USO DE *TRANSFER LEARNING* PARA O RECONHECIMENTO DE GESTOS DINÂMICOS

Clebeson Canuto dos Santos*, Leonardo de Assis Silva†, Felippe Mendonça de Queiroz*, Rodolfo Picoreti*, Jorge Leonid Aching Samatelo*, Raquel Frizera Vassallo*

*Universidade Federal do Espírito Santo
Vitória, Espírito Santo, Brasil

†Instituto Federal do Espírito Santo
Guarapari, Espírito Santo, Brasil

Emails: {clebeson.canuto, leonardo.as.ufes, mendonca.felippe, rodolfo.picoreti, jlasam001}@gmail.com, raquel@ele.ufs.br

Abstract— Static and dynamic gestures are considered important tools for human-machine interaction. Although being more complex, dynamic gestures are preferred because they are considered more natural. Some works try to perform dynamic gesture recognition using multimodal information, obtained from different sensors. However, most environments have only cameras installed (used for surveillance and monitoring), since other sensors typically have a limited range. Therefore, to perform gesture recognition using only visual information appears as a very interesting alternative. Such approach may be used in less sophisticated and more common environments. Thus, in this work we propose a method for dynamic gesture recognition based only on color. The applied technique represents temporal information as spatial information, and the "transfer learning" method is also used to accelerate the convergence of the model and to obtain better results. Evaluation was done using 3579 gestures from the *Montalbano gesture dataset*, classified as 20 different classes. An accuracy of 83.10% was obtained, with 65% of the gestures reaching more than 80% of accuracy. This result shows the proposed approach has an adequate performance and may be improved to be used in future human-machine interaction tasks.

Keywords— Dynamic Gestures, Human-Machine Interation, Transfer Learning, Deep Learning, Visual Information

Resumo— Gestos estáticos e dinâmicos são considerados ferramentas importantes para a interação homem-máquina. Mesmo sendo mais complexos, gestos dinâmicos são preferidos por serem considerados mais naturais. Muitos trabalhos buscam reconhecer gestos dinâmicos utilizando informações multimodais, capturadas com mais de um tipo de sensor. Entretanto, a maioria dos locais possuem apenas câmeras instaladas (para vigilância e monitoramento), já que outros sensores normalmente têm alcance limitado. Assim, reconhecer gestos usando apenas informações visuais pode ser uma alternativa muito interessante, permitindo-se usar tal abordagem em ambientes menos sofisticados e mais comuns. Por isso, neste trabalho é proposto um reconhecedor de gestos dinâmicos baseado apenas em cor, onde a técnica aplicada representa informações temporais como informações espaciais. Usa-se ainda o método de *trasfer learning* a fim de se acelerar a convergência do modelo e se obter melhores resultados. A avaliação do método foi feita usando 3579 gestos, retirados do banco de dados *Montalbano gesture dataset*, e distribuídos em 20 classes distintas. Como resultado, obteve-se uma acurácia de 83,10%, sendo que 65% dos gestos alcançaram mais de 80% de acurácia. Isso mostra que a abordagem proposta tem desempenho adequado, podendo ainda ser melhorada, para um uso futuro em tarefas de interação homem-máquina.

Palavras-chave— Gestos Dinâmicos, Interação Homem-Máquina, Transferência de Aprendizado, Aprendizado Profundo, Informação Visual

1 Introdução

Devido à importância dos gestos para a interação homem-máquina (HMI - do inglês *Human-Machine Interaction*), há décadas vários estudos têm concentrado esforços em reconhecedores cada vez mais eficazes. Uma lista dos principais reconhecedores utilizados pode ser vista em (Mitra and Acharya, 2007; Jaliya et al., 2016), sendo que em (Saikia and Saharia, 2016) mostram-se apenas aqueles destinados ao reconhecimento de gestos dinâmicos.

Dentre os vários métodos, aqueles que usam mais de uma informação (cor em formato RGB, profundidade, esqueleto, e outras), também conhecidos como multimodais, estão obtendo um melhor desempenho (Neverova et al., 2016). Isso porque informações como profundidade e esqueleto complementam de maneira positiva a informação de cor, o que facilita a separação das classes pelo reconhecedor (Escalera et al., 2017).

Entretanto, o uso de reconhecedores multimodais tem estado normalmente restrito a ambientes específicos, pois para se adquirir informações de profundidade, assim como outras informações extras ao RGB, é necessário que os ambientes interacionais possuam sensores como por exemplo *Microsoft Kinect* [1], *Asus Xtion Pro* [2], *Intel Realsense* [3]. Isso gera uma dependência negativa para a interação, pois muitos ambientes públicos ou privados estão comumente equipados apenas com câmeras, como por exemplo câmeras de monitoramento, não suportando inicialmente reconhecedores multimodais. Porém essa restrição tem motivado o estudo

[1] https://msdn.microsoft.com/en-us/library/hh438998.aspx
[2] https://www.asus.com/3D-Sensor/Xtion_PRO/
[3] https://click.intel.com/realsense.html

e desenvolvimento de métodos de reconhecimento baseados apenas em informações visuais, que buscam extrair das imagens não só cor, mas também informação de profundidade e até mesmo posição de juntas e esqueletos. Neste caso, com mais dados disponíveis, seria possível se usufruir das vantagens de um reconhecedor multimodal, apesar de se usar apenas um tipo de sensor.

