

Universidade Estadual Paulista  
"Júlio de Mesquita Filho"  
Faculdade de Ciências  
Departamento de Computação

Bárbara Caroline Benato

# Sistema para Otimização baseada em Meta-Heurística de Redes Neurais por Convolução

Bauru, São Paulo  
Janeiro de 2017



Universidade Estadual Paulista  
"Júlio de Mesquita Filho"  
Faculdade de Ciências  
Departamento de Computação

Bárbara Caroline Benato

Sistema para Otimização baseada em Meta-Heurística de Redes  
Neurais por Convolução

Trabalho de Conclusão de Curso apresentado à  
Disciplina Trabalho de Conclusão de Curso II,  
como requisito final do curso de Bacharelado  
em Sistemas de Informação da Universidade  
Estadual Paulista Júlio de Mesquita Filho para  
a obtenção do Título de Bacharel em Sistemas  
de Informação

Orientador: Prof. Dr. Aparecido Nilceu Marana

Bauru, São Paulo  
Janeiro de 2017

Bárbara Caroline Benato

## **Sistema para Otimização baseada em Meta-Heurística de Redes Neurais por Convolução**

Trabalho de Conclusão de Curso apresentado à Disciplina Trabalho de Conclusão de Curso II, como requisito final do curso de Bacharelado em Sistemas de Informação da Universidade Estadual Paulista Júlio de Mesquita Filho para a obtenção do Título de Bacharel em Sistemas de Informação

Trabalho aprovado. Bauru, São Paulo, Janeiro de 2017:

---

Prof. Dr. Aparecido Nilceu Marana  
UNESP/Faculdade de Ciências

---

Prof. Dr. José Remo Ferreira Brega  
UNESP/Faculdade de Ciências

---

Prof. Dr. João Paulo Papa  
UNESP/Faculdade de Ciências

Bauru, São Paulo  
Janeiro de 2017

Dedico esta, bem como todas as minhas demais conquistas, aos meus pais e familiares.



# Agradecimentos

Gostaria de expressar minha gratidão, primeiramente, aos meus pais, Júlio e Hélen, e familiares, Vô Mauro, Vó Lurdinha, Valéria, Patrici, Marcos, João Paulo, Daiane, Helena, Fabrício, Vó Bela, Gorete, Andrea e Mateus, que me proporcionaram as condições adequadas durante a vida toda para que eu pudesse estudar e me formar. Ao meu primo Gustavo, por sempre se prontificar em ajudar e ensinar, quando desespero e a ansiedade apareciam. Ou pelos puxões de orelha quando o cansaço superava a atenção. Obrigada por todo exemplo, amor, carinho, respeito, educação e confiança. Obrigada estarem sempre do meu lado e terem feito esse sonho se realizar. Eu amo vocês!

Ao professor e tio João Paulo, por incentivar o estudo e foco nas melhores notas, ciente das oportunidades que o conhecimento me traria. Obrigada por sempre tentar tirar e cobrar o melhor de mim em todos os momentos. Ao professor e orientador Nilceu, por ter dedicado seu tempo, atenção e paciência tanto durante a Iniciação Científica, quanto no desenvolvimento deste trabalho de conclusão, bem como pelo exemplo de conduta e de didática. Ao professor e coordenador de curso Remo, em especial pela paciência e também pela ajuda com os processos burocráticos sempre confusos e atrapalhados. Agradeço, ainda, todos aqueles professores que me proporcionaram ensinamentos e não somente aulas em sala, e que me fizeram ter vontade também de transmitir o que se aprendeu.

Aos meus amigos de curso e os descolados, Lillian, Everton, Lucas, Juliana, Matheus e Renan. Obrigada por todo apoio, conversas, diversão, festas e amizade. É claro que não podia faltar umas briguinhas. Ao Marcelo, meu amigo e não só colega, pela prontidão e ajuda com as matérias, trabalhos e provas quando eu mais precisei, além de acreditar e participar do projeto de informática comigo. Aos parceiros de reuniões de curso, reuniões do CAada e colegas que Bauru me trouxe e que participaram dessa etapa.

Aos amigos que sempre me acompanharam, torceram e aliviaram a pressão que Bauru me trazia: Larissa, Adrieli, Daiane, Pamela, Bianca, Mateus, Pedro, Bruna, Carolina, Thais e Lucas. Obrigada pelos desabafos e por tornarem sempre as coisas mais leves, mesmo não estando presentes fisicamente nos meus dias.

Por fim, agradeço a Deus por permitir que eu chegasse até aqui e que fizesse meu caminho sempre ser perfeito e mágico naquilo que deveria ser. Obrigada pela minha família,

pelos meus professores, pelos meus amigos, pelas pessoas que cruzaram meu caminho, pelos lugares onde eu estive e também por aqueles que eu não estive. Que o lado bom das coisas possa sempre ser lembrado. E que assim seja.

Obrigada a todos que contribuíram para a conclusão de mais uma etapa e para a Bárbara que eu me tornei hoje.

“Quando se sonha sozinho é apenas um sonho. Quando se sonha juntos é o começo da realidade.” – Miguel de Cervantes

*“Tudo que é seu encontrará alguma maneira de chegar até você.”*  
*(Chico Xavier)*



# Resumo

Redes Neurais por Convolução (*Convolutional Neural Networks* - CNN) têm sido amplamente empregadas nos últimos anos como uma alternativa aos tradicionais métodos de extração de características. Sua metodologia baseada no processamento hierárquico do cérebro humano tem obtido bons resultados em tarefas de classificação de imagens, dado que são altamente invariantes à mudanças de escala e rotação. Entretanto, um grande problema de tais abordagens diz respeito à configuração de seus parâmetros, os quais podem atingir dezenas e, dependendo do tamanho da rede neural, centenas. Tal configuração de parâmetros é essencial para o bom desempenho das CNN, sendo que uma escolha manual dos mesmos configura um problema de alta complexidade combinatória. O presente trabalho de conclusão de curso visou modelar o problema de otimização dos parâmetros de uma CNN como sendo uma tarefa de otimização evolucionista, dado que poucos estudos nesse contexto têm sido realizados. Desta forma, o trabalho em questão objetivou desenvolver um sistema capaz de aplicar um algoritmo evolucionista recentemente proposto pela literatura, denominado Otimização das Aves Migratórias (*Migrating Birds Optimization* - MBO), para otimização dos parâmetros de uma CNN. Adicionalmente, a abordagem proposta foi avaliada no contexto de reconhecimento de expressão facial em humanos.

**Palavras-chave:** Reconhecimento de Padrões; Meta-heurística; Expressões Faciais; Aprendizado em profundidade.



# Abstract

Convolutional Neural Networks (CNN) have been extensively employed in the last years as an alternative to the traditional methods of feature extraction. Their methodology is based on the hierarchical processing of human brain, and they have obtained good results for image classification tasks, since they are invariant to changes in scale and rotation. However, a big issue of such approaches concern with their parameter configuration, which can reach dozens and, depending on the neural network size, hundreds of parameters. Such parameter configuration is essential to the good performance of a CNN, being a manual choice of them a problem with high combinatorial complexity. This project have aimed to modeling the problem of CNN parameter optimization as being an evolutionary optimization task, since just a few works have been conducted in this context. In this way, the present work developed a system capable of apply a recent proposed evolutionary algorithm called Migrating Birds Optimization (MBO) for CNN parameter optimization. Additionally, the proposed approach was evaluated in the context of human facial expression recognition.

