Universidade da Beira Interior
Departamento de Informática

Análise Inteligente de Audiências em Ambientes Exteriores III: Caracterização/Validação de IDs

Elaborado por:

João Pedro da Cruz Brito, Nº 37880

Orientador:

Professor Doutor Hugo Proença

Covilhã, Junho de 2019

Agradecimentos

A conclusão deste trabalho não teria sido possível sem a imprescindível ajuda do meu orientador de projeto, o Professor Doutor Hugo Proença. Os seus conselhos e ensinamentos nortearam este projeto.

De seguida, quero manifestar o meu apreço pelo apoio e companheirismo demonstrados pelos meus colegas de curso e amigos próximos.

Por último, mas não menos importante, gostaria de agradecer o suporte incondicional prestado pela minha família durante o meu percurso académico.

Conteúdo

Lista de Figuras

Lista de Tabelas

Acrónimos

CNN *Convolutional Neural Network*

ILSVRC *ImageNet Large Scale Visual Recognition Challenge*

MLP *Multi-Layer Perceptron*

NGL *Núcleo Geniculado Lateral*

Capítulo 1

Introdução

1.1 Enquadramento

O presente documento e aplicações computacionais associadas enquadram-se na Unidade Curricular de Projeto, inserida no 3º ano da Licenciatura em Engenharia Informática da Universidade da Beira Interior, no ano letivo 2018/2019.

1.2 Motivação

A interação homem/máquina tem-se tornado uma constante no panorama mundial. Assim, a capacidade de um computador equipado com uma câmara (ou outro dispositivo equiparado) reconhecer objetos ou pessoas que estejam à sua frente e tomar ações em conformidade, é uma caraterística cada vez mais apetecível. Inspirada pela naturaza humana, a área de Visão Computacional está cada vez mais presente nas sociedades atuais.

1.3 Objetivos

Este trabalho propõem-se a usar métodos de classificação de imagens, nomeadamente uma *Convolutional Neural Network* (CNN), para resolver dois problemas:

1. Validar um módulo de *tracking* ao discriminar sequências de indivíduos diferentes de sequências de um mesmo indivíduo.

2. Fazer o reconhecimento facial de um indivíduo face a um *dataset* conhecido.

1.4 Organização do Documento

De modo a refletir o trabalho que foi feito, este documento encontra-se estruturado da seguinte forma:

1. O primeiro capítulo – **Introdução** – apresenta o enquadramento, motivação e objetivos do projeto, bem como a organização do presente documento.

2. O segundo capítulo – **Redes Neuronais Convolucionais** – descreve as bases teóricas de uma CNN, passando também pelo papel do Córtex Visual Primário no reconhecimento de imagens por parte do ser humano.

3. O terceiro capítulo – **Problema 1: Validação de Módulo de Tracking** – apresenta o primeiro grande objetivo deste projeto, o método explorado na sua concretização e os resultados obtidos.

4. O quarto capítulo – **Problema 2: Identificação de Indivíduos** – apresenta, numa estrutura semelhante ao terceiro capítulo, o segundo objetivo que se procurou atingir.

5. O quinto capítulo – **Conclusão e Trabalho Futuro** – resume os conhecimentos e aptidões obtidos através da realização deste trabalho, realçando, ainda, aspetos que poderiam ser melhorados e/ou acrescentados.

Capítulo 2

Modelos de Aprendizagem Profunda

2.1 Introdução

O nosso cérebro processa constantemente o ambiente que nos rodeia. Durante todo o processo de aprendizagem e crescimento por que o ser humano passa, o cérebro cria regras que lhe permitem, entre muitas outras coisas, classificar objetos de forma quase instantânea. Este processo começa de forma supervisionada (os nossos pais classificam objetos, inicialmente desconhecido, por nós) levando a que criemos parâmetros que permitirão a identificação autónoma de futuras instâncias de qualquer objeto.

Sabe-se que um neurónio recebe informação através das dendrites, processa o que recebeu e dispara um *output* adequado. Trata-se, claramente, de uma simplificação daquele que é talvez um dos mecanismos mais fascinantes que a Ciência estuda. Contudo, serviu de base para que em 1943 o neurocientista Warren McCulloch e o matemático Walter Pitts propusessem um neurónio artificial. A ideia base seria a mesma de um neurónio biológico, aliada à esperança de que muitas células simples seriam capazes de dar respostas complexas.

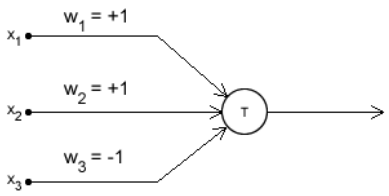


Figura 2.1: Modelo do neurónio artificial proposto por McCulloch e Pitts [1].

2.2 Córtex Visual Primário

Situado no Lobo Occipital, o Córtex Visual Primário (V1) é, a nível da visão, uma das regiões mais estudadas do cérebro. Sendo extremamente especializado na análise de objetos tanto em movimento como estáticos, possui um alto grau de reconhecimento de padrões.

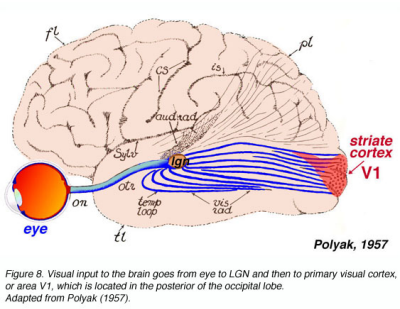


Figura 2.2: Fluxo de informação desde o olho até ao Córtex Visual Primário [2].

No decorrer do processamento de informação (figura 2.2), o *input* visual é obtido pela retina, com intervenção das suas células ganglionares, cones, bastonetes, células horizontais, células bipolares e células amácrinas. As terminações das células ganglionares agregam-se e surgem na parte posterior do olho já sob a forma do nervo ótico. A continuação deste nervo crucial para a Via Visual Primária, denominada de trato ótico, transporta a informação sensorial até ao *Núcleo Geniculado Lateral* (NGL) - localizado no Tálamo, um importante centro nervoso.
Algum processamento é realizado nesta secção, destacando-se as células M (*Magnocellular*), P (*Parvocellular*) e K (*Koniocellular*). Existem funções bem definidas e um grau de especialização diferenciador: os neurónios M são particularmente afetos à deteção de movimento; os neurónios P às caraterísticas espaciais (tamanho e forma) e os neurónios K a alguns aspetos da visão cromática.