Dos trabalhos que utilizam apenas imagens como única fonte de informação, são poucos os que obtém resultados significativos, a exemplo de (Barros et al., 2014; Escalera et al., 2017). Mesmo assim, é possível observar que grande parte deles utilizam vocabulários de gestos próprios e significativamente diferentes entre si. O que facilita muito o reconhecimento. Dessa forma, objetivando um maior alcance da HMI, faz-se necessário estudos sobre novas técnicas capazes de reconhecer gestos dinâmicos baseando-se apenas em informações visuais.

Nos últimos anos o aprendizado profundo (do inglês *deep learning*) tem alcançado o estado da arte de vários problemas dentro das áreas de processamento de imagens e visão computacional. No entanto, mesmo com todo esse avanço, nem todos os problemas são fáceis de serem resolvidos, haja vista a complexidade envolvida e a alta dependência de equipamentos de ponta para o treinamento dos modelos de aprendizagem (LeCun et al., 2015). Sendo o reconhecimento de gestos dinâmicos, um deles.

Dentre as arquiteturas mais utilizadas, as Redes Neurais Convolucionais - CNNs (do inglês *Convolutional Neural Networks*) são de longe uma das mais utilizadas. Esse tipo de rede tem propriedade de compartilhamento dos pesos, o que possibilita o entendimento das operações entre pesos e dados de entrada como sendo operações de convolução. Dessa maneira, considera-se que as CNNs possuem vários filtros que podem ser treinados para extrair características específicas dos dados.

Segundo Kumar (2017), a inicialização dos pesos de uma rede convolucional influencia diretamente na sua capacidade de aprendizagem (convergência). Sendo assim, escolher os pesos iniciais é uma etapa de importância para obtenção de bons resultados no reconhecimento. No entanto, essa tarefa ainda é muito subjetiva, e depende do conhecimento e experiência do criador do modelo. Não obstante, para problemas que compartilham similaridades, os pesos iniciais podem ser aqueles obtidos por meio de um treinamento prévio em outro banco de dados, o que também é conhecido como *Transfer Learning*. Essa técnica consiste na reutilização dos pesos já treinados para poder inicializar um novo modelo. Dessa maneira, a inicialização é bem mais efetiva e o modelo pode convergir mais rápido.

Uma vez que, normalmente, o reconhecimento dos gestos dinâmicos por meio do *deep learning* requer convoluções em três dimensões para poder capturar as suas informações temporais, Barros et al. (2014) utiliza uma técnica capaz de condensar a informação temporal dos gestos em apenas uma imagem RGB, utilizando para isso o histórico do movimento capturado por meio da soma dos módulos das diferenças entre imagens consecutivas. Com isso, o problema de reconhecimento de gestos em vídeos passa a ser em imagens, o que é algo muito interessante do ponto de vista do tempo de processamento e da necessidade de complexidade exigida pela CNN a ser utilizada. Devido à forma da imagem resultante, neste trabalho, essa técnica será mencionada como representação estrela, ou apenas estrela.

Sendo assim, este trabalho traz como proposta principal um reconhecedor de gestos dinâmicos baseados apenas em informação visual, usando a representação RGB de um vídeo em uma imagem e uma rede CNN para a tarefa de reconhecimento dos mesmos. Tal rede fará uso da técnica de *transfer learning* a fim de acelerar o treinamento, bem como alcançar melhores resultados. Portanto, para melhor explanar sobre a proposta, o artigo está estruturado da seguinte maneira: *i*) introdução e motivação para o trabalho; *ii*) explicação sobre a técnica de representação estrela utilizada para caracterizar a informação temporal de cada gesto; *iii*) definições e explicações sobre a técnica de *transfer learning*; *iv*) descrição da proposta; *v*) os experimentos e resultados; e, por fim, *vi*) as conclusões e os trabalhos futuros.

2 Antecedentes

Nesta seção serão descritos os conceitos usados no decorrer do artigo, especificamente: a representação estrela, que permite condensar a informação temporal de um vídeo em uma representação espacial; e *transfer learning*, que permite utilizar redes previamente treinadas na resolução de outros tipos de problemas, apresentando também uma definição matemática para a mesma.

2.1 A representação estrela

A primeira etapa do trabalho tem como objetivo representar o gesto capturado em um vídeo na forma de uma imagem RGB. Para isso, foi utilizada a representação descrita em (Barros et al., 2014), o qual será referenciado neste trabalho como estrela.

Tal representação recebe N *frames* de um vídeo contendo o gesto, e representa o movimento do gesto baseado na soma das diferenças entre *frames* consecutivos representados em escala de cinza. Especificamente, a cada novo *frame* em escala de cinza, a diferença entre ele e o anterior é calculada, e seu resultado é somado ao resultado do par anterior. Para que valores negativos não anulem positivos, utiliza-se o operador de módulo antes da

soma, como pode ser visto na Equação 1

$$M = \sum_{i=2}^{N} |\mathbf{F}_{i-1} - \mathbf{F}_i|, \quad (1)$$

onde: M é a imagem resultante; $\mathbf{F}_i$ o i-ésimo *frame* do vídeo de N *frames* contendo o gesto; e $|\bullet|$ é o operador módulo.