**Key-words:** Pattern Recognition; Meta-Heuristic; Facial Expression; Deep learning.



# Lista de ilustrações

Figura 1 – Arquitetura típica de uma Rede Neural por Convolução. . . . .	22
Figura 2 – Diferença entre os fluxos de ar nas partes superior e inferior da asa de uma ave. . . . .	24
Figura 3 – Janela principal da primeira solução pesquisada. . . . .	26
Figura 4 – Janela principal da segunda solução pesquisada. . . . .	27
Figura 5 – Janela principal da terceira solução pesquisada. . . . .	27
Figura 6 – Exemplos de diferentes expressões faciais da base JAFFE. . . . .	31
Figura 7 – Exemplos de diferentes expressões faciais da base CK. . . . .	31
Figura 8 – Exemplos de diferentes expressões faciais da base TFEID. . . . .	32
Figura 9 – Representação do sistema por meio de diagrama de blocos. . . . .	33
Figura 10 – Diagrama de classe do sistema. . . . .	35
Figura 11 – Caso de uso da seleção do algoritmo utilizado. . . . .	36
Figura 12 – Caso de uso da conversão da base de dados. . . . .	37
Figura 13 – Caso de uso da seleção da base de dados utilizado. . . . .	38
Figura 14 – Caso de uso da configuração da arquitetura de rede. . . . .	39
Figura 15 – Caso de uso do treinamento da rede. . . . .	40
Figura 16 – Caso de uso da exibição dos resultados. . . . .	41
Figura 17 – Aba Otimização. . . . .	44
Figura 18 – Aba Conversão de Base. . . . .	45
Figura 19 – Aba Base. . . . .	46
Figura 20 – Aba Rede. . . . .	46
Figura 21 – Aba Treinamento. . . . .	47
Figura 22 – Aba Resultados. . . . .	48
Figura 23 – Mensagem exibida pela falta de configuração de Otimização. . . . .	50
Figura 24 – Mensagem exibida pela falta de seleção de Base. . . . .	50
Figura 25 – Mensagem exibida pela falta de configuração de Arquitetura. . . . .	51
Figura 26 – Mensagem exibida pela falta de Arquivo de saída e número de Rodadas. . . . .	51
Figura 27 – Arquitetura proposta para os experimentos. . . . .	54

# Sumário

<b>1</b>	<b>INTRODUÇÃO</b>	<b>17</b>
<b>2</b>	<b>DETALHAMENTO DO PROBLEMA</b>	<b>19</b>
<b>3</b>	<b>REFERENCIAL TEÓRICO</b>	<b>21</b>
<b>3.1</b>	<b>Redes Neurais por Convolução</b>	<b>21</b>
<b>3.2</b>	<b>Otimização das Aves Migratórias</b>	<b>23</b>
<b>4</b>	<b>SOLUÇÕES EXISTENTES</b>	<b>25</b>
<b>4.1</b>	<b>Rede Neuronal Artificial para o Reconhecimento de Caracteres com Interface Gráfica</b>	<b>25</b>
<b>4.2</b>	<b>Desenvolvimento de uma interface gráfica para classificadores de imagem</b>	<b>25</b>
<b>4.3</b>	<b>Reprodução de Informação Associada a Expressões Faciais</b>	<b>26</b>
<b>5</b>	<b>TECNOLOGIAS UTILIZADAS</b>	<b>29</b>
<b>6</b>	<b>ESPECIFICAÇÃO DO SISTEMA</b>	<b>33</b>
<b>6.1</b>	<b>Módulos do Sistema</b>	<b>33</b>
<b>6.2</b>	<b>Diagrama de Classe</b>	<b>34</b>
<b>6.3</b>	<b>Diagramas de Caso de Uso</b>	<b>36</b>
6.3.1	Selecionar Algoritmo de Otimização	36
6.3.2	Converter Base de Dados	37
6.3.3	Selecionar Base de Dados	38
6.3.4	Configurar Arquitetura da Rede	38
6.3.5	Treinar Rede	39
6.3.6	Exibir Resultados	39
<b>7</b>	<b>SOFTWARE PARA A OTIMIZAÇÃO DE REDES NEURAIS POR CONVOLUÇÃO</b>	<b>43</b>
<b>7.1</b>	<b>Telas</b>	<b>43</b>
7.1.1	Otimização	43
7.1.2	Conversão de Base	44
7.1.3	Base	45
7.1.4	Rede	45
7.1.5	Treinamento	45

---

7.1.6	Resultados . . . . .	47
<b>8</b>	<b>TESTE E VALIDAÇÃO . . . . .</b>	<b>49</b>
<b>9</b>	<b>UTILIZAÇÃO DO SISTEMA PARA O RECONECIMENTO DE EX- PRESSÕES FACIAIS E RESULTADOS OBTIDOS . . . . .</b>	<b>53</b>
<b>9.1</b>	<b>Metodologia . . . . .</b>	<b>53</b>
9.1.1	Aplicação . . . . .	53
9.1.2	Implementação . . . . .	54
9.1.3	Configuração Experimental . . . . .	56
<b>9.2</b>	<b>Resultados Experimentais . . . . .</b>	<b>57</b>
9.2.1	Base de Dados JAFFE . . . . .	57
9.2.2	Base de Dados Cohn-Kanade AU-Coded Expression . . . . .	58
9.2.3	Base de Imagens Taiwanese Facial Expression . . . . .	59
<b>10</b>	<b>CONCLUSÕES . . . . .</b>	<b>61</b>
	<b>REFERÊNCIAS . . . . .</b>	<b>63</b>



# 1 Introdução

O presente trabalho de conclusão de curso teve como principal objetivo o desenvolvimento de um sistema que pudesse validar o uso de meta-heurísticas para a otimização de técnicas de aprendizado de máquina, em particular, de Redes Neurais por Convolução, no âmbito da classificação de expressões faciais em imagens. O sistema, assim, fornece os resultados com a acurácia da técnica para a base de dados escolhida.

O problema foi motivado pelo elevado número de parâmetros da técnica de aprendizado de máquina. Tal processo de configuração de parâmetros, quando feito manualmente, costuma ser demorado e mais propenso a erros. Outra motivação diz respeito a necessidade de simplificar a metodologia empregada para a utilização de meta-heurística para a otimização de parâmetros de Redes Neurais por Convolução.

Um grande desafio, portanto, foi desenvolver um sistema capaz integrar a biblioteca da técnica de aprendizado em profundidade adotada ao algoritmo de meta-heurística implementado, bem como garantir o bom funcionamento e troca de dados entre os módulos internos do sistema.

O presente documento está estruturado em 9 capítulos, além do atual que contém a introdução. O Capítulo 2 e o Capítulo 3 apresentam, respectivamente, o detalhamento do problema e o referencial. O Capítulo 4 exhibe as soluções existentes acerca do tema proposto. As ferramentas e tecnologias escolhidas para o desenvolvimento da interface proposta são descritas no Capítulo 5. O Capítulo 6 apresenta a especificação do sistema desenvolvido. O detalhamento do sistema para a otimização de redes neurais por convolução é apresentado no Capítulo 7. Os procedimentos de teste e validação da interface são descritos no Capítulo 8. No Capítulo 9 são apresentadas a metodologia e os resultados experimentais obtidos para a validação do sistema. E, por fim, o Capítulo 10 descreve as conclusões dos principais aspectos e ganhos esperados com o presente trabalho, assim como trabalhos futuros.