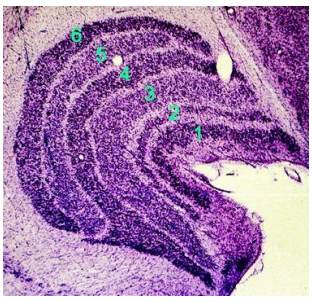


Figura 2.3: As seis camadas que compõe o NGL: 1-2 (células M); 3-6 (células P) e as células K situam-se em zonas intermédias [3].

Finalmente, esta informação filtrada e levemente tratada chega ao seu destino, o Córtex Visual Primário.
A partir dos estudos de Hubel e Weisel na década de 1960, foi possível perceber que os neurónios em V1 são excitados por caraterísticas tanto simples como complexas da imagem a ser analisada. Algumas células respondem melhor a arestas com um determinado ângulo (preferência de orientação), enquanto outras processam a informação vinda dos dois olhos de forma diferente (preferência ocular). Deste modo, a estrutura interna do Córtex Visual Primário é composta por colunas de neurónios com funções e reações a *inputs* semelhantes.
Acredita-se que outras áreas envolventes a V1 desempenham um papel importante no processamento dos estímulos visuais. Porém, existem ainda muitos mistérios por desvendar e que continuarão a motivar a comunidade científica.

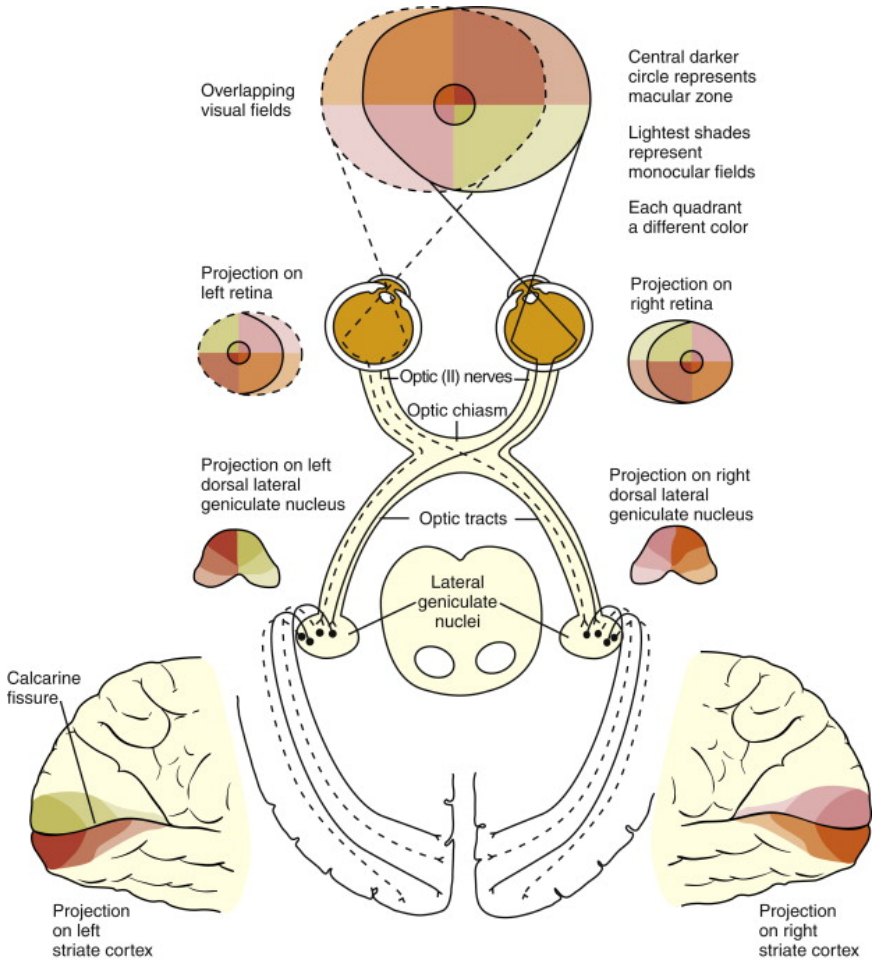


Figura 2.4: Vários componentes envolvidos no processamento de informação visual [4].

2.3 Redes Neuronais Convolucionais

Apresentada a estrutura biológica que inspirou o investimento nas Redes Neuronais artificiais, importa descrever os mecanismos que levam uma CNN a conseguir replicar (com as devidas diferenças) o processo anteriormente descrito.

Tipicamente, uma CNN tem as seguintes camadas: camada(s) convolucional(ais), camada(s) de *pooling* e camada(s) de classificação (camada(s) completamente ligada(s)).

2.3.1 Camada Convolucional

Esta camada procura detetar *features* de baixo nível, como arestas e curvas. Para tal, é usado um filtro (também chamado de *kernel*). Uma imagem tem sempre a forma NxMxC, sendo N o número de linhas, M o número de colunas e C o número de canais de cor. De modo a diminuir o tamanho potencialmente inviável do *input*, o *kernel* (nada mais que uma matriz) passa pela imagem um número permitido de vezes - numa matriz 5x5 e com passo/*stride* igual a 1, o *kernel* cobre-a nove vezes originando um mapa de ativação 3x3 (figura 2.5).
Note-se que outras etapas podem ser acrescentadas ao output da camada convolucional (por exemplo, a função de ativação ReLU promove não-linearidade, algo desejável quando se trabalha com imagens que não são lineares, por natureza)

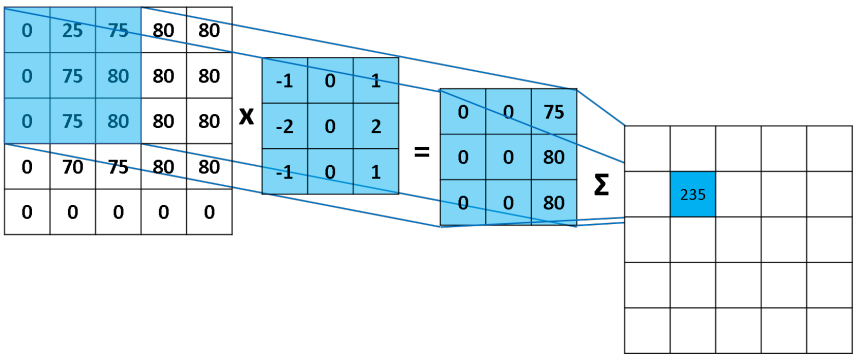


Figura 2.5: Operação de convolução entre uma matriz de entrada 5x5 e um *kernel* 3x3 [5].

