UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

GUILHERME BALDO BENDER

Análise de Redes Neurais Artificiais Aplicadas a um Sistema em Tempo Real de Reconhecimento de Gestos Estáticos de Mão

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Prof. Dr. Dante A. C. Barone

Porto Alegre
2014

AGRADECIMENTOS

Sem mencionar nomes, a todos aqueles que participaram desta jornada.

RESUMO

Situado nos campos de visão computacional e de interação homem-máquina, este trabalho desenvolveu uma análise de redes neurais MPLs como técnica de classificação em um sistema de reconhecimento de gestos estáticos de mão. A entrada do sistema é fornecida por uma câmera comum. Do frame atual, após um processo de segmentação da mão, são extraídos como vetor de atributos os momentos invariantes de Hu, além de uma relação entre perímetro e raiz da área. Foram utilizadas seis classes de gestos, sendo escolhidas segundo a maximização das mínimas distâncias entre elas no espaço de atributos. Considerando-se diferentes formas de rotular-se os padrões de treinamento, a análise consistiu na busca da MLP mais bem adaptada, sendo para isso realizado uma variação de dois parâmetros: número de frames sobre os quais as redes atuam; e número de neurônios na camada oculta. A melhor rede encontrada obteve acurácia de 99.12%, não havendo nenhuma confusão entre as classes de gestos escolhidas, mas apenas nas regiões de transição entre elas.

Palavras-chave: Redes Neurais Artificiais. MLPs. Momentos Invariantes de Hu. Relação Entre Perímetro e Raiz da Área. Reconhecimento de Gestos. Interação Homem Máquina. Visão Computacional.

Analysis of Artificial Neural Networks Applied to a Real-Time System for Static Hand Gestures Recognition

ABSTRACT

Belonging to the fields of computer vision and human-machine interaction, this paper developed an analysis of MLPs neural networks used as a classification technique in a system for the recognition of static hand gestures. The system's input is provided by an ordinary camera. From the current frame, after a segmentation process of the hand, are extracted feature vectors composed by Hu invariant moments and the ratio between perimeter and square root of area. Six classes of gestures, being chosen accordingly to the maximization of the minimum distances between them in the space of attributes were used. Considering different ways to label the patterns of training, the analysis consisted in the search for the most adapted MLP, for which were made a variation of two parameters: the number of frames over which the networks operate; and the number of neurons in the hidden layer. The best network found had an accuracy of 99.12%, and there were no confusion between the chosen classes of gestures, except for the transition regions between them.

Keywords: Artificial Neural Networks. MLPs. Hu Invariant Moments. Ratio Between Perimeter And Square Root of Area. Recognition of Gestures. Human Machine Interaction. Computer Vision.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

EIS	Erros de identificação de segmentos
IDE	Integrated Development Environment
fps	Frames por segundo
MLP	Multilayer Perceptron
RNA	Redes Neural Artificial
SNI	Segmentos não identificados
SFI	Segmentos falsamente identificados

SUMÁRIO

1 INTRODUÇÃO

Desenvolveu-se nesse trabalho uma análise de redes neurais MLPs, utilizadas como técnica de classificação de um sistema em tempo real de reconhecimento de gestos estáticos de mão. O sistema tem como objetivo reconhecer comandos simples que são capturados por uma câmera de vídeo comum. Ele é caracterizado por ser em tempo real pois o seu classificador, para determinar o gesto presente, faz uso somente do *frame* atual ou dos anteriores. As classes de gestos que foram consideradas são estáticas, isto é, com apenas uma imagem é possível caracterizá-las. Contudo, a análise não é realizada sobre imagens isoladas, mas sim sobre sequências de vídeo. Isso por que também foram consideradas as transições entre a realização de um gesto e outro.

A análise consistiu em encontrar a rede MLP mais bem adaptada, sendo que a busca por tal rede foi feita a partir de uma variação extensiva de dois de seus parâmetros: número de frames sobre os quais atua; e número de neurônios na camada oculta. Com o intuito de gerar-se classificadores robustos, especial atenção foi dedicada na escolha das classes de gestos e na rotulação dos padrões de referência da base dados. Não sendo a segmentação de imagens o foco do trabalho, a coleta de vídeos foi realizada de modo a facilitar-se esse procedimento.

Quanto a organização do trabalho, após a sessão introdutória, há quatro capítulos principais:

a. *Técnicas e metodologias utilizadas*. Em que se expõe os procedimentos de segmentação de imagens, extração do vetor de atributos, além do modelo e algoritmo de treinamento das redes neurais.
b. *Arquitetura do sistema de reconhecimento de gestos.* Em que se define a arquitetura do sistema em que as MLPs foram analisadas como classificadores.
c. *Formação da base de dados rotulados*. Em que se apresenta como foi criada a base de dados para o treinamento, validação e teste das MLPs, o que envolveu a definição das classes de gestos, coleta de amostras de vídeos, e rotulação de cada um de seus *frames*.
d. *Resultados.* Em que se define as métricas de avaliação das redes neurais e se apresenta os experimentos realizados.

Com relação aos meios empregados para a realização do trabalho, na coleta de vídeos utilizou-se uma webcam LifeCam HD-3000 da Microsoft. Já para os procedimentos de

segmentação de imagens, extração de atributos, e formação da base de dados utilizou-se a biblioteca de processamento de imagens OpenCV 2.48, sendo a programação realizada na IDE Visual Studio 2012 da Microsoft. A análise das MLPs foram realizadas no software de cálculo numérico MATLAB, versão 7.9.0.

1.1 Motivação e aplicações

Imagine-se deitado no sofá, de olhos fechados, ouvindo as suas músicas favoritas. Você estende a mão no ar, aponta para a direita, e a próxima música começa a tocar; aponta para a esquerda, e a música anterior retorna; ou então apenas abre a mão e a música para. Em outra situação, você está no meio de tráfego intenso, navegando pela cidade utilizando-se do GPS do seu *smartphone*, que está preso a um suporte. É preciso fazer uma atualização de rota, ao mesmo tempo em que a sua atenção deve ficar voltada aos veículos ao redor. Focalizar a visão na tela do *smartphone* e mirar o dedo nos pequenos botões do aplicativo poderia ser arriscado. Nesse caso, a tarefa certamente seria mais fácil e segura se simplesmente se pudesse gesticular no ar o que se deseja.

Esses são dois exemplos de situações cotidianas em que o uso de um sistema de reconhecimento de gestos em tempo real poderia ser conveniente. Há ainda outras, como comandar aplicativos no seu notebook à distância e prover uma interface de interação para jogos mais livre.

No mercado já existem tecnologias que permitem o reconhecimento de gestos, como o sensor Kinect da Microsoft, o controle Wii da Nintendo, o controle PlaystationMove da Sony, além de luvas virtuais de uma variedade de outras empresas. Contudo, o reconhecimento não é feito com o uso de apenas uma câmera comum. Em geral há um hardware dedicado ou especial, com sensores infravermelho de profundidade, câmeras estéreo, controles com acelerômetros ou magnetômetros.

Como atualmente grande parte dos *smartphones*, *tablets* e *notebooks* já vêm com uma câmera de vídeo integrada, a motivação desse trabalho é investigar uma técnica de classificação que poderia fazer parte de um sistema de reconhecimento de gestos de mão que faça uso apenas desse recurso. O sistema deve permitir o envio de comandos simples a computadores, possibilitando realizar, de forma robusta, interações como as descritas no primeiro parágrafo. As MLPs, que possuem uma computação direta e eficiente, treinadas para atuarem sobre atributos de imagem fáceis de extrair, pareceram boas candidatas à tarefa.

1.2 Objetivos do Trabalho

1.2.1 Objetivo Geral

Avaliar a eficácia de redes neurais MLPs de uma camada oculta para o reconhecimento de gestos em sequências de vídeo, treinadas com algoritmo *backpropagation* de aprendizado supervisionado, utilizando-se dados favoráveis ao treinamento.

1.2.2 Objetivos Específicos

Definir classes de gestos, procedimentos de segmentação de imagens e de rotulação, com o intuito de gerar dados de treinamento que permitam as MLPs alcançarem bom desempenho. Com esses dados, analisar de modo sistemático, visando descobrir-se a arquitetura mais adequada, os efeitos da variação de dois parâmetros das MLPs: número de frames sobre o qual a rede atuará; e número de neurônios na camada oculta.

1.3 Trabalhos relacionados

Embora entre os artigos pesquisados durante a elaboração deste trabalho uma grande quantidade deles utilize redes neurais como técnica de reconhecimento de padrões, nenhum dos mesmos apresentou um estudo extensivo sobre os efeitos da variação das topologias de rede no desempenho dos sistemas de reconhecimento de gestos.

No trabalho desenvolvido por G.R.S. Murthy e R.S. Jadon - “Hand Gesture Recognition using Neural Networks” [1] - é proposto um sistema de reconhecimento de gestos de mão que utiliza como técnica de reconhecimento de padrões redes neurais artificiais sem realimentação, treinadas de modo supervisionado, com algoritmo backpropagation, para o reconhecimento de 10 categorias de gestos de mão. Os dados são obtidos por uma webcam e passam por um processo de segmentação, sendo gerado uma imagem binária, que posteriormente é redimensionada para ter uma resolução de 30x30. Tal imagem é utilizada como entrada da rede neural, que foi escolhida para ter 7 neurônios na camada oculta. A precisão média do sistema relatada pelos autores foi de 89%.

Já no trabalho desenvolvido por Mu-Chun Su, Woung-Fei Jean, e Hsiao-Te Chang - "A Static Hand Gesture Recognition Using a Composite Neural Network." [2] - é proposto um sistema em tempo real para o reconhecimento de gestos que utiliza como dados de entrada dez medidas dos ângulos das juntas dos dedos da mão, fornecidas por uma luva especial. O

reconhecimento é realizado por meio de uma rede neural composta, treinada de modo supervisionado pelo algoritmo SDDL (supervised decision-directed learning). O sistema foi avaliado para a classificação de 51 gestos estáticos de mão, sendo cada um deles realizados 10 vezes por 4 pessoas, formando um total de 2040 dados de base, ficando 75% destes para o treinamento da rede, e os 25% restantes para teste. A acurácia relatada do sistema foi de 100% para os dados de treinamento e de 93.9% para os dados de teste.

Assim como no trabalho anterior, o desenvolvido por Pedro Neto, Dário Pereira, et al. - “Real-Time and Continuous Hand Gesture Spotting: an Approach Based on Artificial Neural Networks ” [3] – também propõe um sistema de reconhecimento de gestos de tempo real cuja entrada de dados é aquela fornecida por luvas especias. O sistema é treinado de modo supervisionado para realizar o controle de um robô industrial por meio de gestos de mão. Na arquitetura do sistema são utilizadas duas redes neurais em série, sendo a primeira utilizada para reconhecer se um gesto é comunicativo ou não-comunicativo (corresponde a transição entre gestos) está sendo feito, e a segunda para classificar os gestos comunicativos em suas categorias específicas. Ambas redes neurais são sem realimentação, com uma camada oculta, treinadas com o algoritmo backpropagation. Nos resultados experimentais, é relatado uma taxa de precisão de 99% para o reconhecimento de dez gestos e de 96% para o reconhecimento de trinta gestos.

Além das técnicas de redes neurais, diversas propostas de sistemas de reconhecimento de gestos de mão foram desenvolvidas. Entre outras técnicas, pode-se destacar: modelos ocultos de Markov (HMM – hidden markov models) [4][5][6]; árvores de decisão fuzzy [7]; e máquinas de vetor de suporte (SVM – support vector machine) [8].

2 TÉCNICAS E METODOLOGIAS UTILIZADAS

2.1 Segmentação de imagens

Segmentação de imagens, no contexto de visão computacional, é o processo pelo qual uma imagem digital é particionada em múltiplas regiões, i.e., em diversos conjuntos de pixels. Em geral, o objetivo de tal particionamento é a facilitação da análise da imagem ou então a obtenção de um outro modo de representá-la.

O resultado da segmentação é um conjunto de regiões que cobrem toda a imagem, sendo que cada uma delas é caracterizada pelo conjunto de propriedades comuns que os seus pixels possuem, como, por exemplo, semelhança de textura, intensidade de cor, pertinência ao contorno ou então ao corpo de um dado objeto.

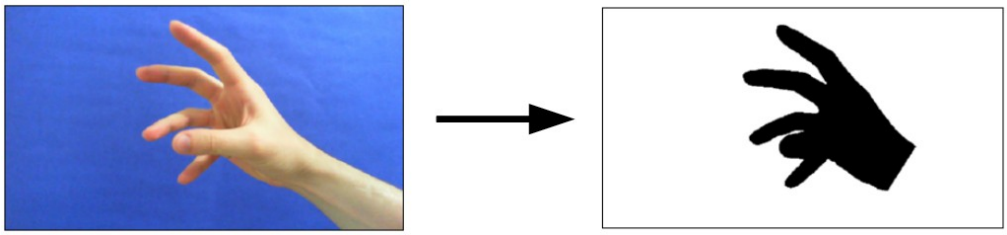
O processo de segmentação de imagens encontra aplicações práticas em diversos campos. Na área médica, a partir de imagens de raio-X, imagens por ressonância magnética ou ultrassonografia, é possível identificar patologias ou tumores, medir a área e volume de tecidos, realizar diagnósticos e analisar estruturas anatômicas. Na área industrial, a segmentação de imagens é a base para dar visão de máquina a robôs, capacitando-os a identificarem demarcadores, objetos e pessoas, de modo que naveguem pelo ambiente com maior segurança.

Há uma gama variada de técnicas de segmentação de imagens. Entre as principais se encontram: aquelas baseadas em formas, que são definidas a partir da detecção de descontinuidades, pontos, linhas e contornos; as baseadas em propriedades dos pixeis, que realizam segmentação de cores e intensidades; e as baseadas em histogramas (curva de frequência das intensidades dos pixeis), que determinam a região a que pertence cada pixel a partir de sua localização entre determinados picos ou vales do histograma [9].

2.1.1 Método de segmentação utilizado

O método de segmentação utilizado tem como objetivo a transformação de uma imagem no espaço de cores aditivo RGB que contenha a imagem de uma mão e uma porção do antebraço em uma imagem binária que descreva apenas a região da mão. Para isso, tem-se como premissa que o plano de fundo da imagem seja conhecido. A Figura 2.1 ilustra o resultado desejado.

Figura 2.1: Resultado desejado da segmentação de imagens



Fonte: elaborado pelo autor.

2.1.1.1 Segmentação da mão e do antebraço contra o plano de fundo

Seja $F(x,y) = \langle F_R(x,y), F_G(x,y), F_B(x,y) \rangle$ a função que descreve o plano de fundo da imagem, $I(x,y) = \langle I_R(x,y), I_G(x,y), I_B(x,y) \rangle$ a função que descreve a imagem a ser segmentada – ambas no espaço de cores RGB - e $B(x,y)$ a imagem binária segmentada.

