de trezentas pessoas que são previamente separadas como genuínas e falsas. Para cada pessoa do banco, existem vinte e quatro assinaturas verdadeiras e trinta assinaturas falsas.

Já o CEDAR, situado na Universidade de Buffalo, que é uma Universidade Estadual dos Estados Unidos, é um centro de pesquisa fundado em 1978 no departamento de informática. Apesar de possuir pesquisas em várias áreas da

computação como reconhecimento de padrões, aprendizagem de máquinas, mineração de dados e recuperação de informações, o grupo é focado na área de análise e reconhecimento de documentos manuscritos. O banco fornecido pelo centro possui cinquenta e cinco escritores e também é composto por assinaturas reais e previamente separadas como genuínas e falsas. Para cada um dos escritores, existem vinte e quatro assinaturas genuínas e falsas.

5.2 Pré-processamento e geração de características

A etapa do pré-processamento das imagens foi utilizada para eliminar possíveis ruídos, utilizando o método de Ridler e Calvard [23], o qual consiste em iniciar um limiar para separar o background do objeto. Após isso é calculada uma média dos pixels de cada uma das duas classes para gerar um novo limiar. Esse método iterativo é executado até que o valor do limiar não sofra uma grande alteração e o último valor encontrado será usado para binarizar as imagens.

Após a etapa do pré-processamento, foi utilizada a Transformada Curvelet para geração dos coeficientes das imagens, a partir dos quais, o vetor de características de cada imagem foi criado, com o objetivo de gerar atributos únicos de cada imagem, para que o classificador consiga separar as classes das assinaturas genuínas e falsas. Para geração do vetor de características, foi utilizada a seguinte fórmula da energia:

$$E(l,r) = \sum_i \sum_j | C_{l,r}(i,j)| \quad (12)$$

onde l e r correspondem à escala e ao ângulo, respectivamente. No artigo não é informado os valores da escala e do ângulo utilizados, apenas que o vetor de características gerado, após o uso da energia, possui dez componentes.

5.3 Classificação e Regra de combinação

Após as etapas de processamento de imagem e extração de características, foram criadas as máquinas de aprendizagem. Na criação dos classificadores, foram utilizadas cinco One-Class SVM para cada escritor presente no banco de dados, cada uma possuindo o Kernel RBF com uma técnica de distância diferente. Os cinco métodos de distâncias utilizados são apresentados na Tabela 1.

Distance metric	$d(x, x_i)$
Euclidean	$\sqrt{\sum_{k=1}^{d}(x_k - x_{i,k})^2}$
Cityblock	$\sum_{k=1}^{d}\lvert x_k - x_{i,k}\rvert$
Chebychev	$\max_{k=1,\ldots,d}\lvert x_k - y_{ik}\rvert$
Correlation	$\frac{\mathrm{cov}(x,x_i)}{\sigma_x \sigma_{x_i}}$
Spearman	$\frac{1-6\sum_{i=1}^{d}(\mathrm{rank}(x)-\mathrm{rank}(x_i))^2}{n(n^2-1)}$

Tabela 1 – Distâncias utilizadas no Kernel da One-Class SVM por Guerbai et al. [1]

Após a geração dos resultados das cinco One-Class SVM de cada escritor, é necessário a utilização de um método que agregue esses resultados em apenas um. Existem várias regras de combinação que podem ser utilizadas, como a média, mínimo, máximo e majority votes [33]. No sistema proposto por Guerbai et al. foi utilizado o majority votes.

5.4 Regra de decisão

Por último, Guerbai et al. propuseram um método para encontrar o *threshold* utilizado na classificação das imagens. Esse método foi necessário devido à pequena quantidade de assinaturas disponibilizadas pelos bancos de dados para cada escritor. A SVM é uma técnica que possui uma boa generalização quando é fornecido uma grande quantidade de dados na fase de treinamento, entretanto para poucos dados, ela não obtém um bom desempenho [1]. Por isso, Guerbai et al. utilizaram a seguinte fórmula para encontrar o *threshold*:

$$t = m_f + k * \alpha_f \tag{13}$$

onde k é uma constante, m_f a média e α_f o desvio padrão, obtidos a partir dos valores das funções de decisão das cinco SVM, que cada escritores possui, durante a fase de validação, que será explana nos tópicos que seguem.

5.5 Desenvolvimento do sistema

Após a explanação de cada etapa do fluxograma, essa seção será destinada para detalhar as fases de treinamento, validação e teste do sistema desenvolvido por Guerbai et al. A Tabela 2, retirada do artigo, mostra a quantidade de escritores e a quantidade de assinaturas utilizadas, por escritor, nas fases de criação (treinamento e validação) e na de teste do modelo.