Os valores do *kernel* são multiplicados com os da área abrangida na matriz inicial, sendo posteriormente somados (a matriz com todos estes números calculados contém as chamadas *convolved features*).

O exemplo retratado na figura 2.5 representa uma simplificação do processo de convolução. Na realidade, podem-se considerar situações adicionais:

1. No caso de uma imagem a cor RGB, existem três canais, pelo que a matriz de *input* tem profundidade igual a 3. O *kernel* é aplicada a cada canal e os três valores resultantes de cada canal são somados, dando, finalmente, o valor que entrará na matriz das *convolved features*.

2. De modo a extrair mais *features*, podem ser utilizados vários *kernels* com pesos diferentes. Assim, obtém-se um volume no fim desta camada, e não um *output* com profundidade igual a 1.

2.3.2 Camada de *Pooling*

Sendo aplicada ao *output* da etapa anterior, a camada de *pooling* tem a função de reduzir a dimensão espacial das *convolved features*, sem perder informação necessária à classificação. Esta camada é, ainda, capaz, de extrair *features* dominantes em certas regiões da imagem de entrada. Consideram-se dois tipos de *pooling*: *max pooling* e *average pooling*.
A técnica de *max pooling* opta por, a partir de uma região definida na matriz das *convolved features*, escolher o maior valor possível (figura 2.6). Apresenta a vantagem de reduzir possível ruído e combater o problema de *overfitting* - manifesta-se quando os resultados no conjunto de teste são muito bons, mas a mesma *performance* não se verifica nos exemplos de teste.
Numa abordagem alternativa, *average pooling* faz a média aritmética simples dos valores abrangidos pelo filtro escolhido, normalmente 2x2 e com passo/*stride* semelhante (figura 2.6).
Comparando as duas variantes, *max pooling* é a mais popular, sendo creditada como a melhor em termos de *performance*.

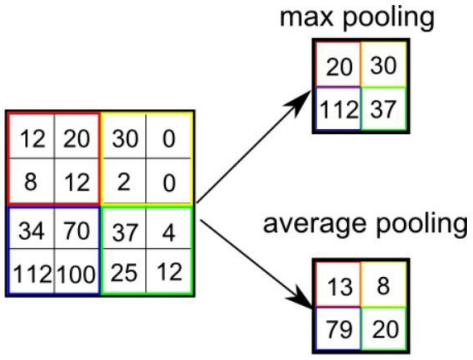


Figura 2.6: 2 tipos de *pooling* [6].

Dependendo da aplicação e *dataset* a analisar, podem ser usadas várias camadas convolucionais e de *pooling* intercaladas.

2.3.3 Camada Completamente Ligada

Em qualquer caso, o volume (matriz com profundidade) resultante da intercalação das camadas anteriormente mencionadas, será achatado na forma de um vetor coluna. Deste modo, é possível fornecer tal vetor, como *input*, a uma rede neuronal tradicional (sendo que os neurónios de cada camada estão ligados com todos os da camada seguinte). Pode-se dizer que as camadas anteriores extraíram *features* da imagem de entrada, enquanto que as camadas que se seguem classificam tudo o que foi obtido (dando mais importância a determinados padrões e formas que outros).

Na figura 2.7, os círculos amarelos representam o resultado da interpolação de "n"camadas de convolução e *pooling*. Os círculos azuis agem como uma rede neuronal (tipicamente um *Multi-Layer Perceptron* (MLP)) com tantas camadas escondidas quantas as necessárias para o problema a ser tratado. Finalmente, a última camada (vermelha), contém os neurónios ligados às classes existentes. É comum usar-se a função *softmax* para obter uma distribuição probabilística sobre os *outputs* finais da CNN.

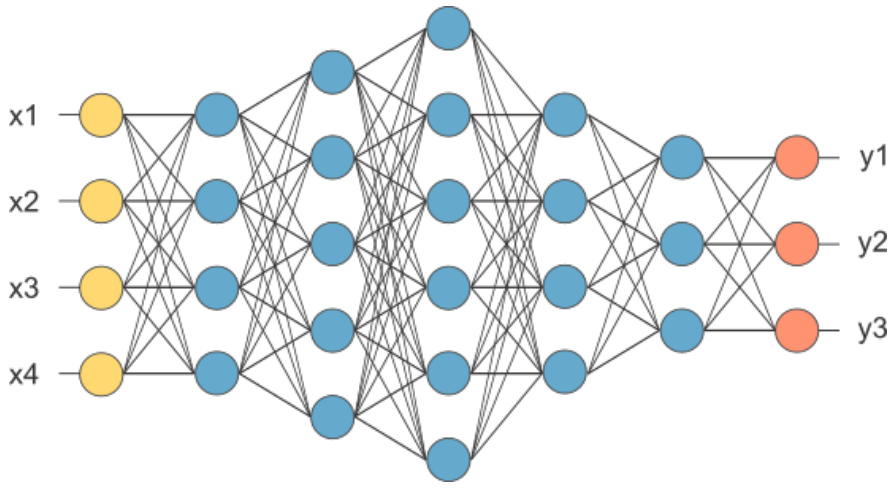


Figura 2.7: Visualização das camadas finais de uma CNN [7].

2.3.4 Função de *Loss*

Uma função de *loss* mede a performance de um modelo de classificação. No caso mais concreto dos problemas trabalhados, foi usada a função *Cross-Entropy*. Apresenta a seguinte fórmula:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p(y_i)) + (1-y_i)\log(1-p(y_i)) \tag{2.1}$$

Na fórmula 2.1, y_i representa a classe verdadeira de um dado ponto *i*, N o número de pontos/elementos a classificar e $p(y_i)$ a probabilidade de o ponto ser da classe 1. Numa análise à fórmula, se o ponto for da classe 1 (aplica-se em problemas binários, logo as classes são 1 e 0), o segundo termo anula-se e fica apenas $log(p(y_i))$. Relembrando a curva logarítmica, uma probabilidade alta faria o termo ser próximo de 0 (aproximando-se pelos valores negativos).
De forma inversa, um ponto de classe 0 anula o primeiro termo e resulta em $log(1-p(y_i))$. Como é de classe 0, a probabilidade de ser de classe 1 é, idealmente, baixa. Logo o valor dentro do logaritmo será próximo de 1 e a imagem respetiva, na função logarítmica, próxima de 0.
Por fim, é invertido o valor resultante do somatório (os logaritmos de cada ponto resultaram em valores negativos) e feita a média das entropias cruzadas obtidas ($\frac{1}{N}$).