Idealmente, se o processo de captura de imagens não sofresse a interferência de ruído e supondo-se que a cor de cada pixel do objeto em primeiro plano fosse diferente da cor do pixel correspondente do plano de fundo, a segmentação da imagem $I(x,y)$ seria extremamente simples, pois bastaria aplicar uma comparação direta de seus pixeis com os de $F(x,y)$, ou seja:

$$B(x,y) = \begin{cases} 1, & se \quad F(x,y) \neq I(x,y) \\ 0, & se \quad F(x,y) = I(x,y) \end{cases}$$

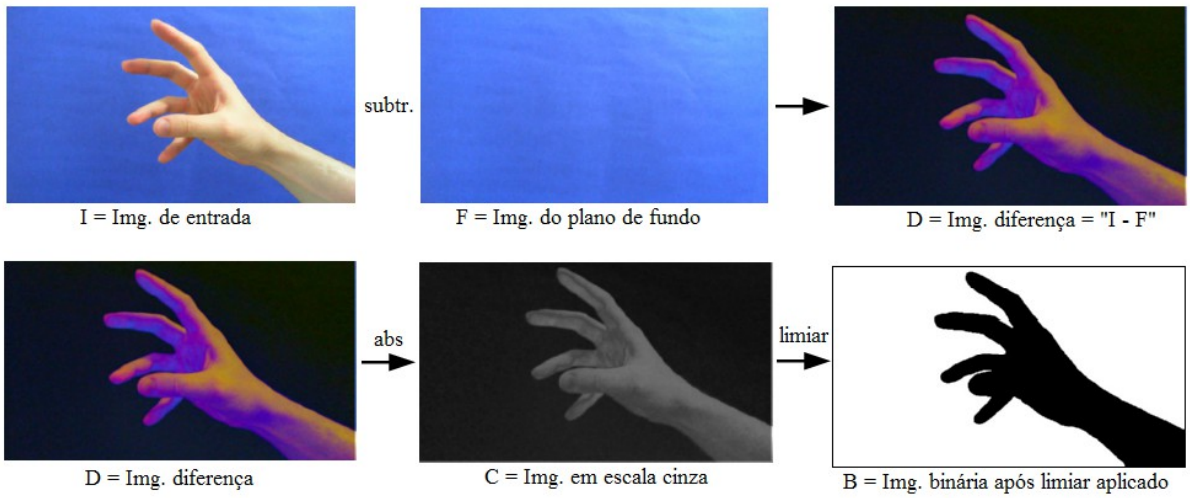
No entanto, como não há como escapar do ruído, a probabilidade de que nem todos pixeis de F sejam iguais aos seus correspondentes em I é alta – na verdade, praticamente nenhum o será. Desse modo, a função de segmentação deve ser modificada. A solução utilizada é simples: primeiro calcula-se a diferença entre F e I aplicando-se a função módulo a cada uma das componentes de cores, gerando-se a imagem D; depois se transforma a imagem D para a escala cinza, gerando-se a imagem C; por fim segmenta-se C a partir da aplicação de um limiar λ. O processo é ilustrado pela Figura 2.2 e descrito pelas equações:

$$D(x,y) = \langle D_R(x,y), D_G(x,y), D_B(x,y) \rangle, \quad onde \quad D_i(x,y) = |I_i(x,y) - F_i(x,y)|$$

$$C(x,y) = D_R(x,y) + D_G(x,y) + D_B(x,y)$$

$$B(x,y) = \begin{cases} 1, & se \quad C(x,y) \geqslant \lambda \\ 0, & se \quad C(x,y) < \lambda \end{cases}$$

Figura 2.2: Processo de segmentação

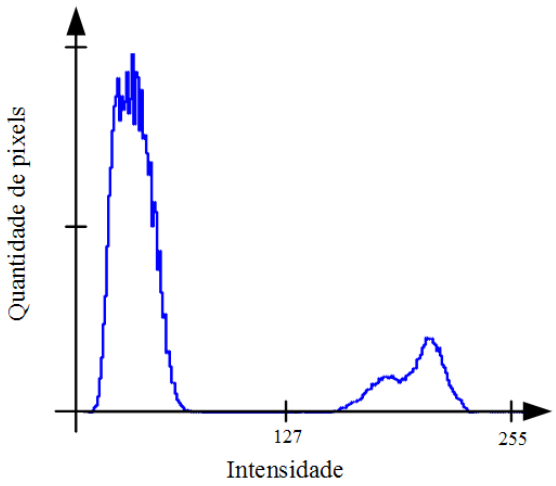


Fonte: elaborado pelo autor.

2.1.1.2 Determinação do limiar

A determinação do parâmetro de limiar λ é realizada a partir de uma análise do histograma da imagem C, isto é, a partir da distribuição de frequências das intensidades dos seus pixeis. Sendo o ruído pequeno, a tendência dos pixeis de do plano de fundo terem intensidades pequenas; e sendo as cores do objeto do primeiro plano bem distintas das cores do plano de fundo, a tendência é que a intensidade de seus pixeis sejam grandes. Desse modo, no histograma de C se formarão dois grandes picos, um para intensidades pequenas, e outro para intensidades grandes, com um grande vale no meio – como é ilustrado na Figura 2.3. Um valor de limiar adequado para a segmentação encontra-se nesse vale.

Figura 2.3: Histograma da imagem em escala cinza



Fonte: elaborado pelo autor.

2.1.1.3 Pré-processamento: aplicação do filtro da mediana

Em geral, ao aplicar-se o procedimento de segmentação descrito, o resultado obtido é perfeito. Contudo, em algumas raras ocasiões pôde-se observar que alguns pixeis isolados estavam do lado errado do limiar calculado, ou seja, havia algum ruído extra em certos pontos das imagens. Para se minimizar esse tipo de problema, é comum aplicar-se às imagens, como etapa de pré-processamento, um filtro da mediana, que consegue remover ruídos do tipo s*alt-and-pepper* ao mesmo tempo que preserva os contornos originais das imagens. Dada uma região retangular em uma imagem de tamanha m por n entradas, o filtro da mediana simplesmente retorna o valor intermediário que divide o conjunto de entradas em dois conjuntos de igual tamanho. O algoritmo do filtro utilizado – de tamanho 3 por 3 – é descrito no apêndice do trabalho.

2.1.1.4 Remoção do antebraço

Após obtida a imagem binária B com o fundo segmentado, precisaremos separar a região do antebraço da região da mão. A solução utilizada, específica para as imagens dos vídeos coletados, procede do seguinte modo:

1. Encontra-se a linha do antebraço, definindo-a pelo centroide da imagem, cuja definição é dada por:

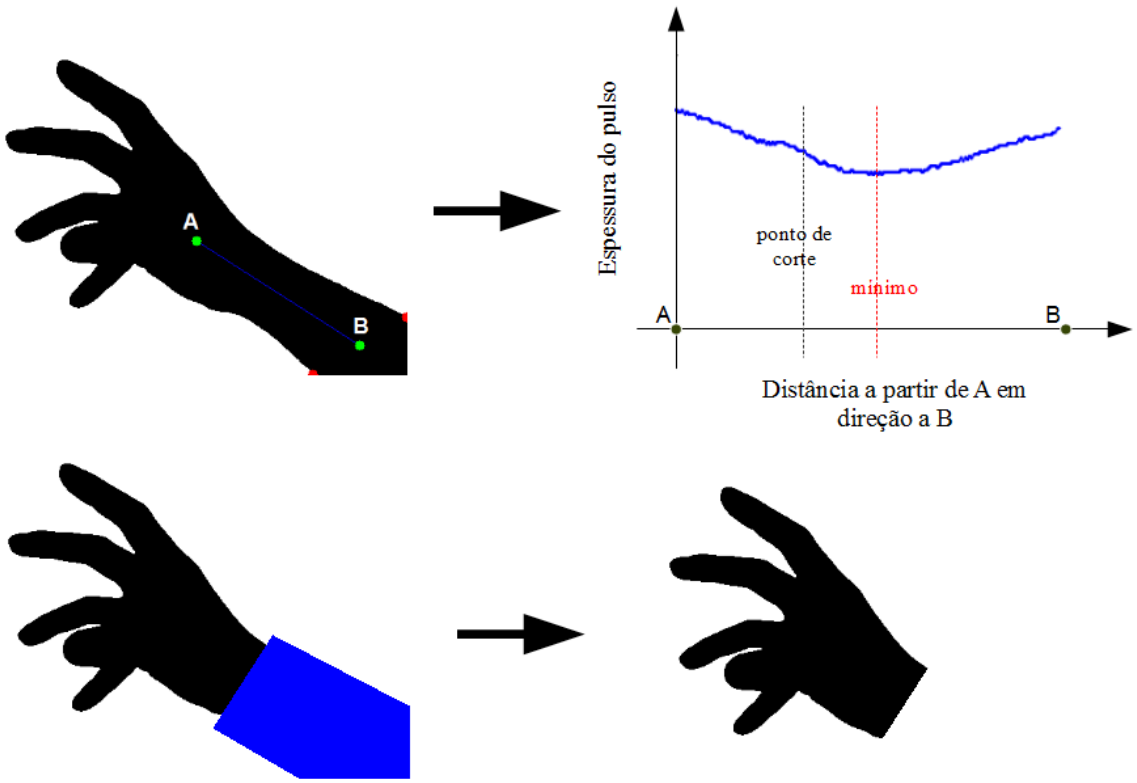
$$\vec{P}_C = \frac{\sum_{\vec{p}_i \in B} B(\vec{p}_i)\,\vec{p}_i}{\sum_{\vec{p}_i \in B} B(\vec{p}_i)}$$

e pelo ponto médio dos dois pontos da borda do antebraço que interceptam os limites da imagem.

2. Percorre-se a linha do antebraço, encontrando-se o ponto de menor espessura.
3. Define-se a linha de corte do pulso – ortogonal à linha do antebraço – pelo ponto mais próximo do centroide com espessura menor do que 1.1 vezes a menor espessura.
4. Remove-se o antebraço a partir da linha de corte do pulso e dos dois pontos da borda pintando-se o polígono correspondente.

O procedimento é ilustrado pela Figura 2.4 e descrito pelo pseudocódigo “Algoritmo de remoção do antebraço”, encontrado no anexo.

Figura 2.4: Remoção do Antebraço



A = centroide, B = ponto médio das bordas.

Fonte: elaborada pelo autor.

2.2 Extração de atributos

De modo geral, quando os dados de entrada para algum algoritmo de visão computacional são grandes demais para serem processados – devido a limitações de tempo ou de espaço – e supõe-se que sejam demasiadamente redundantes, pode vir a ser útil transformar cada entrada para uma representação reduzida, com um conjunto mais limitado de propriedades. Tal conjunto é conhecido como de vetor de atributos e o processo de obtê-lo chama-se extração de atributos.

Além de poder reduzir o tamanho da representação, a extração de atributos também pode ser utilizada para extrair informações mais significativas para o processamento. Por exemplo, se um dado reconhecedor de padrões recebe como entradas uma lista de posições (x, y) no plano cartesiano, porém as classes a serem reconhecidas estão relacionadas ao ângulo que cada posição faz com o eixo das abscissas, provavelmente será mais eficiente fornecer ao algoritmo o valor do ângulo diretamente, em vez de esperar que ele descubra como calcular $\arctan(\frac{y}{x})$.

No caso de um reconhecedor de gestos a partir de imagens de vídeo, como no presente trabalho, seria interessante que os dados de entrada do sistema já carregassem em si certas propriedades importantes, a saber: se um gesto for realizado perto ou longe da câmera, se vier a aparecer acima ou abaixo, à direita ou à esquerda, ou então se estiver rotacionado na imagem, seria bom que ele viesse igualmente a gerar os mesmos dados de entrada para cada caso. Os momentos invariantes de Hu satisfazem essas propriedades e são frequentemente utilizados, com bons resultados, em publicações a respeito de reconhecimento de padrões em imagens.

2.2.1 Momentos de imagem

Momentos de imagem podem ser utilizados para condensarem informações relativas a área, posição, orientação, e outros parâmetros relevantes de figuras contidas em imagens digitais [10]. A definição base para os momentos de ordem (i + j) para uma imagem C(x, y), em escala cinza, é dada pela equação:

$$M_{ij} = \sum_{(x,y) \in C} x^i y^j C(x, y)$$

Os momentos de ordem zero e ordem um podem ser utilizados para o cálculo do

centroide da imagem, definido por:

$$\langle \bar{x}, \bar{y} \rangle = \langle \frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}} \rangle$$

De posse do centroide, define-se momentos que são invariantes à translação, chamados de momentos centrais. Eles são definidos por:

$$\mu_{ij} = \sum_{(x,y) \in C} (x-\bar{x})^i (y-\bar{y})^j C(x,y)$$

Tais momentos podem ainda ser tornados invariantes à escala dividindo-os por um fator de escala adequado, conforme a seguinte fórmula:

$$\eta_{ij} = \frac{\mu_{ij}}{\mu_{00}^{(1+\frac{i+j}{2})}}$$

O próximo passo é obter-se momentos invariantes à rotação. Tal passo foi dado em 1962 por Ming-Kuei Hu, da Universidade de Siracusa, que propôs uma combinação dos momentos invariantes à translação e escala de modo a criar um conjunto de momentos que também fossem invariantes à rotação. Tal conjunto compreende sete momentos, chamados de momentos invariantes de Hu, e são definidos pelas equações:

$$H_1 = \eta_{20} + \eta_{02}$$

$$H_2 = (\eta_{20}-\eta_{02})^2 + 4\eta_{11}^2$$

$$H_3 = (\eta_{30}-3\eta_{12})^2 + (3\eta_{21}-\eta_{03})^2$$

$$H_4 = (\eta_{30}+\eta_{12})^2 + (\eta_{21}+\eta_{03})^2$$

$$H_5 = (\eta_{30}-3\eta_{12})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2 - 3(\eta_{21}+\eta_{03})^2] + (3\eta_{21}-\eta_{03})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2 - (\eta_{21}+\eta_{03})^2]$$

$$H_6 = (\eta_{20}-\eta_{02})[(\eta_{30}+\eta_{12})^2 - (\eta_{21}+\eta_{03})^2] + 4\eta_{11}(\eta_{30}+\eta_{12})(\eta_{21}+\eta_{03})$$

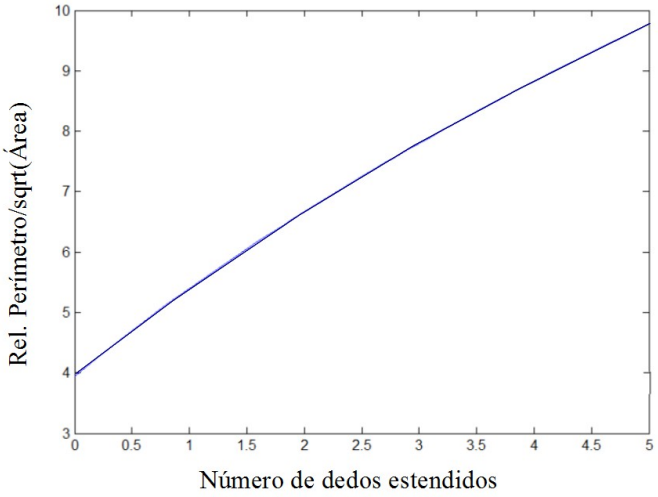
$$H_7 = (3\eta_{21}-\eta_{03})(\eta_{30}+\eta_{12})[(\eta_{30}+\eta_{12})^2 - 3(\eta_{21}+\eta_{03})^2] - (\eta_{30}-3\eta_{12})(\eta_{21}+\eta_{03})[3(\eta_{30}+\eta_{12})^2 - (\eta_{21}+\eta_{03})^2]$$

2.2.2 Relação entre perímetro e raiz da área

Um outro atributo, além dos momentos invariantes de Hu, que também é invariante à escala, translação e rotação, é a relação entre o perímetro e a raiz da área extraídos da figura na imagem. Essa relação é bastante útil particularmente no contexto de reconhecimento de gestos de mão. Tal utilidade, conforme é demonstrada pelo gráfico da Figura 2.5, vem do fato

de a relação ser praticamente linear em função do número de dedos estendidos na realização do gesto.

Figura 2.5: Interpolação da relação perímetro e raiz da área para gestos



Fonte: elaborada pelo autor.

Além disso, o cálculo do atributo é simples. Para uma certa figura em uma dada imagem binária B, a área será simplesmente dada pela contagem dos pixeis com valor 1, e o perímetro poderá ser aproximado pelo somatório da contribuição de cada pixel individual, sendo considerado o número de pixeis vizinhos pertencentes à figura. O pseudocódigo para o cômputo do perímetro aproximado é descrito no apêndice "Algoritmo Perímetro Aproximado".

2.3 Redes Neurais Artificiais – MLPs

Assim como a sua análoga em sistemas nervosos biológicos, as redes neurais artificias são capazes de aprender, podendo ser treinadas para encontrarem soluções de problemas, reconhecer padrões, classificar dados e fazer previsões de eventos futuros. O comportamento que apresentarão é determinado pelo número de unidades computacionais que possuem, pelas interconexões entre essas unidades, e pelos pesos das interconexões. Estes são ajustados automaticamente por algum algoritmo de aprendizado até que a rede apresente um desempenho satisfatório na tarefa desejada.

As redes neurais analisadas como reconhecedores de gestos são os perceptrons multicamadas (MLPs, do inglês *multilayer perceptron*). Essa é uma importante classe de redes

neurais que tem sido utilizada com sucesso para a solução de problemas difíceis e diversos, sendo treinadas de modo supervisionado pelo popular algoritmo de *retropropagação do erro* [11]. Além disso, a escolha de analisar-se esse tipo particular de RNA é devido as MPLs serem aproximadores universais de funções, como provado pelo teorema de Chybenko, sendo para isso necessário apenas uma camada oculta [12].