Banco	Etapa de construção			Etapa de avaliação		
	Escritores	N_p	N_t	Escritores	N_g	N_f
CEDAR	30	8	16	25	24	30
GPDS	160	8	16	140	24	30

Tabela 2 – Divisão dos dados para criação do modelo de Guerbai et al. [1]

Para a etapa de criação do modelo, foram utilizadas trinta escritores do banco de dados CEDAR e cento e sessenta escritores do banco de dados GPDS. Os escritores restantes de cada banco foram utilizados para avaliar o desempenho do sistema.

Para a fase de treinamento, com o intuito de simular o dia a dia das empresas, foi utilizada uma pequena quantidade de assinaturas genuínas, representadas na Tabela 2 por N_p. Essa fase do processo também foi utilizada para encontrar os parâmetros, porcentagem de *outliers* e o parâmetro γ utilizado no Kernel das SVM, que iriam gerar *One-Class* SVM com o máximo de *support vectors*.

Na fase de validação do sistema, foi utilizada a quantidade restante de assinaturas genuínas N_t para criação do conjunto de validação. Esse conjunto foi construído para cada um dos cento e sessenta escritores e foram separados em dois subconjuntos: assinaturas verdadeiras e assinaturas falsas do tipo randômica. O subconjunto de assinaturas verdadeiras de um escritor foi gerado com as suas N_t assinaturas genuínas e o subconjunto de assinaturas falsas foi construído com a soma das N_t assinaturas genuínas dos outros escritores.

A fase de validação foi utilizada para encontrar o valor do *threshold* utilizado na regra de decisão do sistema. A partir dos valores das funções de decisão, utilizando o conjunto de validação, foram encontrados os valores de m_f e α_f da equação (13). O valor do k, encontrados através das medidas de avaliação FAR e FRR, variou em um *range* de [-3, 3] com o objetivo de encontrar o valor que resultaria no $FAR = FRR$.

Na fase de teste, foram utilizados os escritores da etapa de avaliação da Tabela 2. Para cada escritor, assim como na fase de treinamento, foram criadas cinco One-Class SVM utilizando os parâmetros calculados na fase de treinamento do sistema. Já para a regra de decisão, foi utilizado o *threshold* calculado na fase de validação. A partir da construção desse novo modelo, o sistema foi testado utilizando as assinaturas genuínas e falsas de cada escritor presente na etapa de avaliação do sistema.

Com essas características apresentadas nas etapas de treinamento, validação e teste, o modelo criado foi caracterizado como um sistema *off-line* independente do escritor.

6. One-Class SVM para verificação de assinaturas baseada em parâmetros dependente do escritor

Nessa seção serão explanadas as diferenças entre o sistema proposto nesse trabalho e o sistema de Guerbai et al. Esse trabalho utilizou os mesmos fluxogramas da Figura 8 e 9, entretanto a etapa da extração de características e as fases de validação do sistema foram modificados para melhorar o desempenho do sistema proposto por Guerbai et al. Nos tópicos que seguem, será explicada cada fase do sistema e suas diferenças em relação ao artigo utilizado como modelo.

6.1 Extração de características

Antes de realizar a extração das características das imagens, foi utilizada a técnica de Ridler e Calvard para binarização e eliminar possíveis ruídos. Posteriormente, foi usada a transformada Curvelet para geração dos coeficientes das imagens. Baseado nos coeficientes encontrados, foram utilizadas duas técnicas para geração do vetor de características.

A primeira técnica utilizada foi a energia apresentada na fórmula (13), a mesma utilizada no artigo de Guerbai et al [1]. Mais adiante, no tópico de resultados, será demonstrado que a técnica de energia não gerou vetores de características que apresentassem bons resultados no treinamento da SVM.

A segunda técnica utilizada foi a do desvio padrão. A mesma já foi utilizada para a geração do vetor de características no artigo [24] em conjunto com a energia, gerando bons resultados. Com o desvio padrão, a SVM

conseguiu um bom nível de generalização, o que gerou uma taxa de erro pequena na fase de teste.

6.2 Construção do modelo

Após a aplicação das técnicas de pré-processamento e extração de características das assinaturas manuscritas, as imagens foram divididas em três conjuntos: treinamento, validação e teste. Os conjuntos de treinamento e validação foram construídos usando apenas as assinaturas genuínas, assim como no sistema de Guerbai et al. Para criação desses conjuntos, foi utilizado o banco de dados GPDS que será mostrado na Seção 7.