## 2 Detalhamento do Problema

Técnicas de aprendizado de máquina são amplamente utilizadas para a extração de características de, por exemplo, sinais ou imagens. Essa tarefa é realizada empregando um conjunto de treinamento como base, visando o aprendizado da técnica, para posterior classificação de um outro conjunto de dados (DUDA; HART; STORK, 2000). A otimização de tais técnicas refere-se ao estudo e seleção dos melhores parâmetros responsáveis por conduzir uma melhor taxa de reconhecimento.

Dentre as técnicas de aprendizado de máquina, podemos destacar aquelas baseadas em Aprendizado em Profundidade, do inglês *Deep Learning*, as quais caracterizam-se por sua hierarquia no momento de extração e aprendizado do conhecimento. Tais técnicas são baseadas no processamento de informações do cérebro humano, o qual também o faz de maneira hierárquica, ou seja, cada etapa é responsável por extrair e codificar diferentes níveis de informação. Muito embora técnicas de aprendizado em profundidade tenham sido propostas há alguns anos atrás (LECUN et al., 1998; HINTON, 2002), só recentemente essas técnicas tiveram seu merecido reconhecimento, dado que, quando criadas, os computadores não tinham poder computacional suficiente para efetuar o treinamento dessas técnicas de maneira eficaz.

Redes Neurais por Convolução (*Convolutional Neural Networks* - CNNs) (LECUN et al., 1998) têm sido bastante empregadas atualmente, principalmente devido à sua capacidade de descrição em baixo e alto nível das informações contidas em uma imagem. Tais técnicas caracterizam-se por conter diferentes estágios com operações similares, tais como filtragem, normalização e sub-amostragem. Destacam-se, também, por serem abordadas no contexto de visão computacional, processamento de imagens e de fala. Entretanto, CNNs possuem uma grande quantidade de parâmetros, que vão desde o número e tamanho de cada filtro utilizado na etapa de filtragem até o parâmetro que controla o nível de sub-amostragem de uma imagem, por exemplo. Tais parâmetros ainda podem ser distintos entre cada etapa (camada) de execução de uma CNN.

Muitos trabalhos têm empregado buscas aleatórias com o intuito de “popular” o conjunto de parâmetros de uma CNN. No entanto, trabalhos recentes evidenciam que técnicas de otimização por meta-heurísticas têm obtido resultados mais eficazes do que uma busca aleatória (FEDOROVICI et al., 2012). Sendo assim, este trabalho de conclusão de curso visou a elaboração e o desenvolvimento de um sistema que pudesse viabilizar a aplicação de novas técnicas de otimização por meta-heurísticas no contexto de otimização de parâmetros de uma CNN. Mais especificamente, objetivou-se a aplicação de uma técnica recente chamada

de Otimização das Aves Migratórias (*Migration Birds Optimization* - MBO) (DUMAN; UY-SAL; ALKAYA, 2012) no contexto supracitado. Esse algoritmo é baseado no comportamento de voo de aves em bando, as quais buscam reduzir o gasto de energia e aumentar o alcance de voo. Por fim, a validação da presente trabalho consistiu na classificação a partir de imagens.

## 3 Referencial Teórico

### 3.1 Redes Neurais por Convolução

Redes do tipo CNN podem ser vistas como uma representação de uma classe maior de modelos baseados na arquitetura de Hubel e Wiesel, os quais realizaram um estudo seminal em 1962 sobre o córtex primário de gatos. Esse trabalho identificou, basicamente, dois tipos de células: as células simples, as quais possuem uma tarefa análoga à etapa de filtragem por banco de máscaras, e as células complexas, as quais realizam uma tarefa semelhante à etapa de amostragem das CNNs.

O primeiro modelo que simulou uma Rede Neural por Convolução em computador foi o amplamente conhecido “Neocognitron” (FUKUSHIMA; MIYAKE, 1982), o qual aplicava um algoritmo de treinamento não supervisionado na etapa de filtragem por banco de máscaras, bem como um algoritmo de treinamento supervisionado na última camada da rede. Posteriormente, LeCun et al. (LECUN et al., 1989; LECUN et al., 1990) simplificaram essa arquitetura propondo a utilização do algoritmo de retropropagação para o treinamento da rede como um todo de maneira supervisionada. Com o decorrer do tempo, várias aplicações que fizeram uso de CNNs foram surgindo.

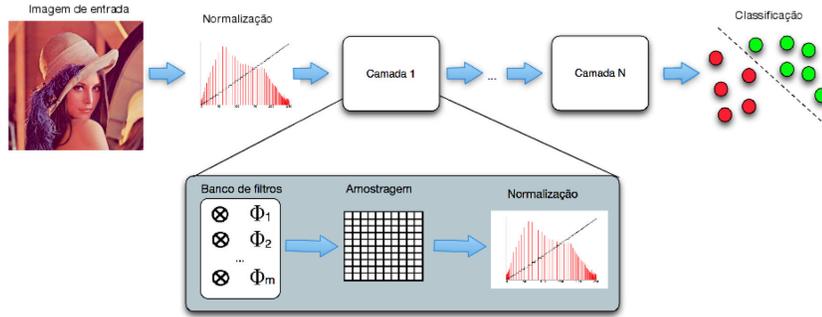
Basicamente, uma CNN pode ser entendida como uma sequência de três camadas de processamento de imagem. Dada uma imagem de entrada, uma CNN essencialmente extrai uma representação de alto nível da mesma, chamada de *imagem multibandas*, cujos atributos dos *pixels* são concatenados em um vetor de características para posterior aplicação de técnicas de reconhecimento de padrões. A Figura 1 apresenta a arquitetura de uma Rede Neural por Convolução.

Em cada camada da CNN são executadas três operações, sendo a primeira delas um convolução com um banco de filtros, a segunda uma amostragem e a terceira uma etapa de normalização. Conforme pode ser observado na Figura 1, existe ainda a possibilidade de ser realizada uma operação de normalização no início de todo o processo. Segue, abaixo, uma breve descrição de cada uma dessas etapas:

**Banco de Filtros:** Seja uma imagem multibandas  $\hat{I} = (D_I, \vec{I})$  tal que  $D_I \in \mathbb{N}^2$  é o domínio da imagem, e  $\vec{I} = \{I_1(p), I_2(p), \dots, I_m(p)\}$  corresponde ao vetor de atributos de um *pixel*  $p = (x_p, y_p) \in D_I$ . Quando  $\hat{I}$  é uma imagem tons de cinza, por exemplo, temos então que  $m = 1$  e  $\hat{I} = (D_I, I)$ .

Seja  $\phi = (\mathcal{A}, W)$  um filtro com pesos  $W(q)$  associado a cada *pixel*  $q \in \mathcal{A}(p)$ , onde  $\mathcal{A}(p)$

Figura 1 – Arquitetura típica de uma Rede Neural por Convolução.