2.3.5 Notas finais

Nos últimos 20 anos, várias arquiteturas para CNNs foram propostas, destacando-se as seguintes:

1. **LeNet-5**, proposta por Yann LeCun em 1998, propunha-se a classificar dígitos, escritos à mão, digitalizados com uma resolução de 32x32 em escala de cinza.

2. **AlexNet**, descrita por Alex Krizhevsky num *paper* publicado em 2012, venceu a *ImageNet Large Scale Visual Recognition Challenge* (ILSVRC) do mesmo ano com uma vantagem de mais de 10% para o segundo lugar. Conta com cinco camadas convolucionais (algumas seguidas de camadas de *pooling*) e três camadas completamente ligadas. Propõe, ainda, o uso da função de ativação ReLU. Esta arquitetura demonstrou como a profundidade de uma rede é crucial para o desempenho final, bem como, técnicas de aumento de dados (espelhar ou cortar imagens) e de *dropout* (remover neurónios aleatoriamente) ajudam a combater o problema de *overfitting*.

3. **GoogleNet/Inception**, vencedora da ILSVRC de 2014 com um erro top-5 de 6.67% (entrando no nível de performance do próprio ser humano). Conta com 22 camadas, vários módulos Inception (fazem várias convoluções e *pooling*) e reduz o número de parâmetros da AlexNet (60 milhões) para apenas 4 milhões.

4. **VGGNet**, perdeu em 2014 para a GoogleNet, sendo creditada pela capacidade de extração de *features*. Como desvantagem, contém 138 milhões de parâmetros.

5. **ResNet**, vencedora da ILSVRC de 2015, possui 152 camadas. Note-se que, mesmo assim, possui menor complexidade que a VGG.

Devido à reconhecida capacidade de classificação de imagens, foi usada uma CNN (mais concretamente, a arquitetura Inception) com vista a solucionar os dois problemas descritos nos capítulos seguintes.

Capítulo 3

Problema 1: Validação de Módulo de Tracking

3.1 Introdução

Durante um processo de *tracking* de qualquer objeto, erros podem acontecer. No caso específico abordado pelo presente projeto, a partir de imagens captadas por painéis de *outdoor*, *bounding boxes* foram colocadas ao redor das pessoas que passavam pelos locais abrangidos. IDs únicos foram igualmente atribuídos, constituindo, em teoria, a ação de *tracking*.
Assim, o primeiro desafio proposto passou por validar os IDs atribuídos aos diferentes indivíduos, tomando como entrada uma sequência de *patches* e avaliar se diz respeito ao mesmo indivíduos ou a indivíduos diferentes.

(a)

(b)

Figura 3.1: Exemplos de sequências respeitantes a um mesmo indivíduo (a) e a indivíduos diferentes (b).

3.2 Pré-processamento dos dados

Antes de passarem pela etapa de aprendizagem, os dados foram trabalhados de modo a configurarem entradas válidas na CNN. A figura 3.2 apresenta alguns casos práticos e a figura 3.3 apresenta de forma esquemática as etapas necessárias, sendo expostas com maior detalhe nas secções seguintes.

(a) **(b)** **(c)**

Figura 3.2: 3 principais tipos de movimento: aproximação (a), afastamento - com dados de pose - (b) e lateral (c).

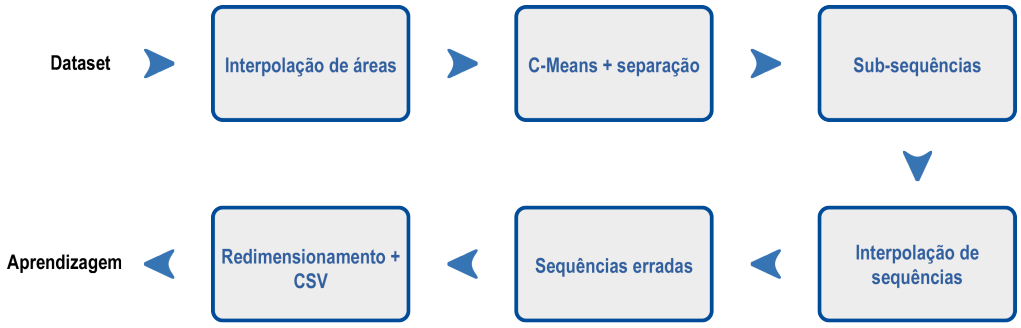


Figura 3.3: *Pipeline* de pré-processamento.

3.2.1 Interpolação de áreas

Cada *patch* de cada indivíduo possui, naturalmente, uma altura e uma largura (em píxeis), e, por consequência, uma área. Numa primeira fase é importante discriminar o tipo de movimento de uma pessoa para a geração de instâncias erradas de uma sequência ser possível (mais se acrescenta num próximo ponto). Assim, as áreas são calculadas, normalizadas no intervalo [0,1] e interpoladas (para haver um número constante para todos os indivíduos).
Note-se que existe a possibilidade de existirem menos *patches* do que o número definido (25, por exemplo). A solução passou por obter o gráfico do movimento com os dados reais e calcular dados artificiais. A figura 3.4 apresenta um movimento de afastamento expandido.

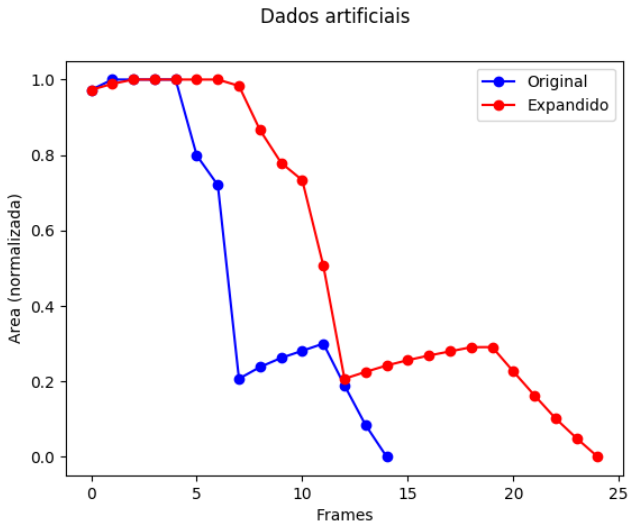


Figura 3.4: Obtenção de mais valores de área.