Os três passos (treinamento, validação e teste) da construção do reconhecedor de padrão serão explanados nos tópicos que seguem.

6.2.1 Treinamento

Foram usados para teste cinco conjuntos de treinamento com tamanhos diferentes. O primeiro possui quatro assinaturas, o segundo seis, o terceiro oito, o quarto dez e o quinto doze das vinte e quatro assinaturas genuínas que cada pessoa possui. Isso foi feito com o objetivo de testar se o tamanho do conjunto de treinamento é diretamente proporcional aos resultados obtidos pelo sistema.

O conjunto de treinamento não foi utilizado apenas para treinar as SVM, eles também foram usados para encontrar, experimentalmente, os parâmetros do classificador. Para isso foi utilizada a ferramenta Weka [39] que possui várias bibliotecas da área de Inteligência Artificial implementadas, entre elas a Libsvm [22], biblioteca usada no projeto para construção do classificador. A ferramenta Weka foi necessária para executar a validação cruzada nos dados de treinamento.

6.2.2 Validação

O conjunto de validação utilizado nesse projeto possui tamanho diferente do de Guerbai et al, e assim como o conjunto de treinamento, possui somente assinaturas genuínas. Foi utilizada, para construção do conjunto de validação, a metade das assinaturas genuínas que sobraram após a formação do conjunto de treinamento e a outra metade foi utilizada para formar o conjunto de teste. O conjunto de validação foi utilizado para calcular o *threshold* de cada escritor que é usado na fase de teste, diferentemente de Guerbai et al. que utilizou apenas um *threshold* para todos os escritores. A utilização de um *threshold* para cada usuário caracteriza o sistema como dependente do escritor, sendo beneficiado com as vantagens desse sistema, apresentado na Introdução.

O cálculo do *threshold* é um passo crucial na criação da máquina e foi calculado utilizando a fórmula (12), a mesma utilizada por Guerbai et al. Apesar da fórmula ser a mesma, os valores de k, m_f e α_f foram encontrados de formas diferentes. Como cada escritor possui o seu próprio *threshold*, a média e o desvio padrão foram encontrados utilizando apenas as imagens da mesma classe, diferentemente de Guerbai et al. que utilizou o conjunto de todos os escritores para achar um *threshold* único. Já o valor de k foi calculo usando o mesmo range ([-3,3]), porém foi iterado com valores menores, de 0.01 em 0.01. Para a escolha do valor da constante k, foi respeitada a mesma regra: $FAR = FRR$.

6.2.3 Teste

Já o conjunto de teste, cujo objetivo é testar a eficiência do método proposto, foi construído a partir da união dos dados das assinaturas falsas com

os dados das assinaturas genuínas que sobraram depois da formação dos conjuntos de treinamento e de validação. Para alcançar tal objetivo, além das métricas de avaliação da fase de validação (FRR e FAR), foi também utilizada a métrica AER, definida pela fórmula (3).

7. Resultados

Nesse capitulo serão apresentados os resultados obtidos pelo sistema desenvolvido nesse projeto. Será feita uma comparação entre os resultados obtidos por Guerbai et al [1]. e os resultados obtidos nesse trabalho de graduação. Entretanto, antes disso, será feita uma breve análise do banco de dados e das ferramentas utilizadas para o desenvolvimento da máquina.

7.1 Ferramentas utilizadas

Todo o sistema desenvolvido por Guerbai et al. foi reimplementado neste trabalho, porém para uma comparação mais justa dos resultados, foram utilizadas as mesmas ferramentas. São elas:

- Matlab
- Libsvm
- Curvelab

Além dessas utilizadas por Guerbai et al., a seguinte ferramenta foi usada para testar a eficiência do método de extração de característica:

- Weka

7.1.1 Matlab

Matlab é uma plataforma otimizada para resolução de problemas da engenharia e da ciência. A linguagem baseada em matriz é a forma mais natural do mundo para expressar matemática computacional [20]. Além desse motivo, a plataforma Matlab foi escolhida por causa da compatibilidade com as bibliotecas Curvelab [21] e Libsvm [22]. Todo o projeto foi implementado utilizando está linguagem.

7.1.2 Libsvm

LIBSVM [22] é uma biblioteca para Support Vector Machines (SVMs). A biblioteca foi desenvolvida em 2000, com o objetivo de ajudar os usuários a aplicar facilmente SVM às suas aplicações. A LIBSVM, cujo o desenvolvimento aconteceu na Universidade Nacional de Taiwan, ganhou grande popularidade na aprendizagem de máquinas e em muitas outras áreas, facilitando o uso da SVM tanto para classificação quanto para regressão.