A representação estrela gera uma imagem de 3 canais, onde o primeiro canal é a matriz M, o segundo e o terceiro canal são os resultados de filtrar a matriz M com as duas máscaras de Sobel (Gonzalez and Woods, 2000).

De acordo com os autores, os canais relacionados às máscaras de Sobel permitem uma melhor discriminação do tipo de movimento presente em M. Entretanto, ao usar uma rede CNN, existe a possibilidade de que as camadas convolucionais da rede assumam o papel desse tipo de máscara na etapa de treinamento, caso o processo de otimização conclua ser importante. Dessa maneira, este trabalho não utilizará os canais relacionados a tais máscaras.

Na Figura 1 pode ser vista a representação estrela RGB aplicada a um vídeo contendo o gesto "basta", retirado do banco de dados utilizado nos experimentos deste trabalho, o qual será descrito em detalhes na Seção 4.1.

Figura 1: Representação estrela do gesto "basta".

2.2 Transfer Learning

Uma rede CNN é similar a uma rede neural, contendo camadas de neurônios que realizam operações a partir dos resultados de camadas anteriores. Os parâmetros de tais operações são conhecidos como pesos e ajustados em uma etapa de treinamento via um algoritmo de otimização, conhecido como *Back Propagation*. Nesse sentido, a inicialização dos pesos de uma rede CNN pode influenciar diretamente na sua capacidade de treinamento. Uma má inicialização dos pesos pode fazer com que o algoritmo de otimização inicie em uma região do domínio onde existam vários mínimos locais.

Desse modo, ao invés de inicializar os pesos com valores aleatórios, é melhor utilizar valores que já tenham sido treinados com outro banco de dados. Tal procedimento recebe o nome de *Transfer Learning* (transferência de aprendizado). É importante destacar que a técnica *Transfer Learning* vai além da inicialização dos pesos, permitindo também o reaproveitamento da arquitetura da rede original. Isso possibilita mais rapidez na implementação de modelos de aprendizagem que utilizam arquiteturas de *deep learning*.

Como mostrado em (Karpathy, 2017), existem três casos de importância na aplicação de *transfer learning*, a saber:

1. *Quando o novo banco de dados é grande e semelhante ao utilizado na rede original já treinada.* Aqui, pode-se utilizar os pesos da rede original sem a necessidade de efetuar uma etapa de treinamento.

2. *Quando o novo banco de dados é pequeno, mas semelhante ao utilizado na rede original já treinada.* Neste caso, os pesos da rede são inicializados com os valores dos pesos da rede original, e usando o novo banco, deve-se efetuar uma etapa de treinamento, de maneira a adequar os pesos ao problema trazido pelo novo banco. A essa técnica dá-se o nome de ajuste fino (do inglês *Fine Tunning*).

3. *Quando a novo banco de dados é diferente do utilizado na rede original previamente treinada.* Nesse caso, independente do seu tamanho, deve-se efetuar o *fine tunning*.

Uma característica interessante das redes CNN aplicadas a imagens, é que as primeiras camadas são responsáveis por descrever bordas, as intermediárias descrevem formas e as finais descrevem padrões de mais alto nível, também conhecidas como camadas semânticas. Sendo assim, quando o banco de dados difere do utilizado na etapa de treinamento da rede (ou seja, existe a necessidade de *fine tunning*), as camadas iniciais e intermediárias costumam trazer melhores resultados para o reconhecimento, pois selecionam características mais gerais.

Outro ponto interessante sobre *transfer learning*, é que comumente pode-se reutilizar apenas a parte convolucional da rede e não seu classificador, uma vez que pode-se optar por uma rede totalmente conectada com um número diferente de neurônios. Ou mesmo utilizar um outro classificador, fazendo com que a rede reutilizada sirva apenas como um extrator de características.

2.2.1 Definição matemática

A fim de melhorar o entendimento sobre o *transfer learning*, uma definição matemática do método de aprendizagem é dada a seguir.

Em geral, um domínio $\mathcal{D}$ consiste de dois componentes: um espaço de características $\mathcal{X}$; uma distribuição de probabilidade marginal $P(\mathbf{x})$, onde $\mathbf{x} = \{x_1, ..., x_n\} \in \mathcal{X}$. Agora, considerando um domínio específico $\mathcal{D}$, uma tarefa $\mathcal{T}$ será definida por duas componentes: um espaço de rótulos $\mathcal{Y}$;

uma função preditiva objetivo $f(\bullet)$, que não é observada, mas pode ser aprendida pelos dados do conjunto de treinamento; sendo, que o conjunto de treinamento consiste em um conjunto de pares $\{\mathbf{x}_k, \mathbf{y}_k\}$, onde $\mathbf{x}_k \in \mathcal{X}$ e $\mathbf{y}_k \in \mathcal{Y}$. A função $f(\bullet)$ pode ser usada para predizer o rótulo correspondente $f(\mathbf{x})$ de uma nova instância $\mathbf{x} \in \mathcal{X}$. De um ponto de vista probabilístico $f(\mathbf{x})$ pode ser interpretada como $P(\mathbf{y}|\mathbf{x})$.