3.2.2 Algoritmo C-Means e separação de sequências

O algoritmo C-Means representa uma forma de *fuzzy clustering*, na qual os elementos são agrupados em *clusters* de acordo com as suas caraterísticas. Elementos de um mesmo *cluster* tendem a ser o mais semelhantes possível, enquanto que se procura maximizar a diferença entre elementos de *clusters* distintos. De notar que um elemento pode pertencer a um ou mais *clusters*.
A noção de pertença (ou *membership*) é basilar neste algoritmo. Pontos/elementos mais afastado do centro de um *cluster* têm graus de *membership* mais baixos; em sentido inverso, quanto mais perto do centro, maior é o grau.
O propósito deste algoritmo passa por minimizar a seguinte função objetivo:

$$\underset{C}{\operatorname{argmin}} \sum_{i=1}^{C} \sum_{k=1}^{N} u_{ik}^{m} ||x_k - c_i||^2. \tag{3.1}$$

Na fórmula 3.1 o parâmetro *C* representa o número de grupos a considerar, *N* o número de pontos/elementos, *m* o *fuzzifier* (tem efeito sobre a performance do algoritmo) e u_{ik} o coeficiente de *membership* (aparece multiplicado pela distância de cada ponto - x_k - a um dado centro - c_i). Por sua vez, os coeficientes de *membership* e os centros possuem as seguintes fórmulas, respetivamente:

$$u_{ik} = \frac{1}{\sum_{j=1}^{C} \left(\frac{||x_k - c_i||^2}{||x_k - c_j||^2} \right)^{\left(\frac{1}{m-1}\right)}}. \tag{3.2}$$

$$c_i = \frac{\sum_{k=1}^{N} u_{ik}^{m} x_k}{\sum_{k=1}^{N} u_{ik}^{m}}. \tag{3.3}$$

A execução do algoritmo baseia-se em em várias iterações com vista a minimizar a função objetivo (3.1) e cujo critério de paragem é, normalmente, atingido quando a diferença entre os coeficientes obtidos em iterações sucessivas for menor que um dado parâmetro (ϵ).

O C-Means foi aplicado sobre os valores normalizados e interpolados das áreas, para obter os dois padrões dominantes (essencialmente, os movimentos de afastamento e aproximação) - figura 3.5. Os valores de cada indivíduo são depois comparados com cada padrão, determinando com qual dos dois existe maior semelhança. Adicionalmente, os dados de pose (fornecidos com este *dataset*) são também usados para determinar o tipo de movimento, nomeadamente através da posição dos ombros entre si.

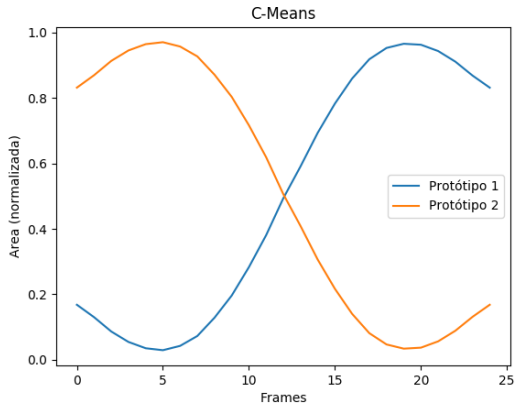


Figura 3.5: Padrões extraídos com o C-Means.

3.2.3 Geração de sub-sequências

De modo a aumentar a quantidade de dados disponíveis, foram geradas sub-sequências a partir das sequências separadas anteriormente. Uma sub-sequência é simplesmente uma variação da original. Por exemplo, a partir de uma sequência com 20 *patches*, pode-se obter uma variação que contém as imagens no intervalo [1,15].

3.2.4 Interpolação de sequências

Dado que nem todos os indivíduos têm o mesmo número de *patches*, definiu-se um tamanho constante para as sequências dadas à CNN. Todas as sequências que não cumprirem com este requisito são descartadas. Por outro lado, se uma sequência tiver mais imagens que as necessárias, será interpolada para atingir o tamanho desejado.

3.2.5 Geração de sequências erradas

Para cada sequência foi gerada uma instância errada. Se o tamanho definido para as sequências for "x", "x-1"ou "x-2"imagens mantêm-se na sequência errada. No caso de só se trocar uma imagem, a que não se manteve é substituída por uma de outra sequência (efetivamente outro indivíduo do mesmo conjunto - treino, validacao ou teste - com um tipo de movimento semelhante). O processo é semelhante se se trocarem duas imagens. Esta troca pode ser feita de forma aleatória ou de forma mais informada (e difícil de detetar):

1. **Troca aleatória:** É sorteado um indivíduo do mesmo conjunto. Uma ou duas *patches* são colocadas na instância errada da sequência original, completando a errada.

2. **Troca informada:** É escolhido o indivíduo mais aproximado ao da sequência original, sendo que todos os indivíduos candidatos são classificados num total de 100 pontos - 80 dizem respeito à cor da roupa e 20 às *soft biometrics* dos indivíduos, como o tipo e cor de cabelo, a fisionomia do corpo, entre outros.
   O indivíduo com menor distância em termos de cor em cada um dos 16 pontos de esqueleto recebe 80 pontos; o segundo melhor recebe um pouco menos e assim sucessivamente (com os extras o processo é análogo). Note-se que a distância de cor para cada ponto de esqueleto é a distância entre as três cores RGB dos pontos.
   Estando escolhido o indivíduo com melhor classificação, são colocadas as *patches* na instância errada a ser construída num dado momento.

(a)

(b)

Figura 3.6: Troca aleatória (a) e informada (b).

3.2.6 Redimensionamento das sequências e ficheiro CSV

A CNN usada requer que todas as imagens de entrada tenham um tamanho constante. Assim esta etapa passa por redimensionar todas as *patches* para um tamanho único (definiu-se 60x100).
Por último, é gerado um ficheiro CSV em que cada linha representa uma sequência, sendo marcadas como destinadas a treino, validação ou teste, bem como se são certas (sempre o mesmo indivíduo) ou erradas.

3.3 Fase de aprendizagem

O processo de treino da CNN possui variabilidade, nomeadamente na geração de instâncias erradas para cada sequência válida. Tais variações têm impacto na performance sobre os conjuntos de treino, validação e, sobretudo, de teste.
A figura 3.7 apresenta várias instâncias de aprendizagem, cujas diferenças residem na variablidade acima mencionada.
Na figura 3.7 estão as seguintes configurações:

(a) Treino e validação informados + teste informado.

(b) Treino e validação informados + teste aleatório.