Fonte: Elaborado pelo autor.

denota uma máscara quadrada de tamanho  $L_A \times L_A$  centrada em  $p$ , e  $q \in \mathcal{A}(p)$  se, e somente se,  $\max\{|x_q - x_p|, |y_q - y_p|\} \leq (L_A - 1)/2$ . No caso de filtros multibandas, os seus pesos podem ser representados como vetores  $\vec{W}_i(q) = \{w_{i,1}(q), w_{i,2}(q), \dots, w_{i,m}(q)\}$  para cada filtro  $i$ , e um banco multibanda de filtros pode ser então definido como  $\phi = \{\phi_1, \phi_2, \dots, \phi_n\}$ , onde  $\phi_i = (\mathcal{A}, \vec{W}_i)$ ,  $i = \{1, 2, \dots, n\}$ .

Assim, a convolução entre uma imagem de entrada  $\hat{I}$  e um filtro  $\phi_i$  produz a banda  $i$  da imagem filtrada  $\hat{J} = (D_J, \vec{J})$ , onde  $D_J \in D_I$  e  $\vec{J} = \{J_1, J_2, \dots, J_n\}$ ,  $\forall p \in D_J$ :

$$J_i(p) = \sum_{\forall q \in \mathcal{A}(p)} \hat{I}(q) \vec{W}_i(q). \quad (3.1)$$

Os pesos de  $\phi_i$  são usualmente gerados a partir de uma distribuição uniforme, i.e.,  $U(0, 1)$ , e posteriormente normalizados com média zero e norma unitária para garantir o seu espalhamento dentro da esfera unitária.

**Amostragem:** essa operação é de fundamental importância para uma CNN, a qual objetiva proporcionar invariância translacional para as características extraídas. Seja  $\mathcal{B}(p)$  as regiões de amostragem de tamanho  $L_B \times L_B$  centrada em  $p$ . Adicionalmente, seja  $D_K = D_J/s$  uma operação de amostragem regular a cada  $s$  *pixels*. Assim, a operação de amostragem resultante na imagem  $\hat{K} = (D_K, \vec{K})$  é definida como segue:

$$K_i(p) = \sqrt[\alpha]{\sum_{\forall q \in \mathcal{B}(p)} J_i(q)^\alpha}, \quad (3.2)$$

em que  $p \in D_K$  denota cada *pixel* da nova imagem,  $i = \{1, 2, \dots, n\}$  corresponde às bandas da imagem, e  $\alpha$  é um parâmetro que controla a sensibilidade da operação.

**Normalização:** a última operação de uma CNN é a sua normalização, a qual é um mecanismo amplamente utilizado para melhorar a performance desse tipo de arquitetura

de aprendizado em profundidade (COX; PINTO, 2011). Essa operação é baseada nos mecanismos encontrados em neurônios corticais (GEISLER; ALBRECHT, 1992), sendo também definida sob uma região quadrada  $\mathcal{C}(p)$  de tamanho  $L_C \times L_C$  centralizada no *pixel*  $p$ , tal que:

$$O_i(p) = \frac{K_i(p)}{\sum_{j=1}^n \sum_{\forall q \in \mathcal{C}(p)} K_j(q) K_i(q)}. \quad (3.3)$$

Assim, a operação acima é realizada para cada *pixel*  $p \in D_O \subset D_k$  da imagem resultante  $\hat{O} = (D_O, \vec{O})$ .

Assim como as Redes Neurais por Convolução, alguns algoritmos de otimização foram previamente estudados durante o período de Iniciação Científica realizada pela aluna. Tais meta-heurísticas, em especial, foram então escolhidas para validar o presente trabalho de conclusão de curso.

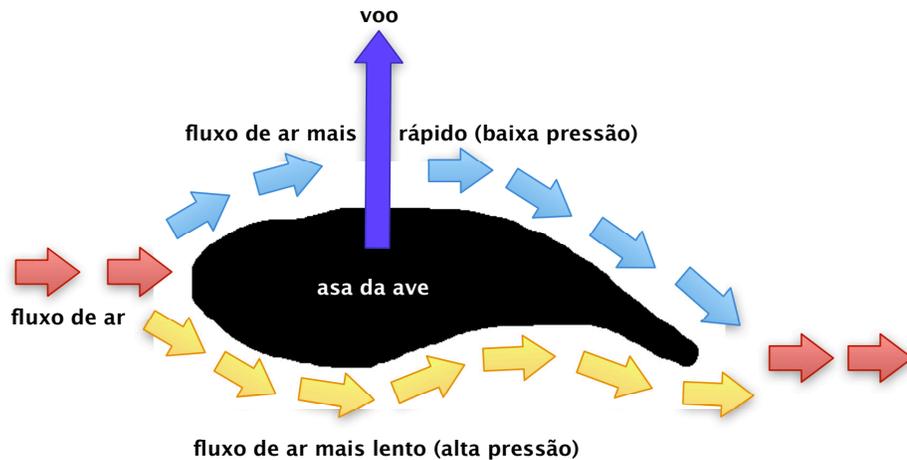
## 3.2 Otimização das Aves Migratórias

A técnica de Otimização das Aves Migratórias (*Migration Birds Optimization - MBO*) (DUMAN; UYSAL; ALKAYA, 2012) é baseada na formação “V” durante o voo das aves migratórias, a qual é uma disposição espacial bastante eficiente para minimizar a energia gasta durante o seu voo (DUMAN; UYSAL; ALKAYA, 2012). O formato da asa de um pássaro trabalha de maneira similar a um aerofólio, o qual é utilizado para “cortar” o ar quando o mesmo encontra-se em direção contrária a ele. O fluxo de ar sobre a parte superior do aerofólio tem que se mover mais distante do que da parte inferior. Consequentemente, o fluxo de ar na parte superior tem uma pressão menor do que na parte inferior, sendo esta diferença de pressão que permite com que as aves consigam voar.

Considerando um único pássaro em sua tentativa de levantar voo, a sua velocidade é o fator mais importante dessa etapa. Isto deve-se ao fato de que, quanto maior a velocidade, menor a diferença de pressão entre as partes superior e inferior de suas asas, fazendo com que uma maior pressão de subida seja atingida, permitindo, assim, com que o pássaro voe. A Figura 2 ilustra esse procedimento.

O ar deslocado por um pássaro contribui para o voo de outra ave, reduzindo, assim, a sua energia gasta para manter-se no ar; isso explica o fato de as aves voarem em conjunto seguindo uma formação específica, chamada de formação “V”. Esse tipo de formação é também utilizado pelas aves para evitar com que elas colidam entre si, mantendo contato visual. Lissaman e Schollenberger (B.; A., 1970) foram um dos pioneiros a realizar esse estudo, argumentando que, quanto maior o número de aves em conjunto nessa mesma formação, maior

Figura 2 – Diferença entre os fluxos de ar nas partes superior e inferior da asa de uma ave.



Fonte: Elaborado pelo autor.

será a energia poupada. Os autores estimam que um grupo de 25 aves voando em conjunto, por exemplo, possui um alcance de voo 70% maior do que uma ave voando sozinha.