Para um melhor entendimento da notação definida acima, suponha-se o problema de classificação de imagens, então: em relação ao domínio $\mathcal{D}$: $\mathcal{X}$ é o espaço de todas as imagens possíveis; $\mathbf{x}$ é uma imagem particular; x_i é o i-ésimo vetor ou tensor de características correspondente à i-ésima imagem; n é o número de pixels ou vetores de características em $\mathbf{x}$. Em relação à tarefa $\mathcal{T}$: $\mathcal{Y}$ é o conjunto de rótulos e nesse caso contém os rótulos pelos quais devem ser classificadas as imagens e $\mathbf{y}_k$ assume o valor de um desses rótulos.

Pelas definições acima, tem-se: um domínio $\mathcal{D} = \{\mathcal{X}, P(\mathbf{x})\}$ e uma tarefa $\mathcal{T} = \{\mathcal{Y}, f(\bullet)\}$. Ao considerar uma tarefa fonte e destino, tem-se: O *conjunto de dados do domínio fonte* é definido como o conjunto de pares $\{(\mathbf{x}_{S_1}, \mathbf{y}_{S_1}), ..., (\mathbf{x}_{S_{n_S}}, \mathbf{y}_{S_{n_S}})\}$, onde $\mathbf{x}_{S_k} \in \mathcal{X}_S$ e $\mathbf{y}_{S_k} \in \mathcal{Y}_S$; o *conjunto de dados do domínio destino* é definido como o conjunto de pares $\{(\mathbf{x}_{T_1}, \mathbf{y}_{T_1}), ..., (\mathbf{x}_{T_{n_T}}, \mathbf{y}_{T_{n_T}})\}$, onde, $\mathbf{x}_{T_k} \in \mathcal{X}_T$ e $\mathbf{y}_{T_k} \in \mathcal{Y}_T$. Como é indicado em (Pan and Yang, 2010), na maioria dos casos

$$0 \leq n_T \ll n_S.$$

Seja: um domínio fonte $\mathcal{D}_S$ e a tarefa a aprender $\mathcal{T}_S$; um domínio destino $\mathcal{D}_T$ e tarefa a aprender $\mathcal{T}_T$. Então, o objetivo de *Transfer Learning* é ajudar no aprendizado da função preditiva $f_T(\bullet)$ em $\mathcal{D}_T$ usando o conhecimento em $\mathcal{D}_S$ e $\mathcal{T}_S$, sendo $\mathcal{D}_S \neq \mathcal{D}_T$ e/ou $\mathcal{T}_S \neq \mathcal{T}_T$.

3 Proposta

Nesta seção será descrita a técnica proposta para reconhecimento de gestos dinâmicos, a qual está baseada em duas principais etapas:

- **Pré-processamento**: cada vídeo de entrada é condensado de maneira a formar uma imagem RGB usando uma versão modificada da representação estrela. Cada imagem resultante representa as informações temporais do seu respectivo gesto apenas como informações espaciais.
- **Classificação**: usando o *transfer learning* de uma rede CNN do tipo VGG16 é treinado um classificador de gestos dinâmicos que tem como entrada a imagem RGB da etapa de pré-processamento, e como saída uma das possíveis classes à qual pertence o gesto.

Ambas etapas são explicadas a seguir.

3.1 *Pré-processamento*

Com a finalidade de melhorar a representação da informação temporal presente no vídeo, resolveu-se modificar o cálculo da representação estrela. Desse modo, cada vídeo contendo o gesto completo é divido em três partes. Sendo que cada imagem dos três subvídeos, por estarem no espaço de cor RGB, são convertidas para tons de cinza e o procedimento estrela é calculado, como descrito pela Equação 1. Com isso, cada vídeo é representado por uma imagem RGB. De modo que o canal R contém a matriz M calculada a partir da primeira parte do vídeo, o canal G a partir da parte central e o canal B a partir da parte final do vídeo, assim como representado na Figura 2.

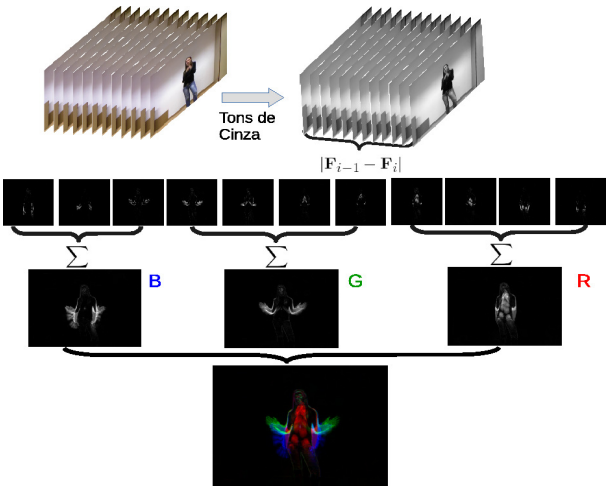

Figura 2: Processo de aplicação da representação estrela modificada para um vídeo.

3.2 *Classificação*

O classificador de gestos dinâmicos está baseado na rede CNN VGG16 (Simonyan and Zisserman, 2014). Essa rede é especializada no problema de classificação de imagens, sendo treinada sobre o banco de dados *ImageNet*, disponibilizado na competição ILSVRC-2014, a qual contém mais de 1,2 milhões de imagens distribuídas entre 1000 categorias distintas (Russakovsky et al., 2015).