7.1.3 Weka

Weka é uma coleção de algoritmos de aprendizagem de máquina, que podem ser tantos aplicados a uma base de dados diretamente ou chamados pelos código de JAVA, para tarefas de mineração de dados. Weka contém ferramentas para pré-processamento de dados, classificação, regressão, agrupamento, regras de associação e visualização. Também é bem adequado para desenvolvimento de novos esquemas de máquina de aprendizagem [25].

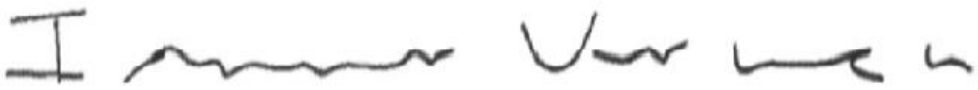
O Weka foi utilizado pela sua facilidade de testar bancos de dados e também por possuir a biblioteca utilizada no projeto, a Libsvm.

7.2 Coleta dos dados

Os dados utilizados nesse projeto também foram fornecidos pelo GPDS. Houve uma tentativa de também adquirir o bando de dados do CEDAR, porém sem sucesso. Apesar do banco ter sido fornecido pelo GPDS, ele é diferente do banco utilizado por Guerbai et al, enquanto o banco utilizado no artigo possuía trezentos escritores, o banco utilizado nesse trabalho possui quatro mil.

O banco utilizado foi o GPDSsyntheticSignature. Apesar da quantidade de escritores ser diferente, ele possui a mesma quantidade de assinaturas genuínas (24) e falsas (30) que o utilizado por Guerbai et al. A Figura 10a e 10b possui exemplos de assinatura genuína e falsa respectivamente.

(a)



(b)

Figura 10 – Assinaturas genuína (a) e falsa (b)

7.3 Resultados malsucedidos

Antes de mostrar os dados reais, primeiro será analisado os resultados obtidos usando os métodos e os parâmetros utilizados no artigo de Guerbai et al.[1]. Com a extração de características usando a energia, a fase de treinamento da One-Class SVM não obteve uma boa generalização da máquina, o que gerou classificações com valores na faixa de 0 à 10% de taxa de acerto. Como os parâmetros da Curvelet não foram especificados no artigo, foram executadas várias tentativas com os parâmetros escala e ângulo possuindo valores distintos, porém sem sucesso.

Depois de tentar vários parâmetros diferentes da Curvelet, foi testado também a modificação da fase do pré-processamento. Como as imagens das assinaturas manuscritas eram muito grandes, em média possuem o tamanho 996x346, acabam retornando uma quantidade elevada de coeficientes. Para tentar diminuir essa quantidade, foram testados dois tamanhos de imagens encontrados na literatura: 128x128 [5] e 256x256 [24]. Porém apesar dessa tentativa, a taxa de acerto do sistema não se alterou.

Por último foi tentado a validação cruzada para cada escritor, com o objetivo de verificar se os parâmetros utilizados no treinamento da máquina de aprendizagem estavam errados, porém essa tentativa também não obteve sucesso.

7.4 Resultados bem-sucedidos

Após as falhas nas tentativas citadas no tópico anterior, foi usada a técnica do desvio padrão para criação do vetor de característica e, logo no primeiro teste, o vetor de característica construído conseguiu gerar uma OC-SVM com média de 80% de acerto para alguns escritores escolhidos aleatoriamente. Com isso, foi executada uma fase de testes para encontrar os parâmetros da Transformada Curvelet que acarretaria numa melhor generalização da OC-SVM. Os valores da escala e do ângulo selecionado foram quatro e dezesseis, respectivamente.

Com os vetores de características gerados, foram executados testes para encontrar os melhores parâmetros das OC-SVM. O caso ideal seria encontrar os parâmetros de cada uma das cinco OC-SVM que cada um dos trezentos escritores possui, entretanto, isso seria um processo custoso. Então foram escolhidos dez escritores e foi executada a validação cruzada no Weka modificando os parâmetros nu e γ. Os valores dos parâmetros selecionados foram 0.2 e valor default da Libsvm, respectivamente.

Com os parâmetros selecionados, foram executados os mesmos testes do artigo escrito por Guerbai et al. A Figura 11 mostra o gráfico da taxa de erro obtidos no presente trabalho, enquanto a Figura 12 mostra o gráfico com os dados gerados por [1].