A apresentação a seguir do modelo das MLPs e do algoritmo de treinamento orientou-se pela nomenclatura e notação utilizadas por Haykin em seu livro *Neural Networks* [11].

2.3.1 Neurônios

Neurônios são as unidades computacionais das RNAs. O modelo de cada um deles, ilustrado na Figura 2.6 , é constituído por três elementos:

a. Um conjunto de *sinapses*, ou *interconexões*, caracterizadas pelos seus pesos. Uma sinapse de um neurônio *j* simplesmente multiplica um sinal de entrada x_i por um peso sináptico w_{ji} . Se esse peso for positivo, a sinapse será excitatória; e ser for negativo, ela será inibitória. O peso sináptico de índice zero, w_{j0} , representa um valor de *bias*, possuindo sempre uma entrada x_0 fixa. Em geral, $x_0 = +1$ ou $x_0 = -1$.

b. Um *somador*, responsável por agregar o sinal de cada sinapse e gerar o *potencial de ativação* do neurônio. Para um neurônio *j*, com *p* entradas, o seu potencial de ativação v_j será dado por:

$$v_j = \sum_{i=0}^{p} w_{ji} x_i$$

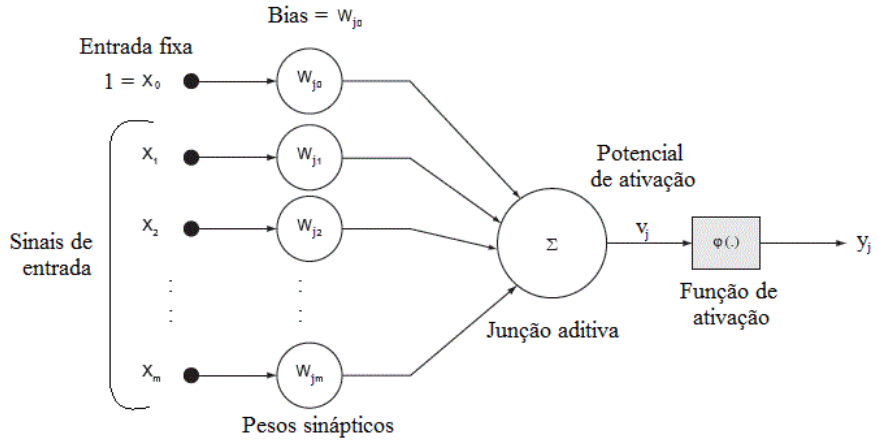
c. Uma *função de ativação* φ , responsável por limitar a amplitude da resposta y_j do neurônio, isto é, $y_j = \varphi(v_j)$. Comumente, escolhe-se φ de modo que y_j fique limitada aos intervalos $[0,1]$ ou $[-1,1]$. As funções de ativação mais utilizadas no projeto de redes neurais são as *sigmoides*, que são definidas por serem limitadas, estritamente crescentes e diferenciáveis. Dois exemplos típicos de sigmoides são a função logística, dada por

$$f(x) = \frac{1}{1 + e^{-x}}$$

e a tangente hiperbólica:

$$\tanh(x) = \frac{2}{1 + e^{-2x}} - 1$$

Figura 2.6: Modelo de Neurônio



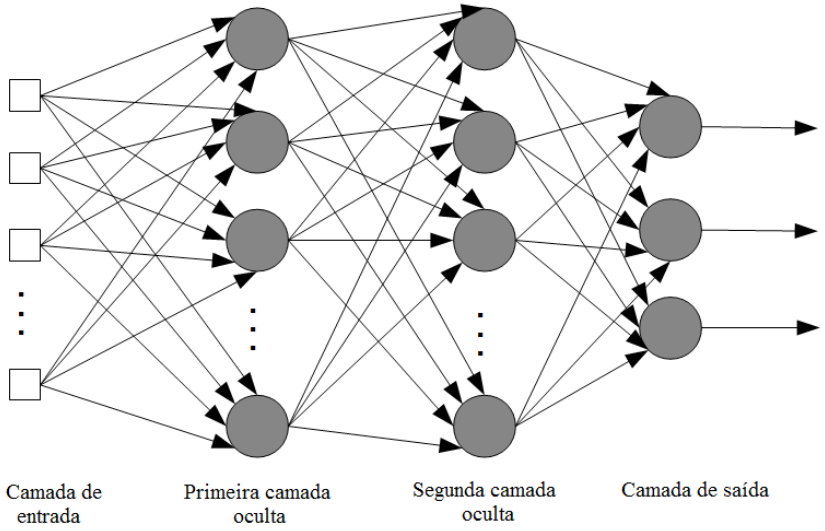
Fonte: http://www.codeproject.com/Articles/175777/Financial-predictor-via-neural-network

2.3.2 Camadas

A arquitetura das MLPs é organizada em camadas, como ilustra a Figura 2.7, sendo que cada uma delas é completamente conectada à seguinte, isto é, cada nodo possui interconexões com todos os outros nodos nas camadas adjacentes. Há três tipos de camadas:

1. A *camada de entrada*, em que são apresentados os *padrões de ativação* (vetor de entrada), que constituirão os sinais aplicados aos neurônios da primeira camada oculta.
2. Uma ou mais *camadas ocultas*, compostas por *neurônios ocultos*. A função desses neurônios é de serem intermediários entre as entradas externas e saída da rede. Ao adicionar-se mais camadas ocultas, a rede torna-se capaz de extrair medidas estatísticas de mais alta ordem dos padrões de entrada [11]. Tais medidas são caracterizadas por utilizem potências de grau três ou maiores dos dados de entrada, sendo opostas as medidas mais convencionais, que utilizam termos constantes, lineares ou quadráticos, como na média aritmética e variância.
3. A *camada de saída*, responsável por realizar a última computação da rede e fornecer as saídas.

Figura 2.7: Exemplo de arquitetura de uma rede MLP com duas camadas ocultas



Fonte: elaborada pelo autor.

2.3.3 Algoritmo de treinamento – Retropropagação do erro

O desenvolvimento do algoritmo de retropropagação do erro representou um marco significativo para a teoria de redes neurais, uma vez que ele provê um método computacionalmente eficiente para o treinamento das MPLs [11]. A função de custo que ele visa minimizar é o erro quadrático médio sobre os padrões de treinamento. Para tal, é realizado um procedimento de descida de gradiente, em que se procede por duas partes. Na primeira delas, a MLP é alimentada com um vetor de entrada e as respostas dos neurônios são computadas, seguindo-se camada por camada até a saída. Ao final, calcula-se o erro da saída gerada com relação a saída desejada. Na segunda parte, o erro é retropropagado e computa-se para cada neurônio o seu gradiente local, que servirá para se atualizar os pesos sinápticos.

A derivação do algoritmo pode ser encontrada em [11], de modo que apenas será feita a sua apresentação. Antes de se prosseguir, são definidas algumas notações:

- n_l : número de neurônios na camada l ou, se $l = 0$, número de entradas da rede;
- $w_j^{(l)} = [w_{j0}^{(l)} \; w_{j1}^{(l)} \; w_{j2}^{(l)} \; ... \; w_{j\,n_l}^{(l)}]^T$: vetor de pesos sinápticos do neurônio j da camada l, em que $w_{j0}^{(l)}$ representa o seu *bias*.
- $v^{(l)} = [v_1^{(l)} \; v_2^{(l)} \; v_3^{(l)} \; ... \; v_{n_l}^{(l)}]^T$: vetor de potenciais de ativação dos neurônios da camada l, isto é:

$$v_j^{(l)} = \sum_{i=0}^{n_{(l-1)}} w_{ji}^{(l)} y_i^{(l-1)}$$

- $y^{(l)} = [y_1^{(l)} \; y_2^{(l)} \; y_3^{(l)} \; ... \; y_{n_l}^{(l)}]^T$: vetor de respostas dos neurônios da camada *l*:

$$y_j^{(l)} = \varphi(v_j^{(l)})$$

- *L* é definido como sendo a camada de saída, e *0* como sendo a camada de entrada. Portanto, o vetor de resposta da rede é dado por $y^{(L)}$ e o vetor de entrada é fornecido por $y^{(0)}$.

Seja $S = \{[x(k), d(k)] \mid k = 1...N\}$ o conjunto de dados de treinamento, em que $x(k) = [x_1(k) \; x_2(k) \; x_3(k) \; ... \; x_{n_0}(k)]^T$ é o k-ésimo vetor do padrão de entrada, e $d(k) = [d_1(k) \; d_2(k) \; d_3(k) \; ... \; d_{n_L}(k)]^T$ a saída desejada para tal padrão. O algoritmo procede pelos seguintes passos:

1. *Inicialização*. Inicialize todos os pesos sinápticos para algum valor aleatório pequeno, uniformemente distribuídos.
2. *Apresentação dos padrões de treinamento*. Defina uma ordenação aleatória para o conjunto de treinamento *S*. Para cada par $(x(k), d(k))$ da ordem resultante, prossiga pelos passos 3 e 4.
3. *Propagação das entradas*. Defina-se $w_j^{(l)}(k)$, $v_j^{(l)}(k)$, e $y_j^{(l)}(k)$ como sendo, para o neurônio *j,* respectivamente: os pesos sinápticos, o potencial de ativação e a resposta, quando da apresentação do padrão (x_k, d_k) . Faça $y_j^{(0)}(k) = x_k$ e atualize até a saída da rede, camada por camada, as respostas dos neurônios. O erro para o neurônio *j* da camada de saída, com relação a resposta desejada é definido por:

$$e(k) = [e_1(k) \; e_2(k) \; e_3(k) \; ... \; e_{n_L}(k)]^T$$

$$e_j(k) = d_j(k) - y_j^{(L)}(k)$$

4. *Retropropagação do erro*. A partir da camada de saída, retorne, camada por camada, calculando-se o *gradiente local* $\delta_j^{(l)}(k)$ de cada neurônio *j*, que é definido por:

$$\delta_j^{(l)}(k) = \begin{cases} \varphi'(y_j^{(l)}(k)) e_j(k), & se \; l = L \\ \varphi'(y_j^{(l)}(k)) \sum_{m=1}^{n_{(l+1)}} \delta_m^{(l+1)}(k) w_{mj}^{(l+1)}(k), & se \; l \neq L \end{cases}$$

onde φ' é a derivada primeira da função de ativação. A atualização dos pesos sinápticos se dará pela pela descida de gradiente, através da *regra delta generalizada,*

sendo η a taxa de aprendizado e α o valor do momento:

$$w_{ji}^{(l)}(k+1) = w_{ji}^{(l)}(k) + \Delta w_{ji}^{(l)}(k)$$

$$\Delta w_{ji}^{(l)}(k) = \alpha[w_{ji}^{(l)}(k) - w_{ji}^{(l)}(k-1)] + \eta \delta_j^{(l)}(k) y_i^{(l-1)}(k)$$

5. *Iteração*. Cada apresentação dos padrões de treinamento, representado pelos passos 3, 4 e 5, é definido como sendo uma *época*. O algoritmo prossegue pela apresentação de novas épocas, com ajuste da taxa de aprendizado e momento, até que um critério de parada satisfatório seja alcançado. Um desses critérios é quando o erro quadrático médio computado sobre todos os padrões de treinamento tenha atingido um valor pequeno aceitável, isto é:

$$\varepsilon_{médio} = \frac{1}{N} \sum_{k=1}^{N} |e(k)|^2 \leq \varepsilon_{crítico}$$

2.3.4 Validação cruzada

O problema em tomar-se como critério de parada apenas a minimização do erro quadrático médio é que a MLP assim treinada pode acabar se ajustado "bem demais" aos padrões de treinamento, ocorrendo o que é chamado de *sobreajuste*. Nesse caso, se forem apresentados padrões de entrada distintos dos que foram utilizados para treinamento, a rede terá um desempenho pobre.

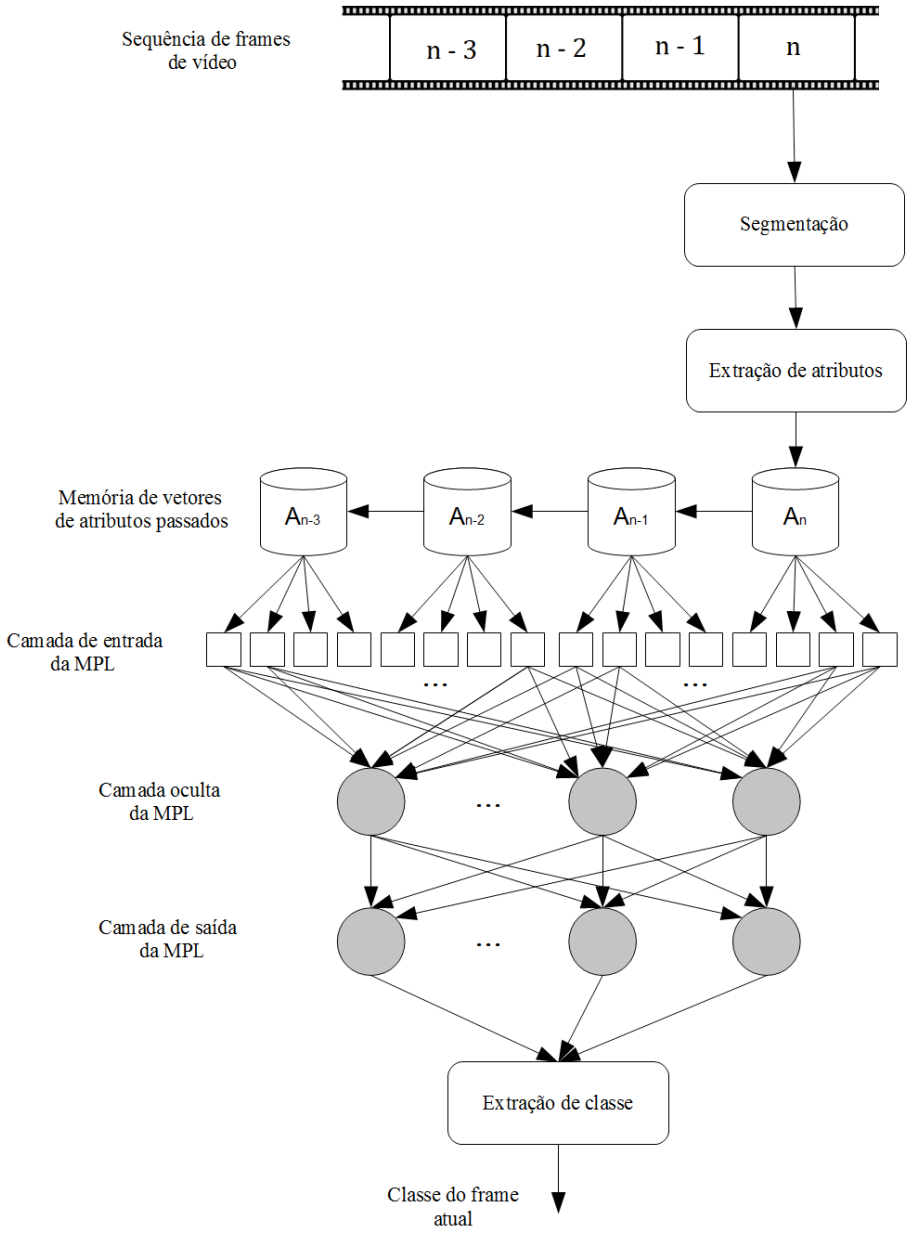
Uma técnica para contornar-se esse problema e permitir que a rede tenha uma boa generalização para novos padrões de entrada é fazer o uso de validação cruzada. Com essa técnica, os dados disponíveis são particionados em três conjuntos: treinamento, validação e teste.

O conjunto de treinamento, apresentado durante o treinamento da rede, é utilizado para o ajuste dos pesos sinápticos de acordo com o erro obtido. Já o conjunto de validação, também utilizado durante o treinamento, serve para avaliar-se o grau de generalização da rede e, então, quando ele deixar de melhorar, parar o aprendizado. Por último, o conjunto de teste não tem efeito sobre o treinamento e provê uma medida independente de desempenho.