A Figura 3 contém as legendas para as Figuras 4 e 5 apresentadas adiante. Na Figura 4 pode-se ver a arquitetura da VGG16 original, a qual possui 13 camadas convolucionais ativadas por uma função ReLU (Nair and Hinton, 2010), seguidas de uma camada de *flatten* (responsável por transformar cada uma das matrizes resultantes da última camada de convolução em um único vetor os dados), uma camada totalmente conectada de 4096 neurônios e, finalmente, uma camada de saída de 1000 neurônios com a função de ativação *softmax*, onde cada neurônio de saída representa uma das classes presente no banco de dados *ImageNet*.

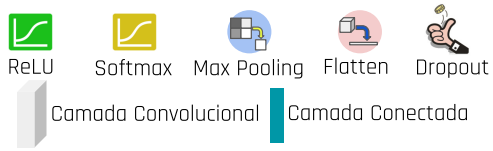

Figura 3: Legenda para as Figuras 4 e 5

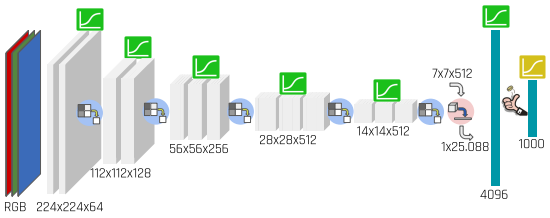

Figura 4: Arquitetura da rede VGG16 original

A Figura 5 traz o classificador proposto neste trabalho a partir da arquitetura original da VGG16. Apenas as suas sete primeiras camadas convolucionais foram utilizadas, uma vez que as últimas camadas da VGG16 foram treinadas para reconhecer características mais elaboradas do problema original, diferentes das necessárias para o reconhecimento dos gestos dinâmicos aqui abordados.

A saída da sétima camada convolucional da VGG16 foi reduzida, utilizando-se um operador de *maxpooling*. Dessa maneira, o resultado gerado pôde ser conectado a um operador de *flatten*.

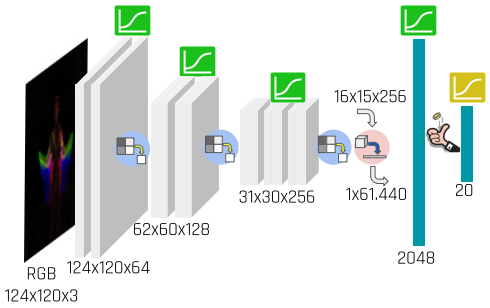

Figura 5: Arquitetura da rede VGG16

O vetor resultante da camada de *flatten* serve como entrada para uma rede alimentada adiante (do inglês *feedforward*) de 2048 neurônios completamente conectados, seguida pela camada de saída composta de 20 neurônios com função de ativação *softmax*. Essa última camada é responsável por predizer a qual das 20 classes de gestos (quantidade classes de gestos contidos na base utilizada - Seção 4.1) pertence a imagem de entrada.

Considerando os três casos de uso de *transfer learning* comentados na Seção 2.2, para o problema em questão, foi necessário realizar o *Fine Tunning* sobre o classificador proposto, já que a VGG16 original foi treinada pra classificar as imagens do *ImageNet*, diferentes das imagens a serem fornecidas após o pré-processamento dos vídeos.

Com a arquitetura proposta, já era possível treinar o modelo e realizar os testes, porém, sabendo-se ser alto o tempo de convergência de uma CNN para um banco de dados relativamente grande, resolveu-se partir para a utilização de mecanismos que visassem reduzi-lo. Assim, segundo LeCun et al. (1998), isso pode ser alcançado diminuindo a variação dos pesos da rede neural (média em torno de zero). No entanto, a grande variação dos dados de entrada faz com que isso não ocorra, problema denominado de deslocamento interno da covariância (do inglês - *internal covariate shift*). Desse modo, ainda segundo LeCun et al. (1998), a solução para isso está na normalização dos dados. Nesse sentido, uma das técnicas mais utilizadas para o *deep learning* é o *batch normalization*, descrito por Ioffe and Szegedy (2015). Essa técnica é uma adaptação do que foi dito por (LeCun et al., 1998), porém, empregada em treinamentos baseados em *mini batchs*.

Assim, em cada uma das duas camadas totalmente conectadas, antes da aplicação da função de ativação, foi aplicado o *batch normalization*.

Uma vez pensado na forma de normalização, é primordial utilizar uma técnica adequada de regularização, a fim de poder diminuir a capacidade de sobreajuste (do inglês *overfitting*) da rede. Isso é comum quando se utiliza as redes neurais profundas, devido principalmente, a seu alto número de parâmetros. Portanto, após vários testes, utilizando os regularizadores $L1$, $L2$ e *dropout* (Srivastava et al., 2014), o $L2$ foi o que trouxe melhores resultados. Com isso, o algoritmo otimizador não recebia apenas o erro obtido na etapa de propagação e sim esse erro acrescido de uma ponderação da norma $L2$ calculada sobre todos pesos da rede.