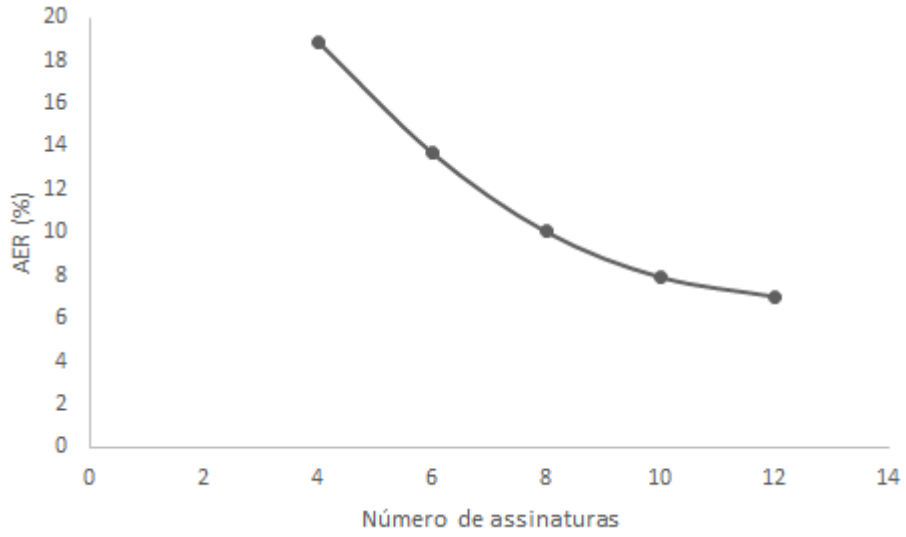


Figura 11 – Taxa de erro vs quantidade de assinaturas dos escritores na fase de treinamento

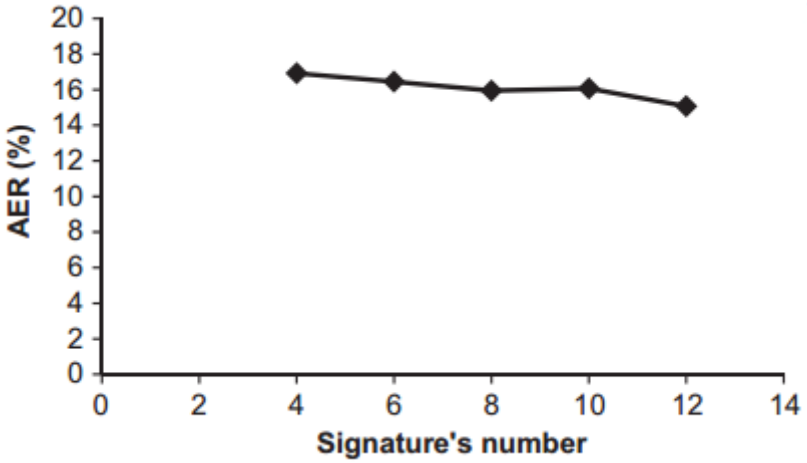


Figura 12 – Taxa de erro vs quantidade de assinaturas dos escritores na fase de treinamento do sistema de Guerbai et al [1]

Como pode-se verificar, o sistema desenvolvido por Guerbai et al. possui uma pequena dependência em relação a quantidade de assinaturas usadas na fase de treinamento, enquanto o presente projeto possui uma maior dependência em relação à quantidade de assinaturas usadas. É notório que com um conjunto de quatro assinaturas, o sistema de Guerbai et al. apresentou uma melhor taxa em relação a este trabalho de graduação, entretanto a partir de seis assinaturas,

o presente trabalho obteve melhores resultados. Esses comportamentos eram esperados já que no sistema desenvolvido por [1], o *threshold* foi global, tirando a dependência de cada escritor em relação aos seus parâmetros, enquanto que o sistema desenvolvido nesse trabalho possui um *threshold* para cada escritor. Essa diferença no processo de obtenção do *threshold*, faz com que o sistema proposto neste trabalho, quando um novo cliente é inserido, possua um custo computacionalmente maior, por outro lado, a máquina apresenta uma taxa de erro menor na identificação das assinaturas. Enquanto que no sistema proposto por Guerbai et al, o resultado é o oposto, fazendo com que o verificador de assinaturas manuscritas possua um custo computacionalmente menor e com uma taxa de erro maior.