Paralelamente, o algoritmo MBO é uma técnica que realiza uma busca pela vizinhança de um determinado agente, ou seja, uma ave. A sua inicialização é realizada de maneira aleatória, sendo os agentes posicionados em uma formação do tipo “V”. Começando com a primeira solução (ave líder do bando), é verificada se a mesma consegue ser melhorada analisando-se a sua vizinhança: caso alguma das aves vizinhas propicie uma solução melhor, ela então passa a ser líder do bando (posição frontal da formação “V”). Para cada uma das aves não-líderes, a sua solução também é avaliada com relação à sua vizinhança, bem como os melhores vizinhos da solução encontrada pela ave à sua frente na formação também são utilizados. Após todas as aves do bando terem sido avaliadas, a melhor solução (líder do bando) é movida para a última posição da formação (é assumido que essa ave já gastou mais energia do que as demais), e a segunda melhor solução passa agora a liderar o bando. Esse procedimento é realizado até algum critério de convergência ser estabelecido.

O número de vizinhos utilizados para a avaliação de uma melhor solução pode ser interpretado como sendo a força necessária para levantar voo: para uma grande vizinhança, o algoritmo assume que as aves estão viajando a uma velocidade baixa (podemos fazer a analogia que, quando estamos viajando a uma velocidade menor, conseguimos interpretar melhor os detalhes à nossa volta). A principal característica que distingue MBO das outras técnicas de otimização meta-heurísticas é o seu mecanismo de beneficiamento entre as possíveis soluções do problema.

## 4 Soluções Existentes

Foram pesquisadas três soluções que envolvessem a criação de sistemas para o problema de classificar e treinar redes neurais de forma simplificada e com aumento das taxas de reconhecimento. Não foram encontradas, até o momento, soluções que envolvessem a criação de um sistema incluindo a otimização dos parâmetros de técnicas de aprendizado em profundidade por meta-heurísticas.

### 4.1 Rede Neuronal Artificial para o Reconhecimento de Caracteres com Interface Gráfica

Na primeira solução analisada (TEIXEIRA et al., 2009). Os autores criaram uma interface para facilitar a visualização dos resultados da rede neural, comparando a imagem de entrada junto com sua classificação feita pela rede, a saída. São analisados também alguns casos em que pixels são alterados na imagem de entrada, dificultando a percepção do classificador.

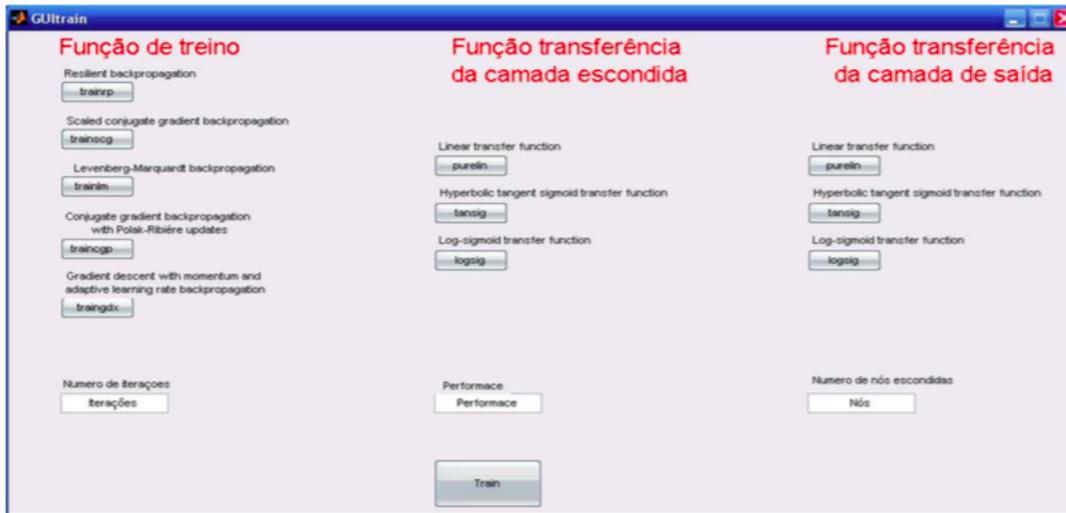
A principais semelhanças com a solução proposta é que este trabalho também usa redes neurais, além de utilizar imagens para alimentar a rede. A interface permite a visualização da saída da rede, ou seja, a classificação da imagem, assim como a solução desenvolvida.

O que difere uma abordagem da outra é que a solução analisada utiliza a rede neural do tipo feed-forward, sem aprendizado em profundidade, enquanto a solução proposta utiliza esta última técnica. A primeira solução utiliza o software Matlab para manipulação de redes neurais, diferentemente da proposta que fez uso da biblioteca em linguagem C, implementada grupo de pesquisa Recogna (MARANA; PAPA, 2010) a qual a autora participa. A validação da técnica é feita em imagens de dígitos, enquanto o presente trabalho pretendeu usar imagens de expressões faciais humanas.

### 4.2 Desenvolvimento de uma interface gráfica para classificadores de imagem

Um segundo trabalho (TEIXEIRA, 2011) foi analisado. Tal estudo visou desenvolver uma interface gráfica que facilite a abordagem no contexto de reconhecimento de dados, para o processamento de uma ou mais imagens. Esta interface apresenta um módulo de pré processamento, um de descritor e extração de características da imagem e outro de

Figura 3 – Janela principal da primeira solução pesquisada.



Fonte: (TEIXEIRA et al., 2009).

classificação. Cada um destes módulos fornece a opção de uso várias técnicas diferentes, ou seja, a interface não é construída para apenas um tipo de extrator de características ou classificador, o que a torna flexível e capaz de comparar técnicas. A ferramenta permite escolher as imagens do sistema de arquivos, mostrar a tarefa dos diferentes descritores na imagem escolhida. Um banco de imagens de formigas foi escolhido.

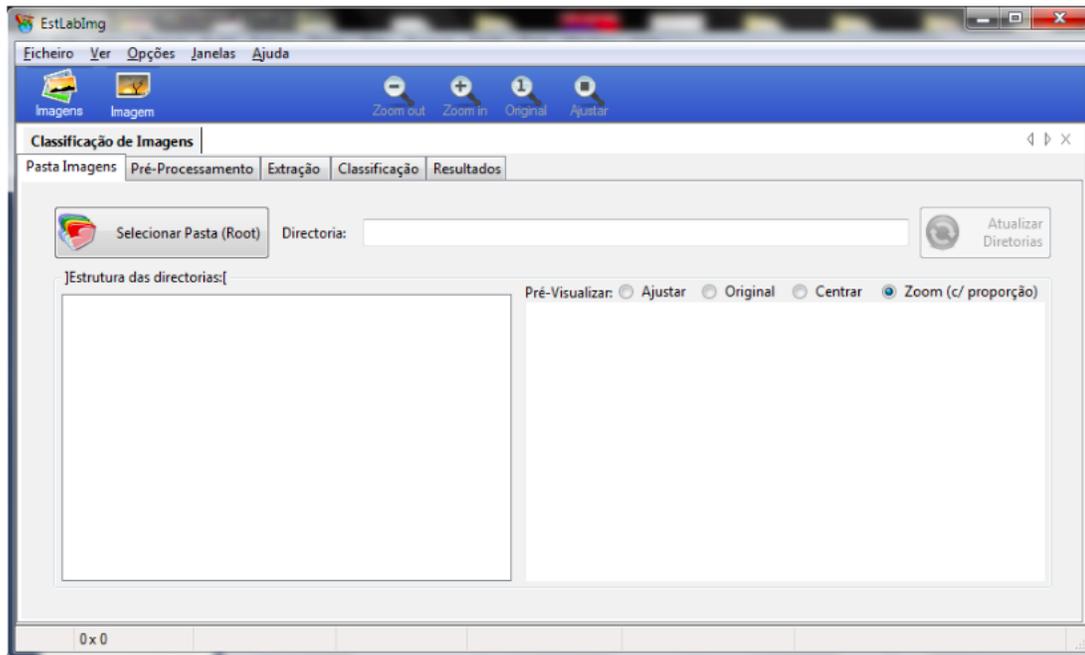
O estudo abordou diversas técnicas e, por isso, mostrou-se muito completo e didático. A forma como a interface foi criada, dividida e implementada está detalhadamente descrita neste segundo trabalho analisado. A interface possibilita visualizar minuciosamente os processos que ocorrem nas etapas de reconhecimento de uma imagem, além de obter o resultado final de classificação da referida imagem, sendo o objetivo da técnica.