3 ARQUITETURA DO SISTEMA DE RECONHECIMENTO DE GESTOS

Na Figura 3.1 apresenta-se a arquitetura global do sistema de reconhecimentos de gestos. Sendo um sistema em tempo real, sua única entrada é o frame atual capturado por alguma câmera de vídeo. Esse frame é segmentado e da imagem resultante extrai-se o vetor de atributos, que é armazenado em memória conforme o número de frames sobre os quais o classificador atua. Uma rede MLP faz o papel de classificador, sendo constituída de apenas uma única camada oculta e possuindo tantas entradas quantos forem o número de componentes de todos os vetores de atributos armazenados. Para cada classe de gestos escolhida há um neurônio correspondente na camada de saída. A resposta de cada um deles serve como estimativa da probabilidade de que o frame pertença a classe de que é responsável, de modo que a saída do sistema é o rótulo de classe que corresponde a classe do neurônio com maior valor de resposta.

Figura 3.1: Arquitetura global do sistema de reconhecimento de gestos



Este é um exemplo específico das possíveis arquiteturas do sistema. Nesta, o classificador atua apenas sobre quatro frames. Fonte: elaborada pelo autor.

4 FORMAÇÃO DA BASE DE DADOS ROTULADOS

4.1 Definição das classes de gestos

O primeiro passo para a formação do conjunto de dados que servirão de base para o treinamento e a análise das redes neurais é a definição de quantas e quais categorias de gestos serão utilizadas.

Com relação ao número de classes, em artigos na área de reconhecimento de gestos é comum encontrar-se o uso de um número reduzido delas. Por exemplo, no trabalho de Chen, et al. [13], o classificador utilizado aprende quatro classes, enquanto que no trabalho de Liu, et al. [14] são utilizadas apenas três. Desse modo, definir-se seis classes de gestos pareceu razoável. Uma dessas classes teria o propósito de servir como gesto de repouso enquanto as cinco restantes serviriam para efetivamente enviar um comando ao sistema.

4.1.1 Coleta inicial de possibilidades de classes

Para auxiliar na definição de quais classes serão reconhecidas, construiu-se um banco de 49 possibilidades de gestos, como mostra a Figura 4.1. Tais possibilidades foram obtidas a partir de uma pesquisa de gestos pertencentes a alfabetos de linguagens de sinais, além de se considerar alguns gestos do cotidiano.

Figura 4.1: Conjunto de possibilidades de classes de gestos

Fonte: elaborado pelo autor.

A partir desse banco de possibilidades de classes também foi gerado um conjunto de parâmetros de normalização para o vetor de atributos extraídos das imagens (Tabela 4.1). Objetivo da normalização é tornar o treinamento das redes neurais mais eficiente e estabilizar os cálculos numéricos, já que os momentos de Hu podem assumir valores muito pequenos. A fórmula de normalização é dada por:

$$Atributo_{normalizado} = \frac{Atributo_{original} - Atributo_{médio}}{Faixa\ de\ variação}$$

Depois de sua aplicação, todos os atributos de todos os gestos considerados se encontrarão na faixa de -1 a 1.

Tabela 4.1: Faixa de valores dos atributos para o conjunto de possibilidades de classes de gestos

Atributo	Valor Médio	Faixa de variação ±
Perímetro/raiz(Área)	7.81483	2.85432
H1	0.221213	0.056705
H2	0.0188885	0.0182435
H3	0.00465068	0.00463972
H4	0.00181899	0.00181873
H5	9.01157e-006	1.07619e-005
H6	0.000215733	0.000301487
H7	-2.94759e-006	6.16213e-006

Fonte: elaborado pelo autor.

4.1.2 Estudo da distância de grupos de classes no espaço de atributos

Do conjunto de possibilidades deve-se escolher seis classes. Uma delas, o gesto de repouso (mão fechada), está fixa e é representada pela classe 1. As outras classes foram escolhidas segundo o critério de maximizar as mínimas distâncias entre quaisquer duas classes no espaço de atributos. Com isso se espera criar uma situação de aprendizado favorável para o classificador.

Com a classe de repouso fixa, o número de possibilidades de escolhas é dado pela combinação de 48 classes tomadas 5 a 5, i.e, um total de $\binom{48}{5} = 1712304$ grupos. Para cada um deles, há um total de 15 distâncias a serem consideradas – uma para cada par de classes.

Todas as possibilidades foram computadas e então ordenadas segundo as maiores menores distâncias entre as classes. Na Tabela 4.2 são mostrados os 5 melhores resultados com as maiores distâncias e na Tabela 4.3 os 5 piores com as menores distâncias. Das 15 distâncias entre as classes, apresenta-se as três menores, ficando entre colchetes as classes a que elas se referem.

Tabela 4.2: Grupos de classes com as maiores menores distâncias

Ordem	**Classes**	**Distância entre classes**			
		Menor	2ª Menor	3ª Menor	Média
1	1, 4 , 12, 21, 26, 48	1.92 {21, 26}	2.35 {1, 4}	2.42 {4, 21}	2.97
2	1, 5, 12, 21, 26 , 48	1.92 {21, 26}	2.29 {1, 5}	2.45 {5, 21}	3.00
3	1, 5, 21, 23, 26 , 48	1.92 {21, 26}	2.23 {5, 23}	2.24 {21, 23}	2.86
4	1, 4, 21, 26, 42, 48	1.92 {21, 26}	2.12 {42, 48}	2.35 {1, 4}	2.97
5	1, 5, 21, 26, 42 , 48	1.92 {21, 26}	2.12 {42, 48}	2.29 {1, 5}	3.00

Fonte: elaborado pelo autor.

Tabela 4.3: Grupos de classes com as menores menores distâncias

Ordem	**Classes**	**Distância entre classes**			
		Menor	2ª Menor	3ª Menor	Média
1712300	1, 7, 19, 35, 37, 39	0.17 {37, 39}	0.17 {1, 19}	0.31 {7, 35}	1.17
1712301	1, 19, 31, 32, 37, 39	0.17 {37, 39}	0.17 {1, 19}	0.28 {31, 32}	1.11
1712302	1, 13, 19, 25, 37, 39	0.17 {37, 39}	0.17 {1, 19}	0.28 {13 ,25}	1.26
1712303	1, 19, 30, 31, 37, 39	0.17 {37, 39}	0.17 {1, 19}	0.23 {30, 31}	1.17
1712304	1, 19, 34, 35, 37, 39	0.17 {37, 39}	0.17 {1, 19}	0.22 {34, 35}	1.12

Fonte: elaborado pelo autor.

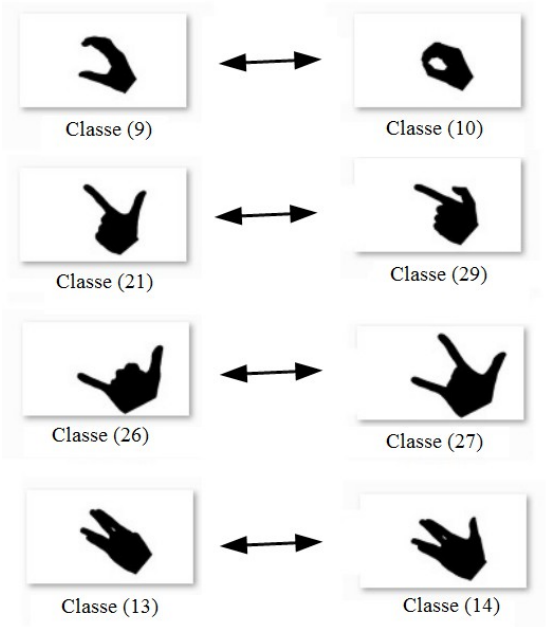
4.1.3 Classes de gestos escolhida

A escolha final das classes a serem reconhecidas foi feita com base na sua colocação na ordenação global de possibilidades, além de algumas restrições adicionais: (a) não pode haver gestos que causem algum desconforto muscular ao fazê-los, como os gestos ilustrados pela Figura 4.2; (b) não pode haver dois gestos em que um esteja no caminho da realização do outro, como ilustrado pela Figura 4.3 – caso contrário o usuário poderia acidentalmente realizar um comando indesejado; (c) os gestos devem ser comuns e fáceis de fazer.

Levando em conta essas considerações, o grupo escolhido ocupa a posição 146 e está ilustrado pela Figura 4.4 (sendo contrastado ao pior grupo possível), e a suas métricas são mostradas na Tabela 4.4.

Figura 4.2: Gestos que provocam desconforto muscular

Fonte: elaborada pelo autor.



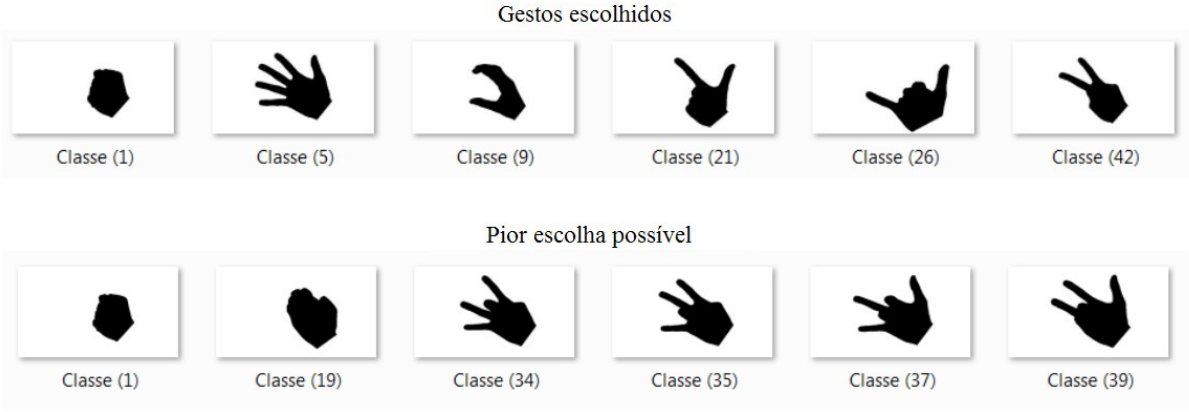
Fonte: elaborado pelo autor.

Tabela 4.4: Métricas das classes escolhidas

Ordem	**Classes**	**Distância entre classes**			
		Menor	2ª Menor	3ª Menor	Média
146	1, 5, 9, 21, 26, 42	1.76 {9, 42}	1.77 {5, 9}	1.92 {21, 26}	2.52

Fonte: elaborado pelo autor.

Figura 4.4: Conjunto de gestos escolhidos e pior escolha possível



Fonte: elaborado pelo autor.

4.2 Coleta de vídeos

A coleta de vídeos foi realizada utilizando-se uma webcam LifeCam HD-3000 da Microsoft, com taxa de 30 fps, resolução de 320 por 240 pixeis, sendo os gestos feitos a cerca de 40 cm da câmera. No total, 10 pessoas participaram da coleta, sendo gerado um vídeo para cada mão. A sequência de gestos realizada pelos participantes passa por cada transição possível de dois gestos, i.e., um total de 30 transições para o grupo de 6 classes.

Para auxiliar na execução da sequência e garantir que todas as transições fossem feitas, foi programado um apresentador de slides simples o qual mostra a intervalos regulares uma representação do gesto a ser feito assim como um mnemônico para ele. A Figura 4.5 ilustra os slides utilizados, em miniatura. O menor período de transição entre slides conseguido de modo que os participantes pudessem acompanhar a apresentação foi de cerca de 1.3 segundos.

Figura 4.5: Slides com ilustração e mnemônicos dos gestos

Fonte: elaborado pelo autor

Como plano de fundo de filmagem foi utilizado um tecido de cor azul. Essa é uma das cores preferidas para a gravação de previsões de tempo em telejornais e de efeitos especiais no cinema, uma vez que ela é a cor de maior contraste com a pele humana, permitindo, assim, que a segmentação entre primeiro e segundo plano seja mais fácil. Para se verificar esse fato, tomou-se como amostra uma região de textura de pele e calculou-se a sua cor média no espaço de cores RGB – Figura 4.6. Nesse caso, a cor mais contrastante foi realmente a azul, porém a cor verde ou preta certamente poderiam ser utilizadas.

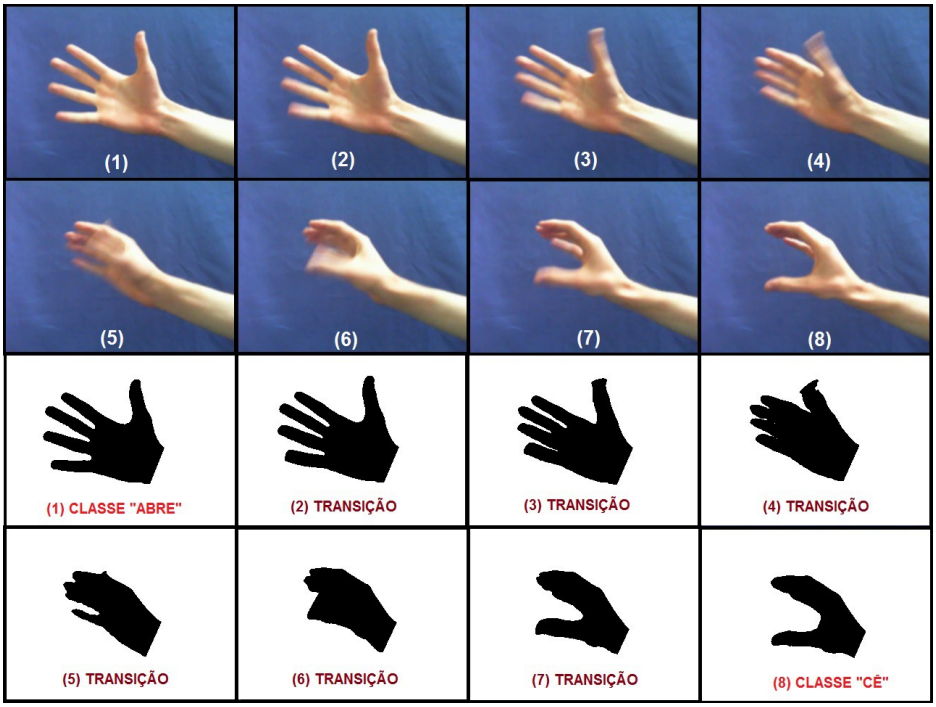
Figura 4.6: Exemplo de textura de pele, a sua cor média, e a cor de maior contraste

Valor RGB médio = (198, 151, 127). Cor RGB de maior contraste = (0, 0, 255). Fonte: http://www.photoshoptextures.com/human-textures/hand-texture.jpg

4.3 Rotulação dos vídeos coletados

Após coletados, os vídeos foram manualmente rotulados, gerando-se um arquivo contendo o padrão de referência para cada frame. Cada um deles recebeu um índice segundo a classe que representa ou então segundo a pertinência a uma região de transição. Tais regiões foram definidas como sendo o intervalo do primeiro frame em que há sinal de movimento para a troca de gesto – o que pôde, em geral, ser identificado pelo aparecimento de um borrão na imagem – até o último frame em que ainda há resquício de movimento. A Figura 4.7 ilustra uma região de transição, com as imagens borradas características.

Figura 4.7: Sequência de frames de uma região de transição de gestos



Fonte: elaborado pelo autor.

Em média, as regiões de transição tiveram duração de 270 ms, ocupando um total de 8 frames. Esse parâmetro serviu como base para a limitação do número de frames sob os quais as redes neurais viriam a ser analisadas. Os dados para as outras regiões podem ser encontrados Tabela 4.5.