Apesar do custo computacional menor na inserção de novos usuários, o sistema proposto por Guerbai et al [1], quando necessário recalcular os parâmetros de algum cliente específico, apresenta um custo computacionalmente alto, pois nesse caso é necessário utilizar os dados dos clientes já cadastrados, enquanto que no sistema proposto nesse trabalho, seria necessário apenas os dados do usuário em questão para geração dos seus parâmetros. Já no sistema desenvolvido nesse trabalho, quando necessário inserir um novo usuário no sistema, só é necessário calcular os seus parâmetros individualmente, fazendo com que, nesse caso, o sistema apresente um custo computacionalmente menor que o sistema proposto por Guerbai et al [1].

Em sistemas de assinaturas manuscritas, é desejável que as taxas FAR e FRR possuam valores baixos. Porém, caso não seja possível, é preferível que o sistema apresente uma boa taxa de FAR em detrimento ao FRR, pois aceitar uma assinatura falsa como verdadeira pode gerar um prejuízo financeiro, enquanto que rejeitar uma assinatura verdadeira gera, apenas, um atraso no processo, pois nesse caso é necessário confirmar os dados com o cliente. A Tabela 3 mostra que, nesse trabalho, a taxa de assinaturas falsas classificadas como verdadeiras obteve um valor muito baixo, o que faz com que o sistema proposto apresente uma boa segurança. Todavia, também é evidente que a quantidade de assinaturas positivas classificadas como falsas (FRR) ficou um pouco acima do desejado, porém é notório que a taxa FRR, nesse caso, é inversamente proporcional ao tamanho do conjunto de treinamento.

Assinaturas	FRR (%)	FAR (%)	AER (%)
4	37.60	0.17	18.89
6	27.22	0.19	13.71
8	19.79	0.27	10.03
10	15.57	0.28	7.92
12	13.78	0.26	7.02

Tabela 3 – Taxas de erro em relação a quantidade de assinaturas genuínas utilizadas na fase de treinamento

Na Tabela 4, foi feita uma comparação com alguns resultados obtidos em sistemas propostos para classificação de assinaturas manuscritas que utilizaram o banco GPDS. Os dados obtidos foram retirados da tabela 6 do artigo de Guerbai et al. Podemos ver que os resultados gerados nesse trabalho só são piores quando se usa um conjunto de treinamento com quatro assinaturas, porém a partir de oito assinaturas, o sistema proposto obteve os melhores resultados.

Referências	Classificadores	Características	Assinaturas para treinamento	AER (%)
Kumar et al [28]	Rede Neural	Surroundedness	24 genuinas + 24 falsas	13.76
Batista et al [11]	HMM + SVM	Grid Segmentation	4 Genuínas	20.53
			8 Genuínas	17.24
			12 Genuínas	16.84
Guerbai et al [1]	OC–SVM	Curvelet	4 Genuínas	16.92
			8 Genuínas	15.95
			12 Genuínas	15.07
Presente trabalho	OC-SVM	Curvelet	4 Genuínas	18.89
			8 Genuínas	10.03
			12 Genuínas	7.02

Tabela 4 – Taxa de erro do sistema proposto neste trabalho e de outros sistemas que usaram banco GPDS

8. Conclusão e Trabalhos futuros

Nesse trabalho foi apresentada uma evolução do sistema de reconhecimento de padrões para classificação de assinaturas manuscritas baseado na técnica independente do escritor apresentada por Guerbai et al., em cujo trabalho este se baseou. O objetivo era diminuir a taxa de erro e como podemos ver, o objetivo do trabalho foi alcançado, pois obteve melhores resultados que o artigo base.

O diferencial do trabalho está na técnica de extração de características. Muitos artigos utilizam desvio padrão junto com a média ou com a energia para gerar o vetor de características do dados. Nesse trabalho foi utilizada apenas a técnica do desvio padrão, e se baseando nos resultados obtidos, pode-se afirmar que a técnica conseguiu criar uma boa generalização do sistema.

O principal mérito do método proposto é, a partir de seis assinaturas na fase de treinamento, ter melhorado a taxa de AER em relação aos trabalhos da Tabela 4. Essa melhora na taxa, aliado com a pequena quantidade de assinaturas usadas na fase de treinamento, faz com que seja viável o uso dessa ferramenta no dia a dia de empresas que usam a assinatura como método de reconhecimento dos seus clientes.

Como era previsto, a técnica *off-line* dependente do assinante, utilizada nesse trabalho, gerou melhores resultados na taxa AER em relação a técnica independente do assinante utilizada por Guerbai et al. Porém como também era de se esperar, quando a quantidade de amostras são pequenas na fase de treinamento, o resultado obtido nesse trabalho ficou um pouco atrás em relação aos trabalhos comparados na Tabela 4.