(c) Treino e validação aleatórios + teste informado.

(d) Treino e validação aleatórios + teste aleatório.

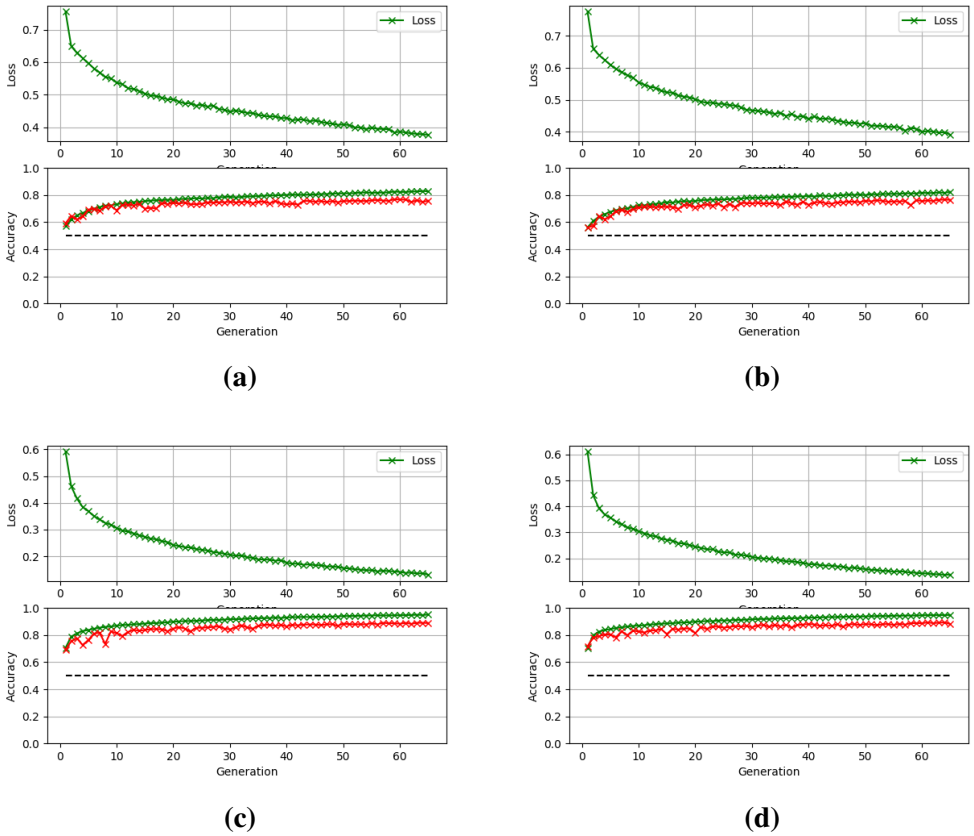


Figura 3.7: Resultados da aprendizagem com 4 configurações diferentes.

Configuração	Sequências	Acerto
Treino e validação informados + teste informado	$\sim$ 62000	78%
Treino e validação informados + teste aleatório	$\sim$ 63000	84%
Treino e validação aleatórios + teste informado	$\sim$ 112000	78%
Treino e validação aleatórios + teste aleatório	$\sim$ 111000	90%

Tabela 3.1: Taxa de acerto sobre diferentes configurações.

A tabela 3.1 mostra que se a rede for treinada com um conjunto fácil, existe uma diferença mais significativa entre a performance com dados de teste mais fáceis e mais difíceis (de quase 90% passa para 78%). Em contraste, com um treino mais difícil a diferença revela-se muito menor (de quase 84% para 78%).
Note-se que as configurações c) e d) têm mais dados de entrada devido ao facto de que para construir as instâncias negativas de forma informada são usados os dados de pose, sendo que nem todas as imagens os têm.

3.4 Discussão e Resultados

Uma análise às sequências analisadas pela rede revela alguns casos típicos de falha: uma troca feita na sequência e que a rede não foi capaz de detetar; uma sequência certa mas que contém uma ou mais *patches* com certo ruído (partes de outras pessoas) ou uma sequência com muita interferência (figura 3.8).

(a)

(b)

(c)

Figura 3.8: Casos típicos de falha.

O programa desenvolvido tem como objetivo tomar como entrada uma sequência de tamanho e dimensões iguais aos usados na aprendizagem, e devolver uma simples frase, consoante a sequência diz respeito ao mesmo indivíduo ou a indivíduos diferentes.
Para testar a solução desenvolvida foram separadas 50 sequências (nunca vistas pela rede) recolhidas de diferentes painéis e em condições de luminosidade diversas, sendo 25 positivas e 25 negativas.
A taxa de acerto foi de cerca de 52% usando tanto o modelo treinado com os conjuntos de treino, validação e teste construídos usando a troca informada como usando a troca aleatória. Os parâmetros usados estão descritos na seguinte tabela:

Parâmetro	**Valor**
Épocas	65
Taxa de aprendizagem	0.001
Probabilidade de *dropout*	1.0
Tamanho de *batch*	100
Sequências	~ 75000
Acerto (conj. teste)	**52%**

Tabela 3.2: Parâmetros base.

Alguns parâmetros foram alterados com vista a avaliar o impacto na performance do sistema. O conjunto de teste usado foi formado pelas 100 sequências separadas (também usadas na performance base, anteriormente descrita). As tabelas seguintes apresentam os parâmetros alterados:

Parâmetro	**Valor**
Épocas	65
Taxa de aprendizagem	**0.0001**
Probabilidade de *dropout*	1.0
Tamanho de *batch*	100
Sequências	~ 75000
Acerto (conj. teste)	**51%**

Tabela 3.3: Mudança na taxa de aprendizagem.

Parâmetro	**Valor**
Épocas	65
Taxa de aprendizagem	0.001
Probabilidade de *dropout*	**0.65**
Tamanho de *batch*	100
Sequências	~ 75000
Acerto (conj. teste)	**56%**

Tabela 3.4: Mudança na probabilidade de *dropout*.

Parâmetro	**Valor**
Épocas	65
Taxa de aprendizagem	0.001
Probabilidade de *dropout*	1.0
Tamanho de *batch*	**150**
Sequências	~ 75000
Acerto (conj. teste)	**54%**

Tabela 3.5: Mudança no tamanho de *batch*.

Neste caso específico e dados os testes efetuados, o parâmetros que potencialmente afeta mais a performance da rede é a probabilidade de *dropout*.