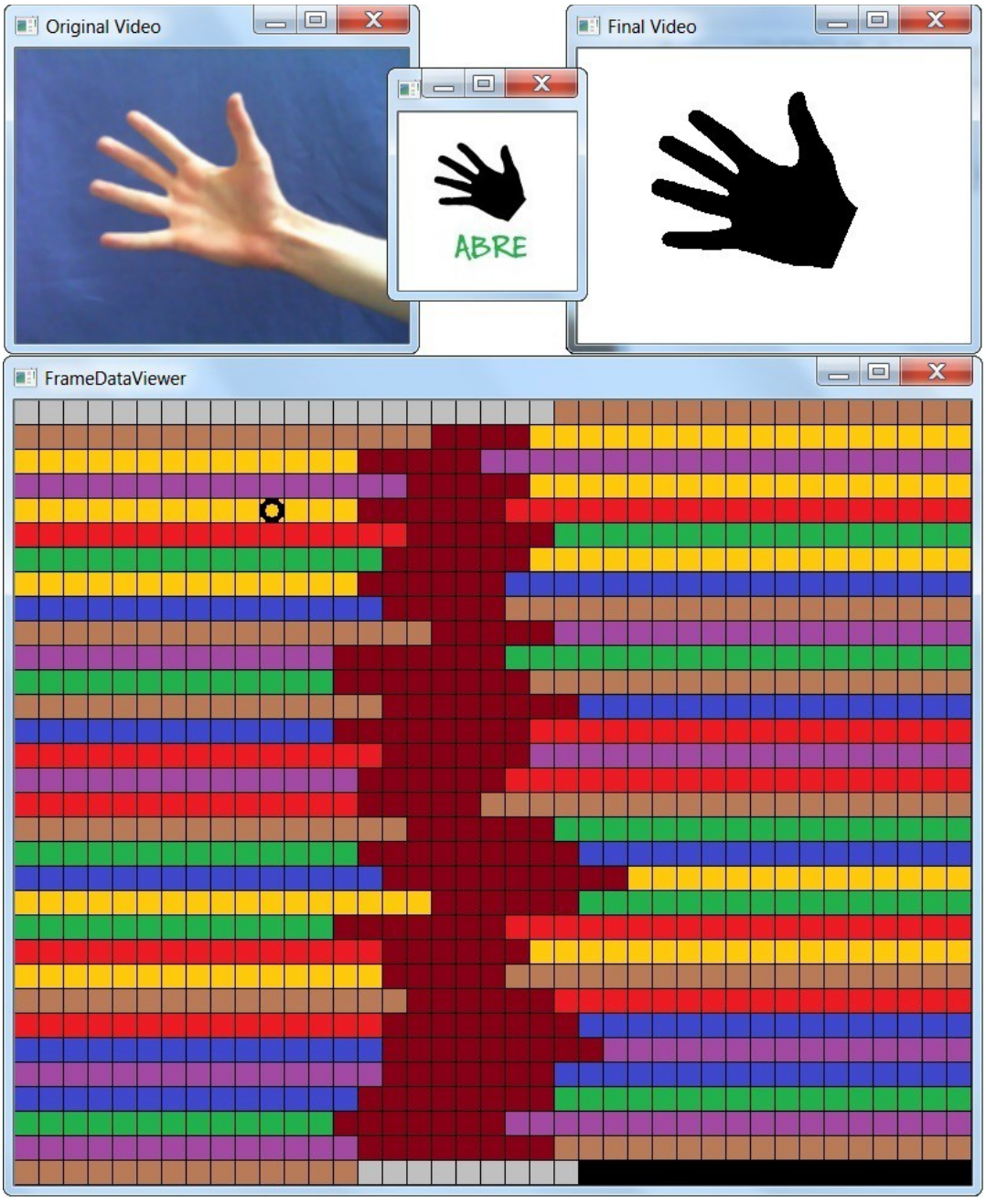
Tabela 4.5: Distribuição dos frames por classe de gesto

Classe	**Porcentagem do total de frames**	**Comprimento médio dos segmentos (frames)**	**Duração média dos segmentos (ms)**
“fecha”	16.17	37.80	1260
“atira”	13.38	31.28	1040
“dois”	12.33	29.68	990
“surf”	12.40	29	960
“cê”	12.22	29.41	980
“abre”	12.94	30.26	1000
“transição”	20.56	8.13	270

Fonte: elaborado pelo autor.

Com o intuito de agilizar o demorado processo de rotulação de milhares de frames e evitar possíveis erros na geração do arquivo de referência, criou-se um programa de auxílio a rotulagem. A sua interface é ilustrada na Figura 4.8. Nela são apresentadas o frame original, o frame segmentado, uma imagem pequena indicando a rotulação do frame atual, e uma matriz bidimensional na qual é mostrada a posição do frame sendo mostrado e a rotulação de todos os frames do vídeo usando-se um código de cores. O programa permite rodar o vídeo em sequência normal ou de trás para frente, fazer pausas, navegar frame por frame horizontal ou verticalmente, fazer seleções de trechos e, por fim, rotulá-los segundo a classe escolhida por meio de comandos do teclado.

Figura 4.8: Interface do programa de auxílio a rotulação



Fonte: elaborado pelo autor.

5 RESULTADOS

5.1 Métricas de avaliação das redes neurais

5.1.1 Erros de identificação de segmentos

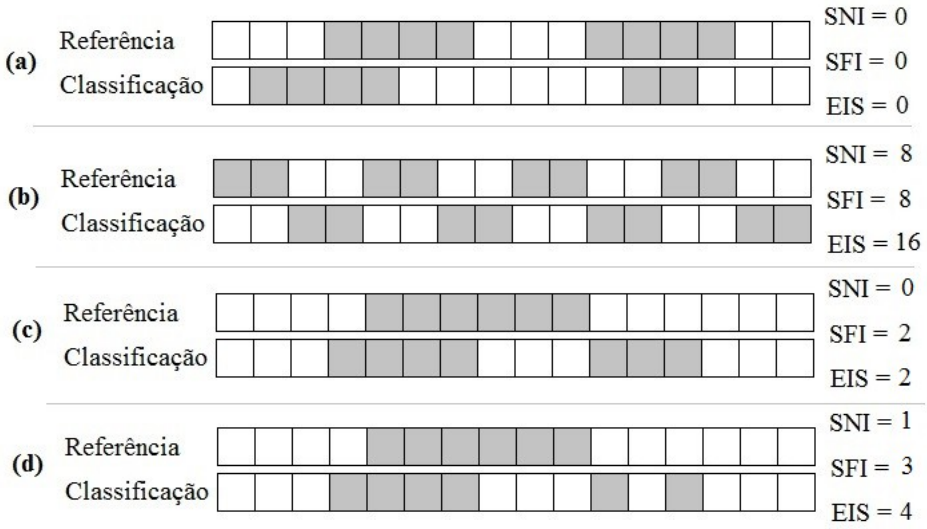
Para que um classificador de gestos seja útil na prática, não é necessário que ele seja capaz de identificar de modo exato, segundo o padrão de referência, a classe de cada um dos frames do vídeo. Contudo, é essencial que, se um gesto for realizado, em algum momento ele venha a ser reconhecido – não importando se a classificação correta se inicie precisamente no primeiro frame correspondente a classe do gesto.

Dito de outro, o classificador será útil se a classificação que tenha realizado puder ser transformada, a partir de um breve deslocamento de bordas dos segmentos, no padrão de referência. Para avaliar essa propriedade do sistema, criou-se duas medidas: SNI (segmentos não identificados), sendo correlacionada ao número de segmentos presentes na referência, mas não identificados na classificação; e SFI (segmento falsamente identificados), sendo correlacionada ao número de segmentos identificados na classificação, porém ausentes na referência. Ambas as medidas somadas correspondem ao EIS, isto é, erros de identificação de segmentos:

$$EIS = SNI + SFI$$

O cálculo de SNI e SFI é realizado pela contagem do casamento entre segmentos da referência e da classificação. É dito que dois segmentos *casam* quando são sobrepostos e possuem o mesmo rótulo. Para o cálculo de SNI, percorre-se cada segmento da referência e verifica-se se há um único segmento na classificação que case com ele. Se não houver tal segmento, o erro é incrementado. Já o cálculo de SFI procede de modo inverso, isto é, percorre-se os seguimentos da classificação e verifica-se se há o casamento com um segmento da referência. Na Figura 5.1 são dados alguns valores de como essas medidas se comportam para alguns exemplos de padrão de referência e de classificação, considerando-se duas classes de segmentos: “branco” e “cinza”.

Figura 5.1: Exemplo de aplicação da métrica EIS



Fonte: elaborado pelo autor.

5.1.2 Matriz de confusão

A matriz de confusão é uma medida complementar a EIS que ajuda a avaliar o sistema frame por frame e classe por classe, permitindo a localização da ocorrência dos erros. Nela são apresentadas, em cada célula, o número de predições realizadas para cada classe *versus* a classificação que deveria ter sido feita, conforme o padrão de referência. O número de acertos se encontram na diagonal principal, enquanto que as demais posições representam os erros na classificação. Obtida a matriz, pode-se ainda calcular outras medidas para a avaliação do classificador, como acurácia, erro, precisão e sensibilidade. Tais medidas são definidas a seguir.

Seja N o número total de classes e $n(x, y)$ o número de predições realizadas para a classe "*y"* quando a classe de referência for "*x"*. A *precisão* do classificador para uma classe C_i é a razão entre o total de predições corretas e o número total de predições realizadas para essa classe, isto é:

$$p(C_i) = \frac{n(C_i, C_i)}{\sum_{k=1}^{N} n(C_k, C_i)}$$

Uma precisão de 100% para uma dada classe significa que o classificador nunca cometerá erros falso positivo para ela, ou seja, se um frame não pertence a classe então nunca será classificado como sendo pertencente. Já a *sensibilidade* é uma medida complementar a precisão, relacionando-se a ocorrência de erros falso negativo. É calculada pela razão entre o

total de predições corretas para a classe e o número total de referências pertencentes a ela:

$$s(C_i) = \frac{n(C_i, C_i)}{\sum_{k=1}^{N} n(C_i, C_k)}$$

Uma avaliação do classificador como um todo é dado pela *acurácia* e pelo *erro*. A primeira é computada pela razão entre o total de predições corretas e o total de predições realizadas. Já o erro é a sua medida complementar, isto é:

$$Acurácia = \frac{\sum_{k=1}^{N} n(C_k, C_k)}{\sum_{i=1}^{N} \sum_{j=1}^{N} n(C_i, C_j)}$$

$$Erro = 1 - Acurácia$$

Na Tabela 5.1 ilustra-se como serão dispostos a matriz de confusão e as medidas adicionais para os experimentos realizados, utilizando-se valores relativos para o número de predições por cada classe, isto é:

$$n_R(C_x, C_y) = \frac{n(C_x, C_y)}{\sum_{i=1}^{N} \sum_{j=1}^{N} n(C_i, C_j)}$$

Tabela 5.1: Estrutura da matriz confusão com medidas de sensibilidade precisão e acurácia, para um classificador de 4 classes

		Classe Predita				**Sensibilidade**
		C_1	**C_2**	**C_3**	**C_4**	
Classe de Ref.	**C_1**	$n_R(C_1, C_1)$	$n_R(C_1, C_2)$	$n_R(C_1, C_3)$	$n_R(C_1, C_4)$	$s(C_1)$
	C_2	$n_R(C_2, C_1)$	$n_R(C_2, C_2)$	$n_R(C_2, C_3)$	$n_R(C_2, C_4)$	$s(C_2)$
	C_3	$n_R(C_3, C_1)$	$n_R(C_3, C_2)$	$n_R(C_3, C_3)$	$n_R(C_3, C_4)$	$s(C_3)$
	C_4	$n_R(C_4, C_1)$	$n_R(C_4, C_2)$	$n_R(C_4, C_3)$	$n_R(C_4, C_4)$	$s(C_4)$
Precisão		$p(C_1)$	$p(C_2)$	$p(C_3)$	$p(C_4)$	*Acurácia*

Fonte: elaborado pelo autor.

5.2 Experimentos

Para a realização dos experimentos dividiu-se o conjunto de dados em duas partes de igual tamanho, cada uma com 8184 frames contínuos, correspondentes a 7 sequências de vídeos de diferentes pessoas. A primeira parte foi utilizada para treinamento e teste das redes neurais, sendo 70% dos dados distribuídos aleatoriamente para treinamento, e o restante para teste. A segunda parte foi reservada para validação.

A metodologia usual de distribuir todos os dados aleatoriamente, de uma só vez, nos conjuntos de treinamento, teste e validação não pôde ser utilizada, uma vez que a medida EIS requer que todos os frames estejam em sequência para ser computada. Tem-se, assim, a razão da divisão dos dados em duas partes contínuas.

Foram realizados 4 experimentos, sendo que cada um deles foi baseado nos anteriores. Nos três primeiros, o objetivo foi encontrar a rede neural com a melhor adaptação no conjunto de validação. A rede escolhida é sempre aquela que possui, primariamente, o menor EIS, e, em seguida, a maior acurácia. Com o intuito de evitar-se escolher uma rede adaptada a um mínimo local, cada configuração foi treinada múltiplas vezes, iniciando-se com pesos de conexões aleatórios. No quarto experimento procura-se encontrar uma razão para o número de neurônios na camada oculta das redes obtidas.

5.2.1 Experimento 1

No primeiro experimento os dados foram rotulados seguindo-se uma filosofia de gesto nulo e gesto de comando. Na classe *nula* inclui-se a classe de gestos “fecha” - utilizada como posição de repouso – e a as regiões de transição. O conjunto de gestos de comando foi formado pelas classes restantes, i.e., “atira”, “dois”, “surf”, “cê”, e “abre”.

Em busca da rede MLP mais adaptada, examinou-se um subconjunto extensivo do espaço de possíveis configuração: de 1 a 8 frames de entrada *versus* de 1 a 50 neurônios na camada oculta. O limite superior de oito entradas foi estabelecido por ser o tamanho médio das regiões de transição, enquanto que o limite de 50 neurônios na camada oculta foi determinado a partir de uma análise esparsa no espaço de configurações, em que se percebeu que redes com mais de 50 neurônios ocultos não geravam melhores resultados do que redes menores.

Cada configuração foi treinada 120 vezes, e o experimento durou 47 horas e 19 minutos. A rede mais bem adaptada, com 34 neurônios na camada oculta, atuou sobre 5

frames de entrada. Obteve um erro SNI de 2, SFI de 8, e, por conseguinte, EIS de 10. A sua matriz de confusão, com medidas adicionais é mostrada na Tabela 5.2.

Tabela 5.2: Matriz de confusão da melhor rede neural do experimento 1

		Classe Predita						**Sens.**
		“nulo”	**“atira”**	**“dois”**	**“surf”**	**“cê”**	**“abre”**	
Classe de Ref.	**“nulo”**	33,40	0,84	0,49	0,83	0,32	0,84	90,95
	“atira”	0,15	13,23	0	0	0	0	98,90
	“dois”	0,18	0	12,15	0	0	0	98,51
	“surf”	0,09	0	0	12,32	0	0	99,31
	“cê”	0,26	0	0	0	11,96	0	97,90
	“abre”	0,07	0	0	0	0	12,87	99,43
Precisão		97,81	94,01	96,13	93,68	97,41	93,85	95,93

Fonte: elaborado pelo autor.

Com uma acurácia de 95,93%, a rede apresentou o melhor resultado possível com relação ao conjunto de classes de comando, não havendo confusão alguma entre elas. Isso certamente é devido a escolha das classes segundo a avaliação das maiores menores distâncias no espaço de atributos.

É interessante notar que a atuação da melhor rede foi em 5 frames. Em parte, isso pode ser explicado pois existe uma interdependência entre os dados. Embora nem todos os dados anteriores ao frame atual possuam alguma informação preditiva sobre a sua classe - tome-se como exemplo um gesto realizado há mais de 10 minutos -, alguns frames próximos a ele certamente possuem tal informação, de modo que o seu conhecimento melhora a possibilidade de uma correta classificação.

Outro ponto de justificativa para a rede atuar em mais de um frame vem do fato de que toda rede MLP que atue em $N + 1$ frames tenha poder computacional igual ou superior a uma rede que atue em N frames. Desse modo, a determinação de um número de frames de entrada que permita criar a melhor rede possível fica limitada a utilidade de se usar mais entradas e a capacidade do algoritmo de treinamento de gerar bons resultados em redes maiores.

Se a medida EIS tivesse sido zero, mesmo com uma matriz de confusão não ideal como a obtida, poder-se-ia parar por aqui e dizer que, seguindo-se a metodologia de treinamento aplicada, a utilização de MLPs como técnica de classificação se mostra suficiente para criar um sistema prático de reconhecimento de gestos. Contudo, a medida EIS não foi

zero e a sensibilidade para a classe de gestos nulos foi de apenas 90,45%. Isso serviu de base para o próximo experimento.

5.2.2 Experimento 2

Devido a baixa sensibilidade obtida para a classe de gestos nulos e uma medida EIS insatisfatória no primeiro experimento, supôs-se que talvez se poderia obter melhores resultados se essa classe fosse separada em suas duas constituintes. A classe "fecha" ocupava 44.8% da classe nula, em uma região concentrada do espaço de atributos, enquanto que a classe de transição estava mais dispersa, ocupando os 55.2% restantes. Imaginou-se que isso poderia ter sido a causa do mal desempenho.