### 4.3 Reprodução de Informação Associada a Expressões Faciais

O terceiro trabalho foi escolhido de forma que abordasse os dados da presente proposta: de expressões faciais (ROSARIO, 2018). A principal ideia do trabalho é implementar uma interface que detecte uma face humana em uma imagem obtida e reconheça automaticamente a expressão facial pelo processamento da imagem e reconhecimento de padrões. A partir da classificação da imagem, a interface utiliza o rótulo para relacionar a imagem com o *emoticon* associado àquela expressão. A ferramenta também permite a classificação de músicas de acordo com as expressões do usuário.

A abordagem dada às expressões faciais é diferente do presente trabalho. Neste ter-

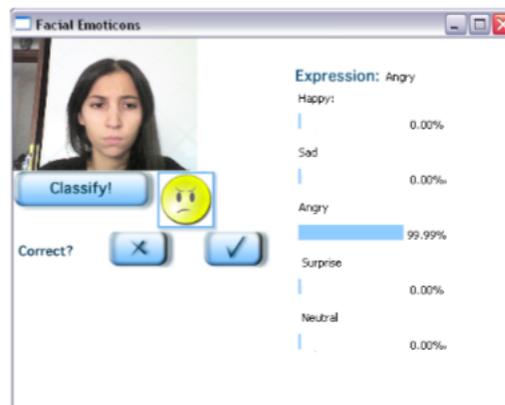
Figura 4 – Janela principal da segunda solução pesquisada.



Fonte: (TEIXEIRA, 2011).

ceiro trabalho analisado, optou-se por usar um modelo que leva em consideração a distância e tamanho dos principais elementos da face, como olho, sobrancelha, nariz e formato da boca, para definir o rótulo da expressão facial. Cada um destes elementos, quando combinados, definem um rótulo.

Figura 5 – Janela principal da terceira solução pesquisada.



Fonte: (ROSARIO, 2018).



## 5 Tecnologias Utilizadas

Muito embora o estudo das CNNs tenha sido realizado, optou-se por utilizar a biblioteca de código aberto de aprendizado em profundidade denominada Caffe (JIA et al., 2014). Tal ferramenta proporciona a implementação de código orientado às Unidades de Processamento Gráfico, do inglês *Graphics Processing Unit* (GPU), permitindo o aumento do poder de processamento dos computadores, fazendo com que a execução dos algoritmos seja realizada de maneira muito mais eficiente. Dado que, geralmente, o aprendizado dos parâmetros ótimos/sub-ótimos de uma CNN é bastante custoso, pois, cada atualização da posição de uma possível solução (ave) requer o uma nova execução da etapa de treinamento da CNN, essa biblioteca é de grande valia para o contexto do proposta em questão.

O algoritmo de otimização MBO foi implementado pela aluna junto a biblioteca de otimização LibOPT-plus, uma biblioteca que conta com diversas técnicas meta-heurísticas desenvolvidas em linguagem de programação C pelo grupo Recogna. Além disso, foi utilizada a biblioteca de integração LibOPT4Caffe (ROSA, 2016) que possibilita assimilar a biblioteca LibOPT-plus com a biblioteca Caffe.

O sistema de otimização de CNNs para o reconhecimento de expressões faciais foi desenvolvido na linguagem de programação Java. Tal linguagem foi escolhida por possuir um pacote, o *JSch*, responsável por criar canais de comunicação remotos e seguros, além de possuir classes e métodos que facilitam a criação da interface e a integração com os comandos da linguagem *ShellScript* do sistemas baseados no modelo UNIX (OPENGROUP, 1995). A biblioteca *JSch* não está presente entre as bibliotecas padrões da linguagem Java e, por isso, precisou ser instalada separadamente. A fim de elaborar a interface para o sistema, o ambiente de desenvolvimento de aplicação NetBeans IDE (ORACLE, 2016) foi utilizado.

Os diagramas utilizados para modelar o sistema proposto foram elaborados com o auxílio do Processon (LTD, 2014), uma ferramenta gratuita disponível *on-line* para a criação e compartilhamento em tempo real.

Foram utilizadas bases de expressões faciais para validação da técnica e da aplicabilidade da interface. Cada base de imagem precisou ser previamente transferida e salva na pasta da biblioteca de *Deep Learning* utilizada, a Caffe, especificamente na pasta “*data*” da biblioteca. Alguns procedimentos referentes a separação das imagens de treino e teste, rotulação e criação de arquivos de texto também foram executados conforme o padrão para preparação de dados da biblioteca Caffe<sup>1</sup>. O nome atribuído a pasta de cada conjunto de imagem é utilizado no sistema para identificar as bases de imagens.

<sup>1</sup> <http://caffe.berkeleyvision.org/gathered/examples/imagenet.html>

Considerando o alto custo computacional que uma CNN requer, foi utilizado um número reduzido do conjunto total de cada base de dados. A partir deste conjunto menor, a proporção de imagens foi de 60% para o conjunto treinamento e 40% para o conjunto de teste. Além disso, em cada base de imagens, uma mesma quantidade de imagens foi selecionada de cada expressão facial.

Deste modo, as seguintes bases de dados de imagens foram utilizadas no presente trabalho:

- Base de dados *Japanese Female Facial Expression* (JAFFE)<sup>2</sup>: essa base de dados contém 213 imagens de 7 expressões faciais (6 expressões básicas mais 1 neutra) obtidas de 10 mulheres japonesas. Foram utilizadas 213 imagens, sendo 128 de treinamento e 85 de teste.
- Base de dados *Cohn-Kanade AU-Coded Expression*<sup>3</sup> (CK): essa base de dados contém 486 imagens de 97 indivíduos com diferentes poses e expressões faciais. Entretanto, utilizamos neste projeto apenas as imagens com pose frontal, totalizando 251 imagens, sendo 147 de treinamento e 104 de teste.
- Base de Imagens *Taiwanese Facial Expression*<sup>4</sup> (TFEID): essa base de dados consiste de 7.200 imagens capturadas de 40 modelos (20 homens e 20 mulheres), contendo 8 expressões faciais diferentes: neutra, raiva, desprezo, medo, felicidade, tristeza e surpresa. Utilizamos apenas as imagens com posição frontal, totalizando 320 imagens, sendo 192 de treinamento e 128 de teste.

As Figuras 6, 7 e 8 mostram alguns exemplos das imagens selecionadas das bases de dados acima mencionadas.