Capítulo 4

Problema 2: Identificação de Indivíduos

4.1 Introdução

O reconhecimento facial de indivíduos, face a um *dataset* conhecido, configurou o segundo desafio proposto. Foi usado o *dataset* AR [8], que contém 134 pessoas e mais de 3000 imagens (todas as pessoas têm imagens com diferente iluminação, expressões distintas ou adereços físicos). Alguns indivíduos registaram, ainda, imagens em dois dias diferentes (as mesmas variações acima mencionadas, mas registadas em dias distintos).

A figura 4.1 apresenta as imagens recolhidas de um indivíduo, a título de exemplo.

Figura 4.1: Exemplos de imagens do *dataset* AR.

4.2 Pré-processamento dos dados

Ao contrário do processo descrito no capítulo anterior, as imagens deste *dataset* já têm todas o mesmo tamanho e não existe necessidade de avaliar o tipo de movimento. Assim, apenas é necessário separá-las por indivíduo, gerar instâncias erradas para cada positiva e gerar o ficheiro CSV final (figura 4.2).

Figura 4.2: *Pipeline* de pré-processamento.

4.2.1 Separação das sequências

Todas as imagens foram agrupadas de acordo com o indivíduo a que pertencem, sendo estes posteriormente divididos em conjuntos de treino, validação e teste (com a seguinte distribuição: 80%, 10% e 10%).

4.2.2 Geração de sequências erradas

Foram exploradas duas formas de combinar as imagens dos indivíduos:

1. **Sequências de duas imagens:** uma sequência certa será composta por duas imagens do mesmo indivíduo. Independentemente das caraterísticas específicas das imagens, se a pessoa for a mesma, a sequência é válida.
   Para gerar sequências erradas, parte-se da sequência certa correspondente e é sorteado um outro indivíduo do mesmo conjunto (treino, validação ou teste) e uma das duas imagens, que se vai trocar pela equivalente do indivíduo sorteado. Por exemplo, se uma instância positiva tiver uma imagem em que o indíviduo sorri e outra em que está de óculos de sol, e for sorteada a dos óculos de sol, esta será substituída na instância errada pela imagem do indivíduo sorteado na qual também tem óculos de sol (figura 4.3).

(a) **(b)**

Figura 4.3: Sequência de duas imagens válida (a) e inválida (b).

2. **Sequências de três imagens:** uma sequência certa será composta por duas imagens do mesmo indivíduo e uma imagem de outro indivíduo do mesmo conjunto. Deste modo, é possível obter mais combinações, não só para a aprendizagem mas também para a futura identificação de imagens nunca vistas.
   Uma instância negativa consiste em manter duas imagens da positiva (uma das duas imagens do mesmo indivíduo e a do indivíduo diferente) e trocar a restante por uma equivalente (figura 4.4).

(a) **(b)**

Figura 4.4: Sequência de três imagens válida (a) e inválida (b).

Os conjuntos de imagens válidos e inválidos foram colocados num ficheiro .csv, necessário para a aprendizagem por parte da CNN.

4.3 Fase de aprendizagem

A aprendizagem decorreu de acordo com os parâmetros e configurações descritos na tabela 4.1, que contém também a taxa de acerto no conjunto de teste (os resultados são semelhantes nesta fase, quer seja com sequências de duas ou três imagens).

Parâmetro	**Valor**
Épocas	65
Probabilidade de *dropout*	1.0
Tamanho de *batch*	100
Sequências	$\sim$ 75000
Acerto (conj. teste)	**82%-83%**

Tabela 4.1: Parâmetros e configurações na aprendizagem.

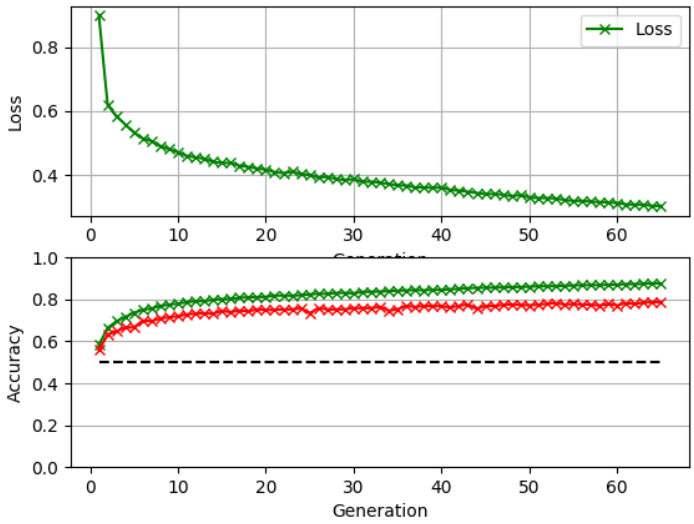


Figura 4.5: Aprendizagem da CNN sobre o *dataset* AR.

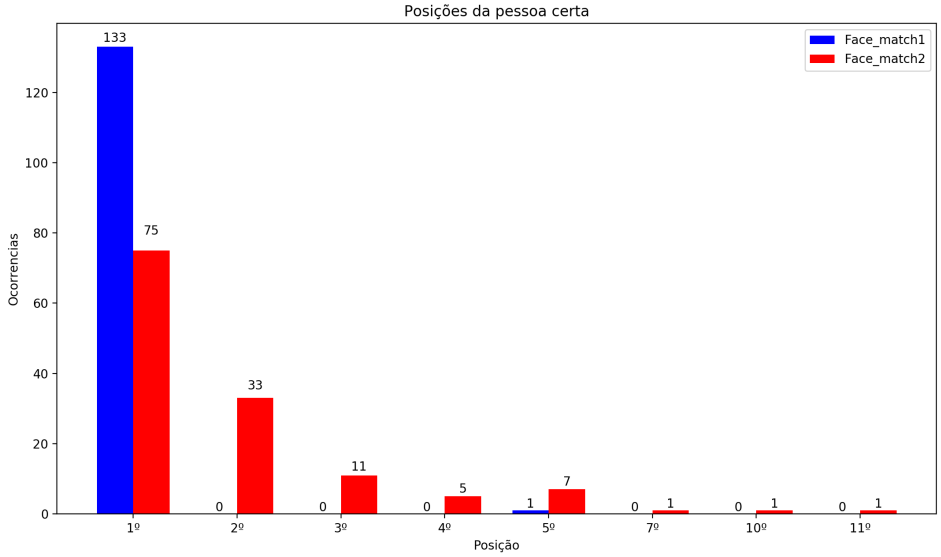
4.4 Discussão e Resultados

Foram desenvolvidos dois programas, usando os dois modelos treinados ora com duas sequências (chamado Face_match1) ora com três (Face_match2), cujo objetivo seria tomar como *input* uma imagem nunca antes vista (colocada numa pasta específica) e retornar um ranking dos indivíduos com maiores chances de serem o indivíduo presente na foto de entrada.
A título experimental, foi retirada, de cada pessoa, a imagem em que está zangada. Todas estas imagens foram então colocadas numa pasta própria e qualquer uma delas pode ser usada para testar o sistema desenvolvido.