Como no primeiro experimento, foi realizada uma busca extensiva no espaço de configurações, sendo cada uma delas, novamente, treinada 120 vezes. Dessa vez, o experimento demorou 38 horas e 25 minutos – cerca de 9 horas a menos do que o anterior. Esse dado, por si só, mostra que com a nova divisão de classes elas ficaram em regiões mais coesas no espaço de atributos, permitindo um aprendizado mais rápido pelas MLPs.

A rede mais bem adaptada, com 33 neurônios na camada oculta, atuou, novamente, sobre 5 frames de entrada. Obteve um erro SNI de 6, SFI de 26, e, por conseguinte, EIS de 32. A sua matriz de confusão, com medidas adicionais é mostrada na Tabela 5.3.

A nova acurácia, de 94,54%, permaneceu próxima a obtida anteriormente. Também permaneceu o resultado perfeito com relação a não confusão das classes definidas a partir do estudo das distâncias no espaço de atributos. Além disso, o valor maior de erro EIS não é indício de que a nova melhor rede é pior do que a obtida no primeiro experimento, mas apenas de que houve uma granulação nos segmentos, e, consequentemente, um aumento nas possibilidades de erro.

O resultado marcante desse experimento, que serve como base para o próximo, é o fato de ter havido algo problemático com a classe de transição, que obteve uma pobre sensibilidade de apenas 78,9%.

Tabela 5.3: Matriz de confusão da melhor rede neural do experimento 2

		Classe Predita							Sens.
		“fecha”	**“atira”**	**“dois”**	**“surf”**	**“cê”**	**“abre”**	**“trans.”**	
Classe de Ref.	**“fecha”**	15,84	0	0	0	0	0	0,33	97,96
	“atira”	0	13,18	0	0	0	0	0,2	98,51
	“dois”	0	0	12,17	0	0	0	0,16	98,70
	“surf”	0	0	0	12,29	0	0	0,11	99,11
	“cê”	0	0	0	0	12,02	0	0,2	98,36
	“abre”	0	0	0	0	0	12,81	0,13	98,99
	“trans.”	0,9	0,84	0,53	0,81	0,44	0,82	16,23	78,90
Precisão		94,62	94,01	95,83	93,82	96,47	93,99	93,49	94,54

Fonte: elaborado pelo autor.

5.2.3 Experimento 3

Uma baixa sensibilidade para a classe de transição significa que, quando o classificador deveria predizê-la, acaba apontando para uma das outras classes. Basicamente, haveria duas possibilidades de explicação para isso: insuficiência de dados de treinamento; ou intersecção de classes no espaço de atributos. Com relação a insuficiência de dados, supos-se que isso não seria o caso, já que a maior parte dos frames (20.56%) são de transição. Desse modo, tomou-se como hipótese de que haveria uma intersecção de classes.

Ao analisar-se a diferença entre os dados de referência e a classificação realizada pela melhor MPL obtida no experimento anterior, observou-se uma presença quase constante de erros de classificação exatamente nas bordas dos segmentos de transição. Isso levou a percepção de que definição das regiões de transição, como sendo do primeiro ao último frame em que se detecta movimentação para a troca gesto, poderia ser a fonte do problema.

De fato, o caráter do primeiro ou último frame de transição, se analisado pelo seu vetor de atributos ou apenas visualmente pela sua imagem segmentada (ver Figura 4.7), não é radicalmente distinto do caráter de um outro frame próximo pertencente a outra classe. Desse modo, para testar-se a hipótese de interseção de classes, realizou-se uma nova rotulação dos dados, em que se permitiu que as bordas dos segmentos fossem levemente deslocadas. Tal deslocamento foi automaticamente realizado utilizando-se a melhor MLP do experimento anterior.

Em média, o avanço ou recuo das bordas dos segmentos foi de 1.53 frames e, no total,

5.34% dos frames receberam uma nova rotulação. Na Tabela 5.4há uma comparação entre o tamanho dos segmentos, antes e depois do deslocamento. Nota-se que houve uma expansão de todos os segmentos sobre as regiões de transição, embora esta ainda tenha permanecido com a maior parte dos dados.

Tabela 5.4: Comparativo entre a distribuição dos frames antes e depois do deslocamento das bordas dos segmentos

Classe	**Porcentagem do total de frames**		**Comprimento médio dos segmentos (frames)**	
	Antes	**Depois**	**Antes**	**Depois**
“fecha”	16.17	16.84	37.80	39.37
“atira”	13.38	14.14	31.28	33.06
“dois”	12.33	12.71	29.68	30.59
“surf”	12.40	13.22	29	30.9
“cê”	12.22	12.39	29.41	29.82
“abre”	12.94	13.6	30.26	31.80
“transição”	20.56	17.11	8.13	6.76

Fonte: elaborado pelo autor.

Com a nova rotulação, o experimento realizado foi como nos anteriores. Realizou-se a mesma busca extensiva no espaço de configuração das MLPs. A duração foi de 24 horas e 39 minutos - cerca de 14 horas a menos do que no último experimento, o que indica uma maior coesão dos dados e, consequentemente, a maior rapidez no aprendizado obtida.

A rede mais bem adaptada, com 36 neurônios na camada oculta, atuou, dessa vez, sobre 4 frames de entrada. Obteve um erro SNI de 7, SFI de 23, e, por conseguinte, EIS de 30. Essa medida EIS foi devido a apenas alguns erros em frames isolados em alguns segmentos, já que, com 72 frames mal classificados, o comprimento médio de trechos errados em sequência ficou em apenas 1,06 frames. A matriz de confusão, com medidas adicionais é mostrada na Tabela 5.5.

A acurácia, de 99.12%, melhorou com relação a anterior, assim como todas as outras medidas de precisão e de sensibilidade, que ficaram bastante elevadas. Em especial, a sensibilidade para a classe de transição passou de 78,9% para razoáveis 96,14%. Essas

medidas evidenciam que a hipótese inicial de intersecção de classes estava correta. Ou seja, a classe de transição estava com um domínio amplo demais, invadindo regiões que seriam mais propriamente pertencentes às outras classes.

Com o bom resultado encontrado, embora ainda que não ideal, acredita-se que se chegou próximo do limite do que é possível de se obter das MLPs para o conjunto de dados e metodologia utilizadas.

Tabela 5.5: Matriz de confusão da melhor rede neural do experimento 3

		Classe Predita							**Sens.**
		“fecha”	**“atira”**	**“dois”**	**“surf”**	**“cê”**	**“abre”**	**“trans.”**	
Classe de Ref.	**“fecha”**	16,8	0	0	0	0	0	0,04	99,76
	“atira”	0	14,10	0	0	0	0	0,04	99,72
	“dois”	0	0	12,7	0	0	0	0,01	99,92
	“surf”	0	0	0	13,18	0	0	0,04	99,7
	“cê”	0	0	0	0	12,35	0	0,04	99,68
	“abre”	0	0	0	0	0	13,54	0,06	99,56
	“trans.”	0,15	0,1	0,06	0,13	0,11	0,11	16,45	96,14
Precisão		99,12	99,3	99,53	99,02	99,12	99,19	98,62	99,12

Fonte: elaborado pelo autor.

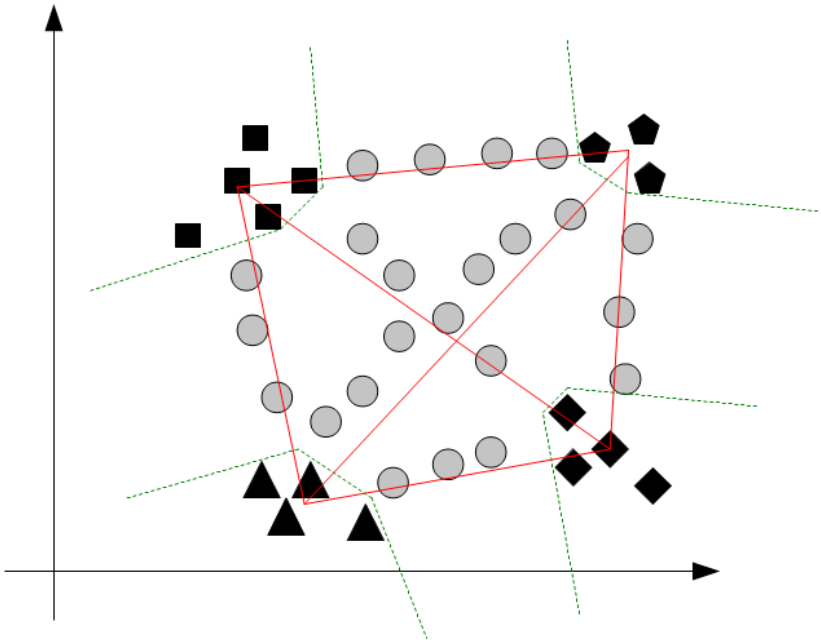
5.2.4 Experimento 4

Nos experimentos anteriores, a melhor MPL atuou com 33, 34, ou 36 neurônios na camada oculta. Embora não haja um método analítico para determinar-se o porquê desses valores, conjecturou-se que isso poderia estar relacionado ao número de hiperplanos necessários para se isolar as seis classes de gestos no espaço de atributos. Há um total de 30 transições entre elas, considerando-se ambos sentidos, de modo que talvez um bom classificador possa ser construído considerando-se apenas 30 hiperplanos. Eles ficariam situados ortogonalmente às linhas de transição que conectam cada classe entre si.

Desse modo, definiu-se um classificador que utiliza 5 hiperplanos para isolar cada classe. Os vetores normais de cada um deles são dados pelas direções entre a posição central da classe que isolam e o restante das posições centrais das outras classes. Já as distâncias à classe isolada foram ajustadas de modo deixem todos os seus elementos de apenas de um

lado, ao mesmo tempo que fiquem o mais próximo do seu centro. A ideia, para quatro classes, está ilustrada na Figura 5.2.

Figura 5.2 Ilustração da ideia da disposição dos hiperplanos do classificador considerado no experimento 4



As figuras pretas representam os padrões das classes de gestos, enquanto que as cinzas as transições entre elas. O tracejado em verde corresponde aos hiperplanos ao redor das classes e as linhas em vermelho as ligações entre os seus centros.
Fonte: elaborado pelo autor.

O classificador assim construído, avaliado com os padrões do último experimento, obteve SNI de 277, SFI de 698 e, portanto, EIS de 1075. A matriz de confusão obtida é apresentada na Tabela 5.6. Praticamente não houve confusão de classes, o que demonstra, novamente, o bom espaçamento entre elas. A sensibilidade perfeita para todas as classes, exceto “surf” e “transição”, demonstra que os 5 hiperplanos foram suficientes para isolá-las. Já a precisão de 100% para a classe de transição demonstra que, se um padrão não estiver contido pelos 5 hiperplanos de alguma classe, ele efetivamente pertence a classe de transição.

As precisões não muito boas para a maioria das classes, uma parca sensibilidade de 25,83% para as transições, e a alta pontuação EIS, apontam a impraticabilidade do classificador. Contudo, como a acurácia obtida foi de 83,77%, há alguma validade na ideia de que o que as MLPs podem ter feito foi ajustar os hiperplanos ao redor das 6 classes, porém

considerando-se vetores normais oblíquos às linhas de transição, de modo a melhor se delimitar as regiões de transição.

Tabela 5.6: Matriz de confusão para o classificador analisado no experimento 4

		Classe Predita							**Sens.**
		“fecha”	**“atira”**	**“dois”**	**“surf”**	**“cê”**	**“abre”**	**“trans.”**	
Classe de Ref.	**“fecha”**	16,84	0	0	0	0	0	0	100
	“atira”	0	14,14	0	0	0	0	0	100
	“dois”	0	0	12,71	0	0	0	0	100
	“surf”	0,02	3,53	0	9,67	0	0	0	73,15
	“cê”	0	0	0	0	12,39	0	0	100
	“abre”	0	0	0	0	0	13,6	0	100
	“trans.”	1,97	5,36	1,91	0,31	1,81	1,33	4,42	25,83
Precisão		89,43	61,4	86,94	96,89	87,25	91,09	100	83,77

Fonte: elaborado pelo autor.

6 CONCLUSÃO E TRABALHOS FUTUROS

A partir dos experimentos realizados, demonstrou-se os seguintes resultados com relação a sistemas de reconhecimentos de gestos em tempo real que utilizem MPLs como técnica de classificação:

a. A utilização dos momentos invariantes de Hu, além da relação entre perímetro e raiz da área, como vetor de atributos, aliada a escolha de um conjunto de classes de gestos que maximizem as mínimas distâncias no espaço de atributos, possibilita que a rede MLP atinja classificação perfeita com relação a confusão entre as classes escolhidas;
b. Criar uma classe específica para as regiões de transições entre gestos permite um aprendizado mais rápido pelas MPLs, embora isso não necessariamente venha a gerar um maior desempenho para o classificar.
c. Permitir que o classificador atue em mais de um frame pode aprimorar a sua capacidade de classificação correta, uma vez que há interdependência entre um frame e o seguinte.
d. Com relação a criação do banco de dados de referência, especial cuidado deve ser tomado ao rotular-se as regiões de transição de gestos, já que, em última instância, essa é uma região cujo início ou fim é impreciso. Sem isso, pode-se comprometer a eficácia do treinamento das redes.
e. O resultado do último experimento demonstra que um classificador construído de modo trivial pode não ser suficiente para se obter bons resultados. Isso sugere que o uso de técnicas mais sofisticadas de classificação são necessárias, como as MLPs utilizadas.

A melhor rede neural encontrada obteve boas métricas de avaliação. Porém, se fosse utilizada sozinha em um sistema prático de reconhecimento de gestos, o usuário poderia ficar descontente. Isso porque, como a medida de erros de identificação de segmentos não foi nula, algumas vezes o sistema reconheceria comandos que não foram realizados.

Com relação a esse problema, há uma solução simples. Pode-se utilizar, como é de praxe em circuitos eletrônicos que utilizam botões, um mecanismo de *debouncer*. Como a acurácia da rede foi bastante elevada e o comprimento dos trechos de classificação errada teve

comprimento médio de apenas 1,06 frames, poder-se-ia esperar, antes de enviar um comando, que a classificação fique estável por alguns frames seguidos. Isso minimizaria o problema.

Para trabalhos futuros, sugere-se avaliar as MLPs para um conjunto maior de classes de gestos, considerando-se a análise das maiores menores distâncias no espaço de atributos, de modo a verificar até que ponto a ausência de confusão entre elas se mantêm. Além disso, com relação a rotulação dos dados, poderia verificar-se se haveria melhor adaptação das redes se fosse realizada uma rotulação a partir de conceitos fuzzy. Talvez isso possa ser particularmente útil para as bordas dos segmentos.

REFERÊNCIAS

[1] MURTHY, G. R. S.; JADON, R. S. “Hand Gesture Recognition using Neural Networks”, Department of Computer Applications, Madhav Institute of Technology and Science, Gwalior, M.P. India.

[2] SU, M.; JEAN, W.; CHANG, H. "A Static Hand Gesture Recognition Using a Composite Neural Networks", Departament of Electrical Engineering, Tamkang University, Taiwan, República da China

[3] NETO, P.; PEREIRA, D.; PIRES, J. N.; MOREIRA, A. P. “Real-Time and Continuous Hand Gesture Spotting: an Approach Based on Artificial Neural Networks”, 2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Alemanha, Maio, 2013.

[4] YOON, H. et al. “Hand Gesture Recogbition Using Hidden Markov Models”, Systems Engineering Research Institute, Image Processing Div., Daejeon, Coréia do Sul.

[5] YANG, Z.; LI, Y.; CHEN, W.; ZHENG, Y. “Dynamic Hand Gesture Recognition Using Hidden Markov Models”, The 7th International Conference on Computer Science & Education (ICCSE 2012), Julho, 2012. Melbourne, Austrália.

[6] SHRIVASTAVA, R. “A Hidden Markov Model based Dynamic Hand Gesture Recognition System using OpenCV”, Dept. of Electronics and Communication Engineering, Maulana Azad National Institute of Technology, Bhopal-462001, India.

[7] JEON, M.; YANG, S.; BIEN, Z. “User Adaptive Hand Gesture Recognition using Multivariate Fuzzy Decision Tree and Fuzzy Garbage Model”, FUZZ-IEEE 2009, Agosto, 2009, Korea.

[8] YUN, L.; PENG, Z. “An Automatic Hand Gesture Recognition System Based on Viola-Jones Method and SVMs”, 2009 Second International Workshop on Computer Science and Engineering, College of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, China.