É importante salientar que, a depender da arquitetura da rede a ser treinada (e/ou do banco de treino e teste), as escolhas das técnicas de normalização e de regularização aqui utilizadas podem não trazer bons resultados. Desse modo, aconselha-se realizar testes objetivando encontrar o par normalizador-regularizador que traga melhor resultado para o problema a ser resolvido.

4 Experimentos e resultados

Nesta seção serão descritos o banco de dados usado nos experimentos; as características de implementação de software e hardware; e os resultados obtidos na avaliação da abordagem proposta em função da métrica de desempenho conhecida como acurácia.

4.1 O banco de dados utilizado

Para a realização dos experimentos, foi utilizado o banco de dados *Montalbano gesture dataset*, fornecido na competição *Chalearn looking at people* - 2013 (Escalera et al., 2013). A base foi capturada utilizando-se um sensor tipo *Kinect* 360. Dessa forma, cada gesto possui informações multimodais: RGB, profundidade, esqueleto, silhueta e áudio. Na competição, cada candidato poderia utilizar

qualquer um dos dados fornecidos para poder reconhecer em um vídeo contínuo qual a sequência de gestos (entre 8 e 20 por vídeo) estava sendo realizada.

Segundo os autores, esse banco consiste de 13858 gestos Italianos culturais/antropológicos distribuídos em 20 tipos distintos. São 7754 vídeos para treino, 3362 para validação e 2742 para teste, lembrando que os vídeos de teste contém vários gestos em sequência e não apenas um.

No entanto, como o objetivo deste trabalho não foi o de participar da competição, mas apenas reconhecer gestos, os mesmos foram segmentados em vários vídeos de acordo com a rotulação fornecida. Ao final, cada vídeo gerado continha apenas um gesto dinâmico. O problema, agora, não seria mais o de reconhecer uma sequência de gestos em um vídeo, e sim classificar os vídeos entre 20 classes distintas de gestos dinâmicos. Com isso, obtiveram-se 11179 gestos, dos quais 6847 são vídeos de treino, 2753 de validação e 3579 de teste.

Apesar de conterem informação multimodal, quando se pensa no uso de gestos como meio de comunicação entre pessoas e dispositivos em ambientes de uma forma geral, é mais interessante que seja considerada apenas a informação RGB dos dados. Conforme mencionado anteriormente, os sensores mais facilmente encontrados em diversos ambientes são câmeras de monitoramento. Sendo assim, neste trabalho buscou-se reconhecer os gestos dinâmicos do banco utilizando apenas seus dados visuais.

A escolha desse banco de dados se deu pelo fato de ser um dos maiores bancos que se têm na atualidade para o problema de reconhecimento de gestos dinâmicos. O que é uma premissa básica para a aplicação de *deep learning*. Outro motivo, mencionado por Tsironi et al. (2017), é que bancos de gestos dinâmicos baseadas em RGB são muito reduzidos, sendo que, aqueles que estão disponíveis são capturados focando apenas as mãos e não o corpo inteiro das pessoas. Os 20 tipos de gestos contidos no banco podem ser vistos na Figura 6.

Após segmentados todos os vídeos, foi aplicada a técnica de estrela modificada, como proposto na Seção 3.2. Todo o processamento foi feito utilizando as bibliotecas *OpenCV* e *Numpy* da linguagem Python.

4.2 Características de Implementação

As etapas de treinamento e teste da arquitetura proposta foram realizadas utilizando *Tensorflow*, *software* de código aberto desenvolvido pela *Google brain Team*[4], destinado à programação numérica utilizando programação baseada em fluxo de dados em grafos (Abadi et al., 2016). Além disso, a máquina utilizada nos experimentos possuía a seguinte configuração: (*i*) sistema operacional Linux,

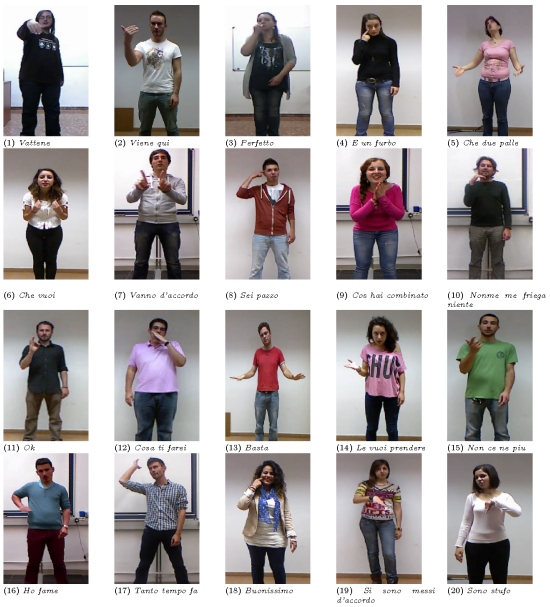

Figura 6: Os 20 gestos do banco de dados utilizado. Fonte: (Escalera et al., 2013)

distribuição Ubuntu Server 16.04; (*ii*) processador Intel Core i7-7700, 3.60GHz com 4 núcleos físicos; (*iii*) memória RAM de 16 GB; (*iv*) unidade de armazenamento de 1TB (disco rígido); (*v*) placa de vídeo Nvidia Geforce GTX 1080, com 8 GB de memória dedicada.