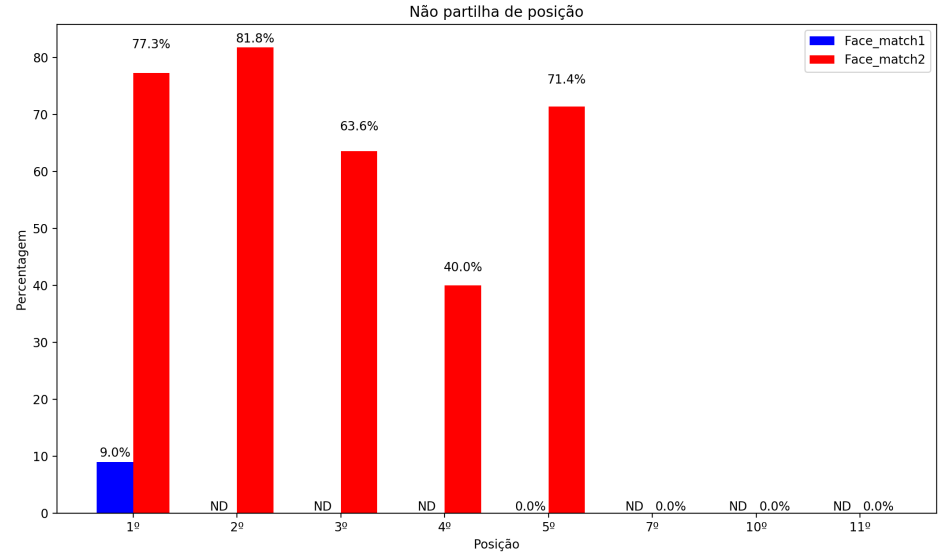
Para testar os dois sistemas desenvolvidos, foi usada uma galeria de imagens conhecidas pela rede (neste caso, as imagens de cada indivíduo com expressão neutra).

Figura 4.6: Objetivo dos sistemas desenvolvidos.

A figura 4.7 apresenta algumas observações retiradas das experiências realizadas com as imagens passíveis de ser testadas (134 imagens com expressão zangada).



(a)



(b)

Figura 4.7: Acertos de cada solução (a) e a taxa de partilha de posições (b).

Pela análise dos gráficos, é possível perceber que a solução que conjuga apenas duas imagens (Face_match1) coloca o indivíduo certo na 1ª posição com maior frequência. Contudo, coloca também outros indivíduos, não sendo por isso a mais determinista.
Por outro lado, a solução que opera sobre três imagens em simultâneo (Face_match2), é menos certeira nas avaliações que faz, mas não apresenta o problema de colocar demasiadas pessoas numa mesma posição.
De um modo geral, os casos em que a rede falhou prendem-se com os exemplos retratados na figura 4.8. Quando o mesmo indivíduo possui imagens muitos distintas, é dada uma resposta negativa falsa. Por outro lado, quando são duas pessoas diferentes, mas parecidas, é dada uma resposta positiva falsa.

(a) **(b)**

Figura 4.8: Casos típicos de falha.

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Conclusões Principais

Este trabalho serviu, acima de tudo, como uma introdução à temática das redes neuronais e respetiva implementação, para procurar dar resposta a alguns desafios. O conhecimento adquirido é e será mais valioso que qualquer resultado prático, formando uma base que se espera expandir no futuro.

5.2 Trabalho Futuro

De forma a suplementar o trabalho desenvolvido, algumas melhorias/adições podem ser destacadas:

1. **Obter imagens com menos ruído (problema 1):** O facto de numa dada imagem de um indivíduo aparecerem, ora outros indivíduos ora partes dos corpos destes, não ajuda à avaliação das sequências. Muitas delas foram dadas como erradas, sendo que se tratavam da mesma pessoa (efeito indesejável do ruído existente).

2. **Usar sequências com mais imagens (problema 2):** tendo observado as diferenças entre usar duas ou três imagens por sequência, o uso de mais imagens/combinações diferentes pode produzir resultados interessantes.

3. **Alterar parâmetros/configurações na rede:** tal como descrito no capítulo 3.4, certos parâmetros podem ter impacto no processo de aprendizagem de uma CNN e na performance sobre o conjunto de teste. Um modelo melhor contribui para um sistema de comparação de indivíduos melhor.

Bibliografia

[1] Utkarsh Sinha. First artificial neurons: The McCulloch-Pitts model, 2019. [Online] https://webvision.med.utah.edu/book/part-ix-brain-visual-areas/the-primary-visual-cortex/. Último acesso a 25 de Maio de 2019.

[2] Matthew Schmolesky. The Primary Visual Cortex By Matthew Schmolesky, 2007. [Online] https://webvision.med.utah.edu/book/part-ix-brain-visual-areas/the-primary-visual-cortex/. Último acesso a 25 de Maio de 2019.

[3] David Heeger. Perception Lecture Notes: LGN and V1, 2006. [Online] https://www.cns.nyu.edu/~david/courses/perception/lecturenotes/V1/lgn-V1.html. Último acesso a 25 de Maio de 2019.

[4] Lee Ann Remington. Chapter 13 - visual pathway. In Lee Ann Remington, editor, *Clinical Anatomy and Physiology of the Visual System (Third Edition)*, pages 233 – 252. Butterworth-Heinemann, Saint Louis, third edition edition, 2012.

[5] MLNotebook. Convolutional Neural Networks - Basics, 2017. [Online] https://mlnotebook.github.io/post/CNN1/. Último acesso a 27 de Maio de 2019.

[6] Sumit Saha. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way, 2018. [Online] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53. Último acesso a 27 de Maio de 2019.

[7] Prabhu. Understanding of Convolutional Neural Network (CNN) — Deep Learning, 2018. [Online] https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148. Último acesso a 27 de Maio de 2019.

[8] Aleix Martinez e Robert Benavente. AR Face Database Webpage, 1998. [Online] http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html. Último acesso a 28 de Maio de 2019.