Porém ainda existe melhorias possíveis para geração de resultados ainda melhores em trabalhos futuros. É valido analisar se algumas das cinco distâncias utilizadas na criação das OC-SVM são necessárias, com o objetivo de diminuir

o custo computacional. Com isso, poderia ser viável a utilização da validação cruzada para encontrar os parâmetros ideais de cada OC-SVM de cada escritor, gerando um sistema reconhecedor de padrão mais robusto.

Também é possível notar que os valores do *thresholds* utilizados geraram uma taxa FRR muito alta, seria interessante no futuro tentar utilizar uma técnica que encontre *thresholds* que equilibre mais as taxas FRR e FAR.

Por último, outra investigação interessante consiste em utilizar outras formas de extrair as características das imagens junto com a Transformada Curvelet. Existem várias técnicas utilizadas em conjunto com a Curvelet na literatura como: PCA e PCA em conjunto com a LDA. A utilização de algumas dessas técnicas poderia resultar em uma máquina mais robusta, pois eliminaria dados ruidosos ou redundantes presentes no vetor de características.

9. Referências

[1] Y. Guerbai, Y. Chibani, and B. Hadjadji, *The effective use of the one-class SVM classifier for handwritten signature verification based on writer-independent parameters*, Pattern Recognition, vol. 48, no. 1, pp. 103–113, Jan. 2015.

[2] Sangram Bana and Davinder Kaur, *Fingerprint Recognition using Image Segmentation*, International Journal Of Advanced Engineering Sciences And Technologies (IJAEST), Vol. 5, Issue 1, pp. 012 - 023, 2011.

[3] Santos. C. Análise de Assinaturas Manuscritas Baseada nos Princípios da Grafoscopia, 2004. 148 f. Dissertação de Mestrado em Ciências da Computação – Universidade Federal Católica do Paraná – Curitiba.

[4] Franco. D, Barboza. F, Freitas. M, Cardoso. N. Uma Ferramenta Computacional Forense para Verificação de Autenticidade de Assinaturas Manuscritas Através de Processamento Digital de Imagens e Redes Neurais Artificiais, Computer on the Beach 2015, pp. 121 – 130, 2015.

[5] F. Liu, L. Zhou, Z.-M. Lu, T. Nie, *Palmprint feature extraction based on Curvelet Transform,* Journal of Information Hiding and Multimedia Signal Processing 6(1):131-139 · January 2015.

[6] D.L. Donoho, M.R. Duncan, *Digital Curvelets Transform: Strategy, Implementation and Experiments*, Stanford University, 1999.

[7] B. Schölkopf, J. Platt, J. Shawe-Taylor, A. Smola, R. Williamson, *Estimating the support of a high dimensional distribution*, Neural Comput. 13 (7) (2001) 1443–1472.

[8] S.N. Srihari, A. Xu, M.K. Kalera, *Learning strategies and classification methods for off-line signature verification*, in: Proceedings of the Ninth International Workshop on Frontiers in Handwriting Recognition IWFHR 04, 2004.

[10] M. Ferrer, J. Alonso, C. Travieso, *Offline geometric parameters for automatic signature verification using fixed-point arithmetic*, IEEE Trans. Pattern Anal. Mach. Intell. 27 (6) (2005) 993–997.

[11] L. Batista, E. Granger, R. Sabourin, *Dynamic selection of generativediscriminative ensembles for off-line signature verification*, Pattern Recognit. 45 (2012) 1326–1340.

[12] Lorena, A. C.; Carvalho, A. C. P. L. F. Uma Introdução às *Support Vector Machines*, Revista de Informática Teórica e Aplicada, vol.14, no2, pp 43-67, 2007.

[13] Santos, D. Avaliação de técnicas de aprendizagem de máquina para predição de engajamento com user-generated content, 2016. 62 f. Trabalho de Graduação (Bacharelado em Engenharia da Computação) - Universidade Federal de Pernambuco – Recife.

[14] Santos, E. Teoria e Aplicação de *Support Vector Machines* à Aprendizagem e Reconhecimento de Objetos Baseado na Aparência**,** 2012. 121 f. Dissertação de Mestrado em Informática – Universidade Federal da Paraiba – João Pessoa.

[15] Albuquerque, R. Monitoramento da cobertura do solo no entorno de hidrelétricas utilizando o classificador SVM (Support Vector Machines), 2012. 107 f. Dissertação de Mestrado em Engenharia de Transportes – Universidade de São Paulo – São Paulo.

[16] HOFMANN, T., SCHöLKOPF, B., SMOLA, A. *Kernel methods in machine learning*, Ann. Statist. Volume 36, Number 3 (2008).