[9] SHAPIRO, L. G.; STOCKMAN, G. C. “Computer Vision”, p.279-325, New Jersey: Prentice-Hall, 2001.

[10] AWCOCK, G. W; THOMAS, R. Feature Extraction. In: Applied Image Processing,. 1 ed., Ed. McGraw-Hill. Londres: Department of Electrical and Electronic Engineering University pf Brighton, 1995. p.162-165.

[11] HAYKIN, S. "Neural Networks", 2 ed. , Upper Saddle River, New Jersey: Prentice Hall, 1999.

[12] CYBENKO, G.: “Approximations by superpositions of sigmoidal functions”. In: Mathematics of Control, Signals, and Systems, 1989.

[13] CHEN, Q.; GEORGANAS, N. D.; PETRIU, E. M., Real-time Vision-based Hand Gesture Recognition Using Haar-like Features. Instrumentation and Measurement Technology Conference – IMTC 2007, Ottawa, IEEE, 2007.

[14] LIU, Y.; YIN, Y.; ZHANG, S. Hand Gesture Recognition Based on Hu Moments in Interaction of Virtual Reality. 4th International Conference on Intelligent Human-Machine Systems and Cybernetics, Qingdao, IEEE, 2012.

APÊNDICE – ALGORITMOS

Filtro da mediana

```
filtroDaMediana3x3(A)
      entrada: A, imagem original
      saída: B, imagem filtrada
INÍCIO
      PARA cada canal de cor de A FAÇA
            INÍCIO
                  PARA cada pixel (x, y) de A FAÇA
                        INÍCIO
                              B(x, y) = mediana3x3(A, x, y);
                        FIM
            FIM
      RETORNA B;
FIM

mediana3x3(A, x, y)
      entrada: A, imagem;
                x, y: posição;
      saída: m, mediana da janela 3x3 centrada em (x, y)
      variáveis: valores, vetor de 9 posições;
                  n, contador;
INÍCIO
      n := 0;
      PARA dx = -1 ATÉ +1 FAÇA
      INÍCIO
            PARA dy = -1 ATÉ +1 FAÇA
            INÍCIO
                  valores[n] := A(x + dx, y + dy);
                  n := n + 1;
            FIM
      FIM
      ordena(valores);
      m := valores[5];
      RETORNA m;
FIM
```

Algoritmo de remoção do antebraço

```
removeAntebraço(B)
      entrada: B, imagem binária segmentada
      saída: R, imagem binária sem a região do antebraço
INÍCIO
      pontoA := centroDeMassa(B);
      {pontoBorda1, pontoBorda2}:= encontraPontosDaBorda(B);
      pontoB := média(pontoBorda1, pontoBorda2);
      PARA s = 0 ATÉ norma(pontoB – pontoA) FAÇA
            INÍCIO
                  {pontoPulso1, pontoPulso2} := encontraPontosDoPulso(B,
pontoA, pontoB, s);
                  espessura[s] := norma(pontoPulso2 - pontoPulso1);
            FIM
      {valorMin, índiceMin} := min(espessura);
      valorDeCorte := 1.1*valorMin;
      índiceDeCorte := encontraPrimeiroÍndice(espessura, valorDeCorte);
      {pontoPulso1, pontoPulso2} :=  encontraPontosDoPulso(B, pontoA,
pontoB, índiceDeCorte);
      R = removePolígono(B, {pontoBorda1, pontoBorda2, pontoPulso1,
pontoPulso2});
      RETORNA R;
FIM

encontraPontosDoPulso(B, pontoA, pontoB, s);
      entrada: B, imagem binária;
                  pontoA, pontoB: pontos que descrevem a linha
                          do antebraço;
                  s: distância a partir de pontoA em direção a
         pontoB;
      saída: {pontoPulso1, pontoPulso2}, pontos da borda do pulso;
INÍCIO
      direçãoTangente := normaliza(pontoB – pontoA);
      direçãoOrtogonal := rotaciona90º(direçãoTangente);
      pontoInicial :=  pontoA + direçãoTangente*s;
      pontoPulso1 := pontoInicial;
      ENQUANTO dentroLimitesImagem(B, pontoPulso1) E  dentroAntebraço(B,
pontoPulso1) FAÇA
      INÍCIO
            pontoPulso1 :=  pontoPulso1 + direçãoOrtogonal;
```

```
      FIM

      pontoPulso2 := pontoInicial;
      ENQUANTO dentroLimitesImagem(B, pontoPulso2) E  dentroAntebraço(B,
pontoPulso2) FAÇA
      INÍCIO
            pontoPulso2 :=  pontoPulso2 – direçãoOrtogonal;
      FIM

      RETORNA {pontoPulso1, pontoPulso2};
FIM
```

Algoritmo Perímetro Aproximado

```
perimetroAproximado(B);
      entrada: B, imagem binária;
      saída: perímetro, perímetro aproximado figura
INÍCIO
      perímetro = 0;
      incremento[0] = 0;
      incremento[1] = 1;
      incremento[2] = sqrt(2);
      incremento[3] = 1;
      incremento[4] = 0;
      PARA cada pixel (x, y) em B FAÇA
            INÍCIO
                  perimetro = perimetro +
                              incremento[contaVizinhos(B, x, y)];
            FIM
      RETORNA perimetro;
FIM
```

ANEXO – Artigo do Trabalho de Graduação 1

Análise de Redes Neurais Artificiais Aplicadas a um Sistema em Tempo Real de Reconhecimento de Gestos Estáticos de Mão

Guilherme B. Bender

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)

Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

gbbender@inf.ufrgs.br

Abstract. *Belonging to the fields of computer vision and human-machine interaction, this paper presents a research proposal in the area of static hand gestures recognition. Relying on artificial neural networks as a basic technique for pattern recognition, it's proposed to study the effects of the variation of two parameters of the architecture of a real time gesture recognition system, namely: (a) the memory of the system, expressed in the number of the network's input; and (b) the number of neurons in the hidden layer. The system possesses as data input only the video stream captured by an ordinary camera. For each video frame is performed the hand segmentation, and then it's computed the geometric relationship between perimeter and area, and also Hu invariant moments, which are used as input variables for the neural network.*

Resumo. *Situado nos campos de visão computacional e de interação homem-máquina, este trabalho apresenta uma proposta de pesquisa na área de reconhecimento de gestos estáticos de mão. Tendo-se como técnica base de reconhecimento de padrões redes neurais artificiais, propõe-se estudar os efeitos da variação de dois parâmetros da arquitetura de um sistema de reconhecimento de gestos em tempo real, a saber: (a) memória do sistema, traduzida no número de entradas da rede; e (b) número de neurônios na camada oculta. O sistema possuí como entradas de dados o vídeo capturado por apenas uma câmera comum. Para cada quadro de vídeo são realizados a segmentação da mão, sendo então calculados a relação geométrica entre perímetro e área, e os momentos invariantes de Hu, que serão utilizados como variáveis de entrada da rede neural.*

1. Introdução

Há décadas temos experimentado uma contínua evolução, em ritmo exponencial, na velocidade de processamento e na capacidade de memória do hardware de que dispomos. Com isso, pesquisas nas áreas que demandam maior poder computacional, como a de visão computacional, e mais especificamente a de reconhecimento de gestos, tem tido cada vez mais oportunidades de aflorarem. À medida que elas prosperam, aplicações vão surgindo e outras

formas de se interagir com os computadores – além do tradicional uso de mouses, teclados e telas de toque – começam a se tornar cada vez mais confiáveis e acessíveis ao grande público.

Recentemente, em outubro de 2013, a área de reconhecimento de gestos ganhou novo destaque com a compra pelo Google da startup Flutter por US$ 40 milhões [1]. O primeiro produto disponibilizado pela empresa, “Flutter App”, faz de uso de uma webcam para reconhecer gestos simples de mão, realizados entre 30 e 180 cm da câmera, para controlar aplicativos como iTunes, Spotify, Netflix, e QuickTime [2][3].

Ao proverem uma forma mais natural de se interagir com computadores, sistemas de reconhecimento de gestos podem ser especialmente úteis no controle de robôs, de gadgets espalhados pela casa e de dispositivos secundários em automóveis. Também podem proporcionar uma experiência mais imersiva e interativa em jogos, além de possibilitarem o uso de sistemas computacionais por pessoas que, de outro modo, não conseguiriam utilizá-los, como certos idosos e pessoas debilitadas.

Contudo, para que esses sistemas sejam confiáveis e possam se tornar mais presentes no nosso cotidiano, métodos robustos de visão computacional são necessários. Com isso em mente, a proposta desse trabalho é estudar extensivamente certa arquitetura de redes neurais artificiais como método efetivo para o desenvolvimento de um sistema em tempo real para o reconhecimento de gestos estáticos de mão.

As redes neurais, empregadas como técnica de reconhecimento de padrões, são bem conhecidas no meio acadêmico por apresentarem boas capacidades de aprendizado e de generalização, além de que, quando treinadas, apresentam uma computação bastante direta e eficiente, tornando-as ideais para aplicações em tempo real. No entanto, para uma dada aplicação específica, o problema de se determinar a melhor topologia de rede ainda não é um problema totalmente resolvido. É justamente neste ponto que pretendemos contribuir com a nossa pesquisa.

2. Objetivo Geral

Desenvolver um sistema de tempo real de baixo custo, acessível, e simples de usar que permita enviar comandos a um computador por meio de gestos de forma robusta, precisa, e que possua um grau de confiabilidade similar ao de dispositivos de entrada padrão, como mouse e teclado.

3. Objetivos Específicos

Avaliar, no contexto do objetivo geral, a eficácia da utilização de redes neurais artificias sem realimentação, com uma camada oculta e treinadas de modo supervisionado, como técnica de reconhecimento de padrões aplicada a um sistema de tempo real para o reconhecimento de gestos estáticos de mão, sendo que a entrada de dados é aquela exclusivamente fornecida por uma câmera de vídeo comum.

Pretende-se concretizar tal objetivo, inicialmente, por meio de uma análise sistemática dos efeitos da variação de dois parâmetros da arquitetura do sistema, a saber: (a) número de quadros sobre o qual a rede neural irá atuar (memória do sistema); e (b) número de neurônios na camada oculta.

4. Trabalhos Relacionados

Embora entre os artigos pesquisados durante a elaboração deste trabalho uma grande quantidade deles utilize redes neurais como técnica de reconhecimento de padrões, nenhum dos mesmos apresentou um estudo extensivo sobre os efeitos da variação das topologias de

rede no desempenho dos sistemas de reconhecimento de gestos.

No trabalho desenvolvido por G.R.S. Murthy e R.S. Jadon - “Hand Gesture Recognition using Neural Networks” [4] - é proposto um sistema de reconhecimento de gestos de mão que utiliza como técnica de reconhecimento de padrões redes neurais artificiais sem realimentação, treinadas de modo supervisionado, com algoritmo backpropagation, para o reconhecimento de 10 categorias de gestos de mão. Os dados são obtidos por uma webcam e passam por um processo de segmentação, sendo gerado uma imagem binária, que posteriormente é redimensionada para ter uma resolução de 30x30. Tal imagem é utilizada como entrada da rede neural, que foi escolhida para ter 7 neurônios na camada oculta. A precisão média do sistema relatada pelos autores foi de 89%.

Já no trabalho desenvolvido por Mu-Chun Su, Woung-Fei Jean, e Hsiao-Te Chang - "A Static Hand Gesture Recognition Using a Composite Neural Network." [5] - é proposto um sistema em tempo real para o reconhecimento de gestos que utiliza como dados de entrada dez medidas dos ângulos das juntas dos dedos da mão, fornecidas por uma luva especial. O reconhecimento é realizado por meio de uma rede neural composta, treinada de modo supervisionado pelo algoritmo SDDL (supervised decision-directed learning). O sistema foi avaliado para a classificação de 51 gestos estáticos de mão, sendo cada um deles realizados 10 vezes por 4 pessoas, formando um total de 2040 dados de base, ficando 75% destes para o treinamento da rede, e os 25% restantes para teste. A acurácia relatada do sistema foi de 100% para os dados de treinamento e de 93.9% para os dados de teste.

Assim como no trabalho anterior, o desenvolvido por Pedro Neto, Dário Pereira, et al. - “Real-Time and Continuous Hand Gesture Spotting: an Approach Based on Artificial Neural Networks ” [6] – também propõe um sistema de reconhecimento de gestos de tempo real cuja entrada de dados é aquela fornecida por luvas especias. O sistema é treinado de modo supervisionado para realizar o controle de um robô industrial por meio de gestos de mão. Na arquitetura do sistema são utilizadas duas redes neurais em série, sendo a primeira utilizada para reconhecer se um gesto é comunicativo ou não-comunicativo (corresponde a transição entre gestos) está sendo feito, e a segunda para classificar os gestos comunicativos em suas categorias específicas. Ambas redes neurais são sem realimentação, com uma camada oculta, treinadas com o algoritmo backpropagation. Nos resultados experimentais, é relatado uma taxa de precisão de 99% para o reconhecimento de dez gestos e de 96% para o reconhecimento de trinta gestos.

Além das técnicas de redes neurais, diversas propostas de sistemas de reconhecimento de gestos de mão foram desenvolvidas. Entre outras técnicas, pode-se destacar: modelos ocultos de Markov (HMM – hidden markov models) [7][8][9]; árvores de decisão fuzzy [10]; e máquinas de vetor de suporte (SVM – support vector machine) [11].

5. Gestos

Gestos são ações corporais visíveis e voluntárias, pelas quais um determinado significado é transmitido [12]. Há diversos tipos de gestos, entre os quais se destacam: gestos de cabeça, de olhos, de mãos, de posições do corpo e de expressões faciais.

A classe de gestos que analisaremos neste trabalho - a de gestos de mão - é a que ocorre com maior frequência. Isso é devido a habilidade e precisão da mão humana em adquirir um grande número de configurações claramente perceptíveis [12].

Outra classificação dos gestos é a sua divisão entre gestos estáticos e dinâmicos. Gestos estáticos são aqueles que podem ser caracterizados por apenas uma imagem, enquanto que gestos dinâmicos envolvem movimento e precisam de uma sequência de imagens para

serem caracterizados.

O escopo desse trabalho é o reconhecimento em tempo real de gestos estáticos de mão. Nessa abordagem, embora os gestos sejam estáticos, há um componente dinâmico no sistema: o usuário realiza uma sequência de sinais ao longo do tempo que devem ser interpretados como comandos a um computador - é como se cada categoria de gesto estático representasse um botão, uma tecla, que o usuário pode apertar.

Uma das dificuldades do reconhecimento em tempo real decorre do fato que na transição entre gestos estáticos que devem ser interpretados como um comando, haverá uma sequência de padrões gestuais não-comunicativos [6]. Nessa sequência não-comunicativa, poderão ocorrer padrões em alguns dos quadros do vídeo que, se analisados isoladamente, podem ser caraterísticos de uma das categorias de gestos definidas como significativas. Consequentemente, um tratamento adequado do problema envolve também a análise das entradas anteriores, tornando-se mais robusto trabalhar com reconhecimento ao nível de uma série temporal, em vez de apenas utilizar-se um classificador aplicado isoladamente quadro a quadro do vídeo.

6. Categorias de Gestos Estáticos

O objetivo é gerar um sistema confiável de interação com um computador que permita enviar comandos simples para controlar os mais diversos aplicativos. Tais comandos podem ter vários significados, tais como: pausa, toca, próximo, anterior, abre, fecha, liga, desliga, e assim por diante.

Certamente seria frustrante para um usuário que, quando interagindo com o sistema, fizesse algum gesto desejando que o filme começasse e o aplicativo simplesmente compreendesse, por exemplo, que o comando fosse para fechá-lo. Ou que, quando não fizesse gesto algum, de modo espúrio o aplicativo reconhecesse que o filme deveria começar.

A fim de tentar minimizar esses problemas e buscar uma maior robustez no sistema, será útil definirmos categorias de gestos estáticos que possuam três características, a saber:

(a) Gestos realizados próximos a câmera de vídeo são interpretados do mesmo modo que seriam se fossem realizados longe dela.

(b) Gestos realizados na porção superior ou inferior, à direita ou à esquerda do campo visual da câmera são igualmente interpretados.

(c) Por fim, independentemente da orientação em que forem realizados, os gestos serão reconhecidos do mesmo modo.

Tais características dos gestos serão refletidas no vetor de atributos extraídos das imagens, e terão, respectivamente às características citadas, as seguintes propriedades: invariância à escala, invariância à translação, e invariância à rotação.