---

<sup>2</sup> <http://www.kasrl.org/jaffe.html>

<sup>3</sup> <http://www.pitt.edu/emotion/ck-spread.htm>

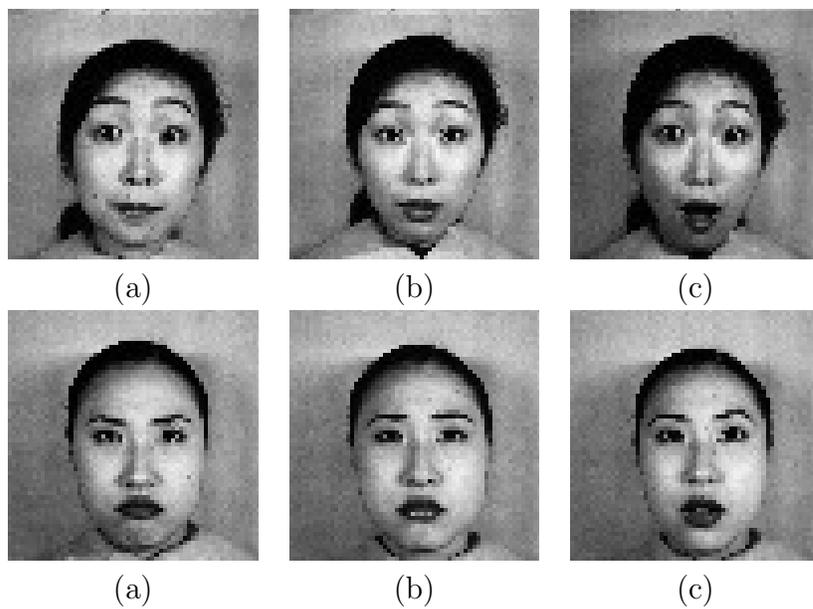
<sup>4</sup> <http://bml.ym.edu.tw/tfeid/>

<sup>5</sup> Disponível em: <http://www.kasrl.org/jaffe.html>, Acesso em set. 2016

<sup>6</sup> Disponível em: <http://www.pitt.edu/emotion/ck-spread.htm>, Acesso em set. 2016

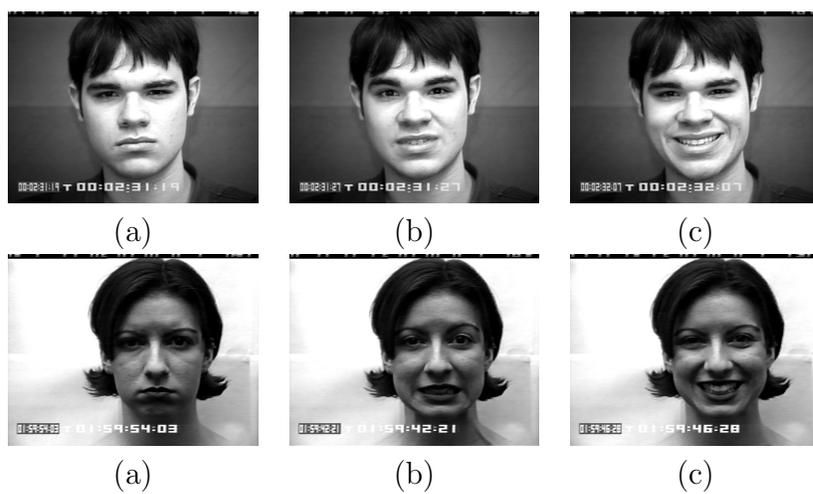
<sup>7</sup> Disponível em: <http://bml.ym.edu.tw/tfeid/>, Acesso em set. 2016

Figura 6 – Exemplos de diferentes expressões faciais da base JAFFE.



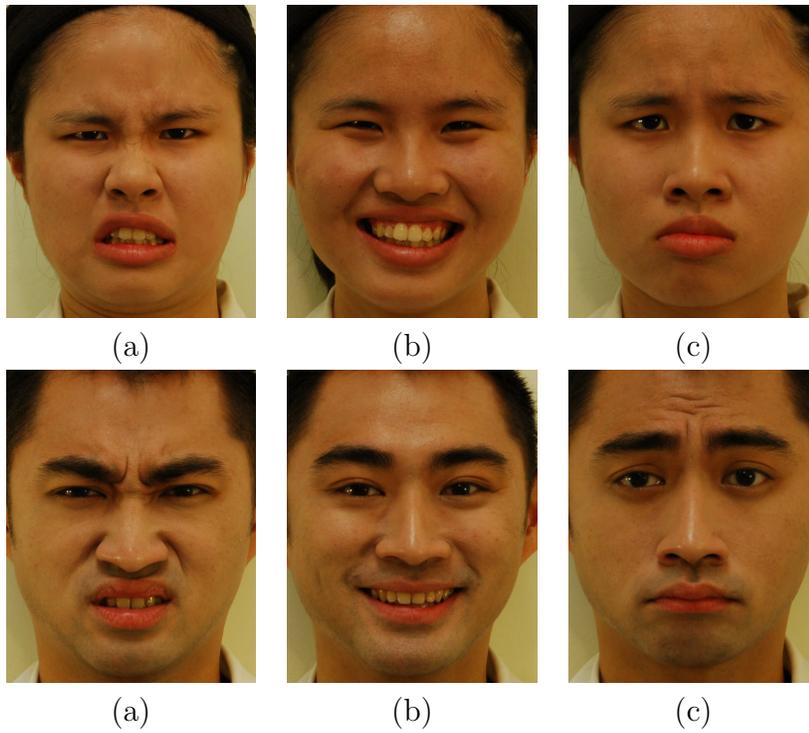
Fonte: Imagens da base JAFFE<sup>5</sup>.

Figura 7 – Exemplos de diferentes expressões faciais da base CK.



Fonte: Imagens da base CK<sup>6</sup>.

Figura 8 – Exemplos de diferentes expressões faciais da base TFEID.



Fonte: Imagens da base TFEID<sup>7</sup>.

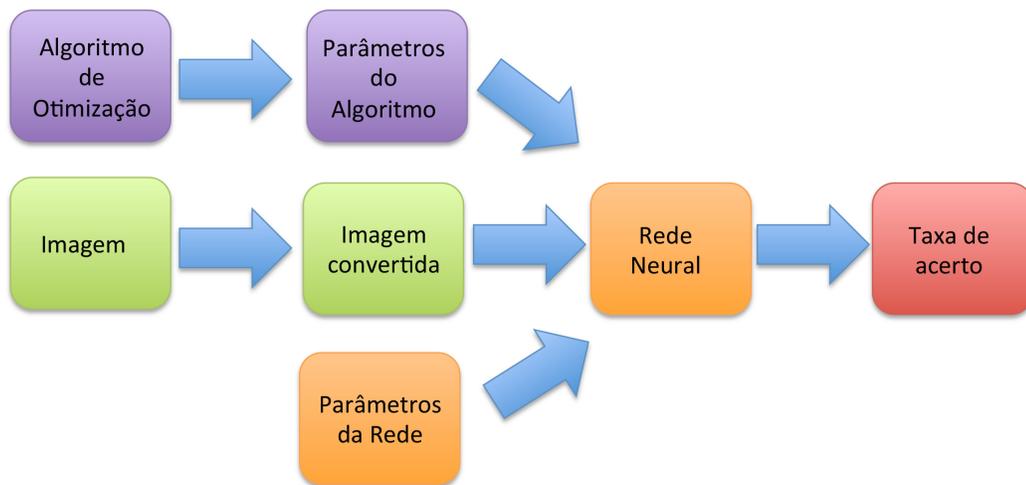
## 6 Especificação do Sistema

A seguir uma representação geral do sistema é apresentada em forma de um diagrama de blocos. Cada bloco corresponde a uma representação lógica de uma parte da solução proposta (módulo). Também são apresentados dois tipos de diagramas UML para melhor ilustrar o projeto do sistema. Por fim, um protótipo do sistema a ser desenvolvido é apresentado.