Como a capacidade da memória RAM não era suficiente para suportar a banco de dados por completo, durante a etapa de treinamento a taxa de *Swap* entre a memória e o disco rígido aumentava drasticamente. Com isso, uma vez que a unidade de armazenamento não era de estado sólido, o tempo de carga do próximo *batch* a ser treinado era considerado grande se comparado com a rapidez a qual a GPU o processava. Provocando uma ociosidade da mesma devido a essa espera. Sendo assim, foi realizado um processo de pré-carga dos dados utilizando as bibliotecas próprias do *Tensorflow*. Para isso, todas as imagens do conjunto de treino foram serializadas em um único arquivo com extensão *tfrecords*, utilizando o protocolo de serialização *Protobuf*[5]. Dessa forma, 7 *threads* ficaram responsáveis por carregar imagens em um *buffer* de tamanho pré-definido, enquanto que, paralelamente, a rede estava sendo treinada na GPU. Com o uso desse procedimento, a média de tempo de ociosidade da GPU caiu de cerca de 40% para menos de 10%, o que possibilitou acelerar ainda mais o treinamento.

Mesmo sendo menor que o banco de treino, os dados de teste também foram serializados e utilizados da mesma maneira que no treinamento.

Finalmente, na etapa de treino foram utilizados os seguintes hiper-parâmetros: *batch size* de 96; número de épocas de treinamento igual a 3000;

[4] `https://research.google.com/teams/brain/`

[5] https://github.com/google/protobuf

learning rate de 5e−3 decaindo 1% a cada época e 5e−2 como escala do regularizador $L2$.

4.3 Avaliação dos resultados

Por tratar-se de um problema de classificação multiclasse, a avaliação adotada foi a métrica de acurácia, calculada como sendo o percentual de gestos classificados corretamente.

Para melhor visualizar o comportamento do classificador proposto, o resultado é apresentado na forma de uma matriz de confusão normalizada (Figura 7), a qual possui uma linha para cada classe verdadeira e uma coluna para cada classe predita. Como característica, cada célula da matriz (linha r, coluna c) indica qual o percentual de gestos pertencente à classe r foi classificado como pertencente à classe c. Dessa maneira, a acurácia do modelo pode ser calculada como a média da diagonal principal da tabela em questão.

Com experimentos realizados, após decorridos 2348 épocas, o modelo atingiu uma média de 10 *batchs* consecutivos com mais de 99% de acurácia, condição de parada antecipada para o treinamento. Dessa maneira, o teste foi realizado, obtendo uma acurácia de $83,10\%$ de acerto. O tempo necessário para o treinamento foi de 12 horas (utilizando o computador antes especificado). É importante salientar que mesmo parecendo muito, sem o *batch normalization*, o treinamento durava mais de 28 horas. Assim, houve um ganho significativo de desempenho no treinamento após a normalização. Outro ponto é que 12h é um tempo razoável ao considerar a complexidade do problema e o tamanho do banco de treinamento.

Como é possível observar, a grande maioria dos gestos obteve uma acurácia de acerto acima de 80%, sendo que os gestos "vattene", "noncenepio", "basta", "sonostufo" , "freganiente" e "messidaccordo" atingiram mais de 90%. Apenas em poucos casos, como em "buonissimo", "cosatifarei", "ok", "chevai", "fame" e "prendere", o valor de acurácia foi menor que 80%. Mesmo assim, o pior caso ainda alcançou 67%, que corresponde ao gesto "ok".

Para comprovar a qualidade do resultado, a arquitetura descrita em (Barros et al., 2014) foi implementada e aplicada ao banco de dados aqui utilizado. No entanto, foi aplicado a representação estrela assim como descrito pelos autores. Como resultado, obteve-se uma acurácia de $61,75\%$, valor $25,69\%$ inferior ao obtido pela abordagem aqui proposta, e mais baixo que o pior valor obtido. Tal resultado já era esperado, uma vez que o *tranfer learning* tem se mostrado uma técnica muito eficiente para as tarefas de reconhecimento em imagens, sem falar que a modificação na técnica estrela aqui apresentada, traz uma maior representatividade das informações temporais dos gestos dinâmicos.

Outro aspecto que mostra a qualidade da proposta, é a obtenção do resultado aqui apresentado apesar da complexidade do problema. Apenas a informação de RGB é utilizada e, dentre as 20 classes do banco de dados, algumas não são facilmente separáveis. Isso leva à conclusão de que o resultado obtido foi bastante satisfatório.

5 Conclusões

O objetivo principal deste trabalho foi propor uma arquitetura para o reconhecimento de gestos dinâmicos que usa unicamente informações de cor. Para isso foi feita uma modificação da representação estrela, de maneira a poder caracterizar as informações temporais de cada gesto apenas em uma imagem RGB. Utilizou-se ainda da técnica de *transfer learning* aplicada à rede CNN VGG16, para poder reconhecer o gesto presente nas imagens RGB geradas a partir de cada vídeo. Como resultado, usando o banco de dados *Montalbano gesture dataset*, obteve-se uma acurácia de $83,10\%$ na classificação de 3579 gestos dinâmicos em 20 classes distintas.

A partir da matriz de confusão foi observado que 13 dos 20 gestos ultrapassaram os 80%, e 6 desses ficaram acima de 90%. Mesmo para os gestos com percentual de acerto menor, o pior caso ainda obteve 67%, o que é interessante ao considerar-se a complexidade do problema.