As categorias específicas de gestos para o qual o sistema será treinado para reconhecer serão definidas na segunda etapa do trabalho. Desejamos antes, para um conjunto razoável de possíveis categorias de gestos, realizar um breve estudo da distância no espaço de atributos entre essas categorias, e então escolher cinco ou mais delas que estejam mais afastadas entre si, novamente com o intuito de tornar o sistema mais robusto e preciso.

7. Categorias de Gestos Dinâmicos a Partir de Gestos Estáticos

A rigor, para ser capaz de enviar qualquer tipo de comando a um computador em um sistema de reconhecimento de gestos, é suficiente o reconhecimento de dois tipos de gestos, digamos gesto “0” e gesto “1”, uma vez que, empregando-se uma codificação binária para a

sequência de gestos estáticos, pode-se especificar qualquer número de gestos dinâmicos, que podem ser associados à qualquer quantidade de comandos.

Ou seja, a definição de um grupo pequeno de categorias de gestos estáticos a serem reconhecidos não é um fator limitante das possibilidades de comandos que o sistema será capaz de interpretar. É claro que com apenas gestos "0"s e "1"s, o envio de comandos se tornaria uma tarefa tediosa, cansativa, e provavelmente difícil de aprender. Contudo, com a definição de apenas 5 categorias básicas de gestos estáticos, e definindo-se uma semântica para cada dois gestos sucessivos, já é possível enviar 25 comandos distintos – apenas um comando a menos do que o número de letras de nosso alfabeto.

8. Captura de Vídeo e Geração dos Conjuntos de Dados de Treinamento e de Teste

Como o foco deste trabalho está na análise das redes neurais e não em métodos de localização e segmentação das mãos no espaço, tornaremos estas duas tarefas o mais simples possível. Coletaremos os vídeos utilizando uma webcam de boa qualidade – a LifeCam HD-3000 da Microsoft –, com condições de iluminação adequadas e fundo contrastante com a cor e textura das mãos. Além disso, apenas a mão estará em foco.

De posse dos vídeos em que são realizadas diversas sequências de gestos representantes das categorias pré-definidas, serão realizadas as seguintes etapas para a geração dos conjuntos de treinamento e de testes:

(a) Cada vídeo é manualmente analisado, anotando-se estruturadamente em um arquivo os quadros em que se julgue que ocorreu a transição de gesto, assim como para qual categoria essa transição foi realizada.

(b) Cada vídeo é automaticamente analisado, com cada quadro passando pelo processo de segmentação da mão e de extração do vetor de atributos. Nesta etapa é gerado um arquivo que indica, para cada quadro do vídeo, qual o seu respectivo vetor de atributos.

(c) Como o aprendizado é supervisionado, para cada vídeo ambos arquivos anteriores serão unidos em um só, que conterá a sequência temporal do vetor de atributos já associado a sua correta classificação.

9. Segmentação da Mão

Como mencionamos na sessão anterior, cada quadro dos vídeos utilizados para gerar o conjunto de treinamento e de teste precisará passar pelo processo de segmentação da mão. Isso envolverá duas etapas: primeiro, a região da mão e do antebraço é segmentada com relação ao fundo; após, os pulsos e o antebraço são removidos, restando apenas a mão.

Uma vez que tomaremos o cuidado de capturar os vídeos dos gestos com um fundo conhecido e com iluminação adequadas, o processo inicial de segmentação ficará facilitado, bastando realizar os seguintes passos:

1) Para cada pixel do quadro realiza-se a subtração algébrica da cor conhecida do fundo do valor da cor do pixel.

2) Aplica-se um limiar de cor ao valor obtido, de modo a separar a mão e antebraço, gerando-se uma imagem binária.

3) Por fim, aplica-se o filtro da mediana para garantir que possíveis ruídos salt-and-pepper sejam removidos ao mesmo tempo que os detalhes fiquem preservados.

De posse da imagem binária, resta remover a região dos pulsos e do antebraço.

Percorrendo-se a borda da imagem pode-se encontrar facilmente dois pontos que pertencem a essa região. A partir desses pontos, seguindo-se na linha do antebraço imagem adentro, calcula-se a sua espessura. O início da região da mão – os pulsos –, pode ser localizada por uma variação mais acentuada da espessura [13].

10. Extração de Atributos

Como expomos na sessão "Categorias de gestos estáticos", optamos por escolher categorias de gestos que atendam a três propriedades: invariância à escala, invariância à translação, e invariância à rotação. Desse modo, embora em princípio poderíamos utilizar como atributos de cada imagem binária da mão segmentada o próprio valor de cada pixel, decidimos realizar uma redução de dimensionalidade, extraindo-se atributos que já carreguem em si as três propriedades desejadas. Com isso, esperamos reduzir o tempo de aprendizagem da rede neural, reduzir a quantidade de dados necessários nos conjuntos de treinamento e de teste, reduzir a complexidade computacional da rede, além de tornar o sistema como um todo mais robusto e preciso.

Desse modo, os atributos que escolhemos utilizar e que atendem aos nossos requisitos são a relação entre perímetro e área, e os momentos invariantes de Hu. Tais atributos geram bons resultados e podem ser recorrentemente encontrados na literatura de reconhecimento de gestos de mão [9][11][14].

11. Reconhecimento de Gestos Utilizando-se Redes Neurais

As técnicas de resolução de problemas baseadas em redes neurais tem se mostrado ferramentas confiáveis no reconhecimento de gestos, apresentando capacidades de aprendizado e generalização muito boas [15]. Contudo, qual a melhor topologia de rede utilizar para uma dada aplicação ainda não é um problema totalmente resolvido.

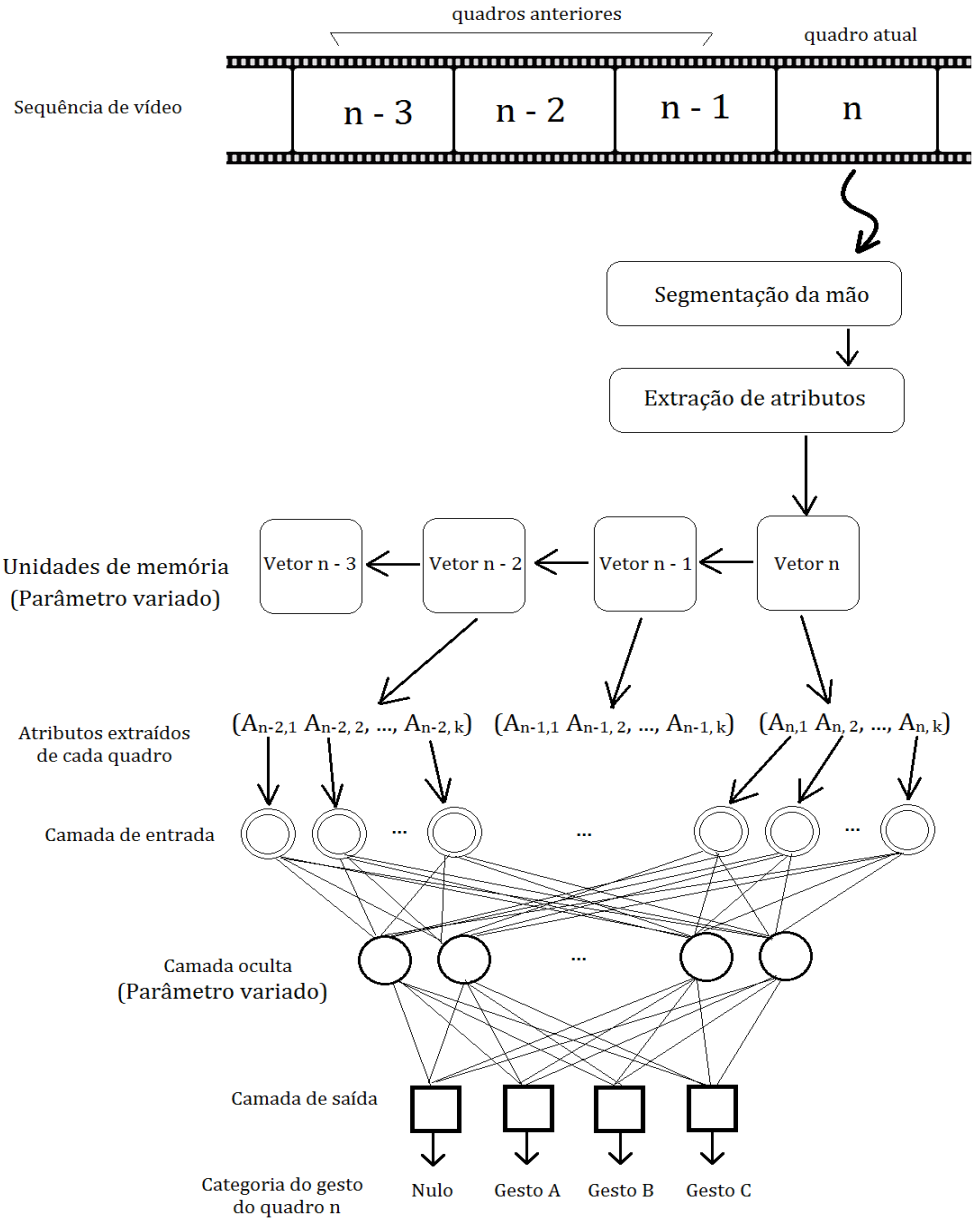
Quanto ao número de camadas da rede, pode-se provar que qualquer função contínua pode ser representada com precisão arbitrária utilizando-se uma rede de apenas uma camada oculta [16]. Tal prova é evidenciada empiricamente pela observação de que redes de uma única camada oculta em geral são suficientes para a maioria dos problemas [17]. No entanto, não se sabe exatamente como determinar o número ótimo de neurônios da camada oculta, de modo que a definição da topologia a ser utilizada é comumente baseada na observação prática de que um número de neurônios entre o tamanho das camadas de entrada e de saída produzirá bons resultados [17].

É nesse ponto que entra o trabalho de análise aqui proposto. Tendo como base uma rede neural sem realimentação, com uma camada oculta, treinada com algoritmo backpropagation (pode ser encontrado no tópico sobre redes neurais, do livro "Inteligência Artificial", de Stuart Russel e Peter Norvig [16]), pretendemos estudar extensivamente o efeito da variação de dois parâmetros da rede sobre a precisão no reconhecimento de gestos de mão, buscando a configuração que é mais adequada a essa aplicação. Tais parâmetros analisados serão o número de entradas da rede e o número de neurônios na camada oculta.

O número de entradas na rede não será variado alterando-se o número de atributos extraídos de cada quadro do vídeo de entrada, mas sim alterando-se a capacidade de memória do sistema, isto é, a rede neural será alimentada pelos atributos do quadro atual assim como pelos atributos de "n" quadros anteriores.

Abaixo ilustramos a arquitetura geral do sistema.

Figura 1. Visão geral do sistema



12. Avaliação das Diversas Topologias de Rede

Cada uma das topologias de rede neural, com o seu número específico de entradas e de neurônios na camada oculta, após treinadas, serão avaliadas utilizando-se os conjuntos de dados de testes. As métricas empregadas serão:

(a) Taxa global de acertos de reconhecimento;

(b) Taxa global de acertos de reconhecimento, considerando-se apenas os segmentos de vídeo em que há algum gesto realizado;

(c) Taxa global de acertos de "reconhecimento" de gesto nulo, considerando-se apenas os segmentos de vídeo em que há transição de gestos;

(d) Taxa de acertos para cada categoria de gesto;

As redes neurais serão treinadas de modo a maximizarem a taxa global de acertos de reconhecimento.

13. Plataformas de Implementação

Pretende-se implementar todas as etapas e algoritmos do trabalho utilizando-se a linguagem C++ e a biblioteca de processamento de imagens OpenCV.

14. Atividades Para a Etapa Experimental do Trabalho

Na segunda etapa do trabalho, as atividades que deverão ser realizadas, resumidamente são:

- Definir as categorias de gestos estáticos a serem reconhecidas.
- Gravar os vídeos que serão utilizados para gerar os conjuntos de dados de treinamento e de teste.
- Segmentar e classificar manualmente os vídeos, demarcando os gestos realizados.
- Implementar o algoritmo de segmentação da região da mão e antebraço contra o fundo.
- Implementar o algoritmo de remoção da região do antebraço.
- Implementar os algoritmos de extração de propriedades.
- Gerar os conjuntos de dados de treinamento e de testes.
- Implementar o algoritmo de treinamento da rede neural.
- Implementar os algoritmos de avaliação da rede neural.
- Executar os algoritmos de treinamento, teste, e avaliação para cada rede neural variando-se o número de variáveis de entrada e o número de neurônios da camada oculta, obtendo-se ao final um registro de resultados.
- Avaliar e interpretar os resultados obtidos.

Referências

[1] "Google buys human-gesture recognition start-up Flutter", BBC News, 3 de outubro de 2013. Disponível em: http://www.bbc.co.uk/news/technology-24380202.

[2] STEVEN, L. "Look Ma, No Trackpad! Flutter Lets You Control Your Mac With Gestures", 27 de março de 2012, Wired. Disponível em: http://www.wired.com/epicenter/2012/03/look-ma-no-trackpad/

[3] METZ, R. "Hold Your Hand Up to Play Some Music", 3 de abril de 2012, MIT Tech Review. Disponível em: http://www.technologyreview.com/news/427400/hold-your-hand-up-to-play-some-music/

[4] MURTHY, G. R. S.; JADON, R. S. “Hand Gesture Recognition using Neural Networks”, Department of Computer Applications, Madhav Institute of Technology and Science, Gwalior, M.P. India.

[5] SU, M.; JEAN, W.; CHANG, H. "A Static Hand Gesture Recognition Using a Composite Neural Networks", Departament of Electrical Engineering, Tamkang University, Taiwan, República da China

[6] NETO, P.; PEREIRA, D.; PIRES, J. N.; MOREIRA, A. P. “Real-Time and Continuous Hand Gesture Spotting: an Approach Based on Artificial Neural Networks”, 2013 IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Alemanha, Maio, 2013.

[7] YOON, H. et al. “Hand Gesture Recogbition Using Hidden Markov Models”, Systems Engineering Research Institute, Image Processing Div., Daejeon, Coréia do Sul.

[8] YANG, Z.; LI, Y.; CHEN, W.; ZHENG, Y. “Dynamic Hand Gesture Recognition Using Hidden Markov Models”, The 7th International Conference on Computer Science & Education (ICCSE 2012), Julho, 2012. Melbourne, Austrália.

[9] SHRIVASTAVA, R. “A Hidden Markov Model based Dynamic Hand Gesture Recognition System using OpenCV”, Dept. of Electronics and Communication Engineering, Maulana Azad National Institute of Technology, Bhopal-462001, India.

[10] JEON, M.; YANG, S.; BIEN, Z. “User Adaptive Hand Gesture Recognition using Multivariate Fuzzy Decision Tree and Fuzzy Garbage Model”, FUZZ-IEEE 2009, Agosto, 2009, Korea.

[11] YUN, L.; PENG, Z. “An Automatic Hand Gesture Recognition System Based on Viola-Jones Method and SVMs”, 2009 Second International Workshop on Computer Science and Engineering, College of Information Science and Technology, Qingdao University of Science and Technology, Qingdao, China.

[12] PEREIRA, A. C. C. “Gesto”, disponível em: http://psicolinguistica.letras.ufmg.br/wiki/index.php/Gesto

[13] XU, Y.; GU, J.; TAO, Z.; WU, DI. “Bare Hand Gesture Recognition with a Single Color Camera” Department of Physics Science and Technology, Soochow University, Suzhou, China.

[14] SHEN, J. et al. “Vision-Based Hand Gesture Recognition Using Combinational Features”, 2010 Sixth International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Computer Application Research Center, Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China.

[15] MADANI, K. “Industrial and real world applications of artificial neural networks - illusion or reality?”, Informatics in Control: Automation and Robotics I, J. Braz et al., Ed. 2006, p. 11–26.

[16] NORVIG, P.; RUSSELL, S. Inteligência Artificial: tradução da segunda edição; tradução de PubliCare Consultoria. Rio de Janeiro: Elsevier – 2004 – 4 reimpressão, p. 713-724.

[17] HEATON, J. “Introduction to Neural Networks for Java”. 2 ed. Publicado por: Heaton Research, Inc. - novembro de 2005.