[17] Sumana, I. *Image Retrieval Using Discrete Curvelet Transform*, 2008. 98 f. Dissertação de Mestrado – Monash University Gippsland – Churchill.

[18] Candes, E. J. & Donoho, D. L. (2000). *Curvelets—A surprisingly effective non- adaptive representation for objects with Edges*, Vanderbilt University Press, Nashville, TN, 2000.

[19] Guha, T., Majmudar, A., WU, Q. *Face recognition by Curvelet based feature extraction*, International Conference on Intelligent Automation and Robotics, LNCS 4633, 2007, pp. 806– 817.

[20] MATLAB. "MATLAB". Disponível em: http://www.mathworks.com/products/matlab/, 2016, acessado em 10/12/2016.

[21] E. Candés, L. Demanet, L. Ying, *CurveLab Toolbox*, Version 2.0.3, Communication. Pure application, Mathematics 57 (2004) 219–266.

[22] Chih-Chung Chang and Chih-Jen Lin, *LIBSVM : a library for support vector machines*. ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[23] T. Ridler, S. Calvard, *Picture thresholding using an iterative selection method*, IEEE Trans. Syst. Man Cybern. 8 (8) (1978) 630–632.

[24] M.S. Shirdhonkar, M. Kokare, *Off-line handwritten signature retrieval using curvelet transforms*, Int. J. Comput. Eng. 3 (4) (2011) 1658–1665.

[25] Refaeilzadeh, P., TANG, L., LIU, H. *Cross Validation*. In Encyclopedia of Database Systems, Editors: M. Tamer Ã-zsu and Ling Liu. Springer, 2009.

[26] S. Geisser. *The predictive sample reuse method with applications.* Jornal of the American Statistical Association, 70(350), 1975.

[27] Meloni, R. *Classificação de Imagens de Sensoriamento Remoto usando SVM*, 2009. 63 f. Dissertação de Mestrado – Pontifícia Universidade Católica do Rio de Janeiro – Rio de Janeiro.

[28] R. Kumar, J.D. Sharma, B. Chanda, *Writer-independent off-line signature verification using surroundedness feature*, Pattern Recognit. Lett. 33 (2012) 301–308.

[29] H. Zhang, D. Hu, *A palm vein recognition system*, International Conference on Intelligent Computation Technology and Automation, vol.1, pp. 285-288, 2010.

[30] E. J. R. Justino, F. Bortolozzi and R. Sabourin, *A comparison of SVM and HMM classifiers in the off-line signature verification*, Pattern Recognition Letters 26 (2005) 1377-1385.

[31] S. N. Srihari, S. Cha, H. Arora, and S. Lee. *Individuality of handwriting. Journal of Forensic Sciences*, 47(4):856–872, July 2002.

[32] David Lewis. *Naive (bayes) at forty: The independence asssumption in information retrieval*. In ECML'98: Tenth European Conference On Machine Learning, 1998.

[33] L.L. Kuncheva, *Combining Pattern Classifiers:Methods and Algorithms*, WileyInterscience Publication, New Jersey, USA, 2004.

[34] K. I. Kim, K. Jung, S. H. Park, and H. J. Kim. *Support vector machines for texture classification*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 24(11):1542–1550, 2002.

[35] W. S. Noble. *Support vector machine applications in computational biology*. In B. Schölkopf, K. Tsuda, and J.-P. Vert, editors, Kernel Methods in computational biology, pages 71–92. MIT Press, 2004.

[36] B. Schölkopf, I. Guyon, and J. Weston. *Statistical learning and kernel methods in bioinformatics*. In P. Frasconi and R. Shamir, editors, Artificial Intelligence and Heuristic Methods in Bioinformatics, pages 1–21. IOS Press, 2003.

[37] I. Daubechies. *The wavelet transform, time-frequency localization and signal analysis*, IEEE Trans. Information Theory, Vol. 36, No. 5, 961-1005, 1990

[38] L. Chen, G. Lu, and D. S. Zhang. *Effects of Different Gabor Filter Parameters on Image Retrieval by Texture*, in Proc. of IEEE 10th International Conference on Multi-Media Modelling, Australia, 2004, pp. 273-278.

[39] WEKA. “WEKA”. Disponível em: http://www.cs.waikato.ac.nz/ml/weka/, 2016, acessado em 15/12/2016.

[40] L. Manevitz, M. Yousef. *One-Class SVMs for Document Classification*, Journal of Machine Learning Research, vol. 2, pp. 139-154, 2001.