Como trabalhos futuros: serão realizados estudos a fim de testar o *transfer learning* com outras arquiteturas CNNs; além de procurar aprimorar a representação das informações dos gestos a serem reconhecidos. Sendo assim, buscar-se-á melhorar ainda mais o método estrela ou mesmo utilizar outros métodos de representação em conjunto.

Agradecimentos

À CAPES e ao CNPq pelo suporte financeiro dado, respectivamente, através da Bolsa de Doutorado concedida ao primeiro autor e de Mestrado ao terceiro.

Referências

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al. (2016). Tensorflow: A system for large-scale machine learning., *OSDI*, Vol. 16, pp. 265–283.

Barros, P., Parisi, G. I., Jirak, D. and Wermter, S. (2014). Real-time gesture recognition using a humanoid robot with a deep neural architecture, *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, IEEE, pp. 646–651.

Escalera, S., Athitsos, V. and Guyon, I. (2017). Challenges in multi-modal gesture recognition, *Gesture Recognition*, Springer, pp. 1–60.

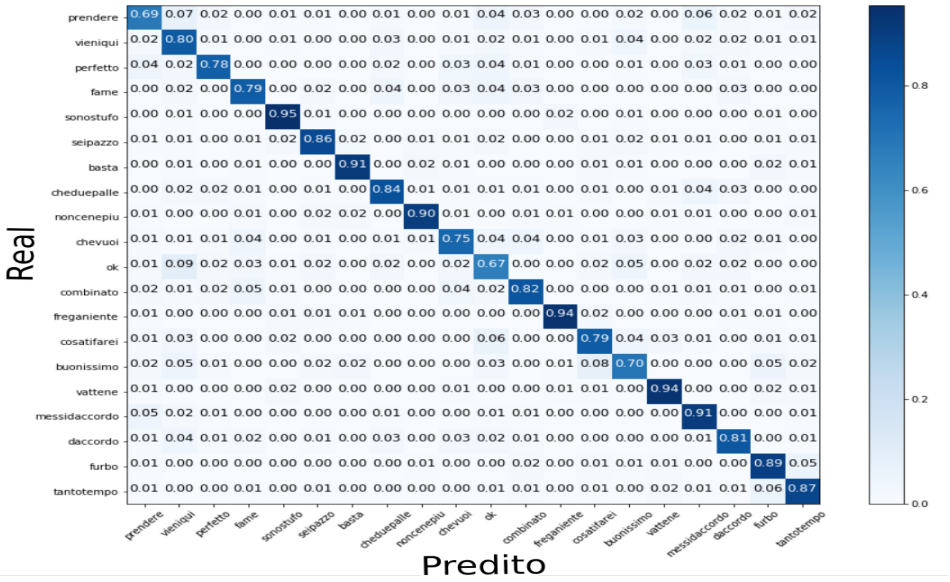

Figura 7: Matriz de confusão das predições feitas pelo modelo proposto treinado.

Escalera, S., Gonzàlez, J., Baró, X., Reyes, M., Lopes, O., Guyon, I., Athitsos, V. and Escalante, H. (2013). Multi-modal gesture recognition challenge 2013: Dataset and results, *Proceedings of the 15th ACM on International conference on multimodal interaction*, ACM, pp. 445–452.

Gonzalez, R. C. and Woods, R. E. (2000). *Processamento de imagens digitais*, Edgard Blucher.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift, *arXiv preprint arXiv:1502.03167* .

Jaliya, U., Thakore, D. and Kawdiya, D. (2016). A survey on hand gesture recognition.

Karpathy, A. (2017). Convolutional Neural Networks for Visual Recognition.

Kumar, S. K. (2017). On weight initialization in deep neural networks, *arXiv preprint arXiv:1704.08863* .

LeCun, Y., Bengio, Y. and Hinton, G. (2015). Deep learning, *nature* **521**(7553): 436.

LeCun, Y., Bottou, L., Orr, G. B. and Müller, K.-R. (1998). Efficient backprop, *Neural networks: Tricks of the trade*, Springer, pp. 9–50.

Mitra, S. and Acharya, T. (2007). Gesture recognition: A survey, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* **37**(3): 311–324.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines, *Proceedings of the 27th international conference on machine learning (ICML-10)*, pp. 807–814.

Neverova, N., Wolf, C., Taylor, G. and Nebout, F. (2016). Moddrop: adaptive multi-modal gesture recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence* **38**(8): 1692–1706.

Pan, S. J. and Yang, Q. (2010). A survey on transfer learning, *IEEE Transactions on knowledge and data engineering* **22**(10): 1345–1359.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M. et al. (2015). Imagenet large scale visual recognition challenge, *International Journal of Computer Vision* **115**(3): 211–252.

Saikia, S. and Saharia, S. (2016). A survey on vision-based dynamic gesture recognition, *International Journal of Computer Applications* **138**(1).

Simonyan, K. and Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition, *arXiv preprint arXiv:1409.1556* .

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research* **15**(1): 1929–1958.

Tsironi, E., Barros, P., Weber, C. and Wermter, S. (2017). An analysis of convolutional long short-term memory recurrent neural networks for gesture recognition, *Neurocomputing* **268**: 76–86.