### 6.1 Módulos do Sistema

Uma visão geral do sistema a ser desenvolvido é dada pelo diagrama de blocos. Cada bloco do diagrama corresponde a um módulo do sistema, segundo seus requisitos.

Figura 9 – Representação do sistema por meio de diagrama de blocos.



Fonte: Elaborado pelo autor.

**Algoritmo de Otimização:** Para o problema de escolha de parâmetros de uma CNN foi aplicado a tarefa de otimização evolucionista. Existem várias técnicas implementadas pelo grupo de pesquisa Recogna(MARANA; PAPA, 2010) e seria interessante se o sistema pudesse escolher entre os algoritmos existentes, a fim de facilitar e viabilizar o emprego da técnica. Este módulo, portanto, inclui uma lista dos algoritmos implementados, a LibOPT-plus(PAPA, 2016), para que um deles possa ser escolhido para a metodologia descrita.

**Parâmetros do Algoritmo de Otimização:** Ainda que o problema de seleção de parâmetros de CNNs tenha sido solucionado, o algoritmo de otimização escolhido necessita ser configurado em um número reduzido de parâmetros. Tais parâmetros variam de acordo com o algoritmo escolhido e, assim, faz-se necessário que os parâmetros possam ser alterados e escolhidos de acordo com a vontade do usuário.

**Imagem:** Este módulo é responsável por carregar e escolher o conjunto de imagens que serão utilizadas. Elas poderão ser escolhidas dentro do próprio sistema.

**Imagem Convertida:** A imagem é transformada e modificada conforme os requisitos da CNN neste módulo. A biblioteca Caffe(JIA et al., 2014), que contém a implementação da CNN, requer um padrão específico de entrada das imagens conforme sua coloração (se colorida ou tons de cinza) e dimensão. Além disso, as imagens são convertidas para estruturas, denominadas vetores de características, que também são específicas para a biblioteca.

**Parâmetros da Rede:** Alguns parâmetros e limiares referentes às CNNs precisam ser escolhidos e configurados. A interface deve possibilitar a alteração destes parâmetros, que referem-se ao número de rodadas globais, amplitude máxima e mínima que os parâmetros da rede podem assumir e número de camadas escondidas, por exemplo.

**Rede:** Neste módulo as informações sobre o conjunto de imagens são fornecidos, como a entrada para a rede e o algoritmo de otimização, e parâmetros escolhidos definem a forma como o processo de treinamento e teste da CNN acontecem. É nele, também, que o processamento da técnica de aprendizado em profundidade ocorre de fato.

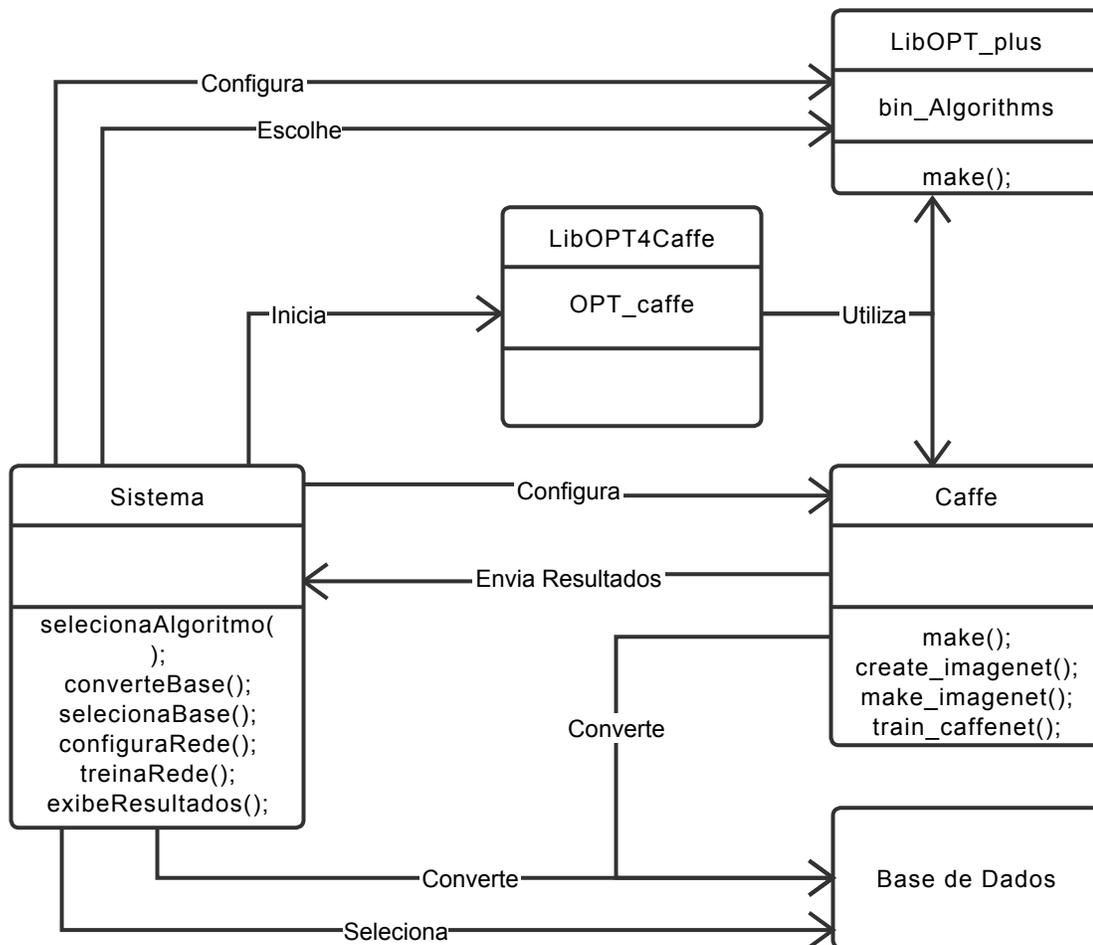
**Taxa de Acerto:** A saída direta da rede é representada por este módulo. A técnica de aprendizado em profundidade tem como sua saída a taxa de acerto para determinado conjunto de dados. Essa taxa representa o grau de exatidão do conjunto de teste em relação ao treinamento.

## 6.2 Diagrama de Classe

O diagrama de classe, apresentado pela Figura 6.2, foi escolhido de forma a melhor ilustrar a interação e comunicação real entre os módulos do sistema.

O sistema é capaz de configurar e escolher o algoritmo de otimização da biblioteca de otimização escolhida, a LibOPT Plus, utilizado no processo de otimização de redes neurais por convolução. Tal comunicação é abstraída na primeira tela do sistema.

Figura 10 – Diagrama de classe do sistema.



Fonte: Elaborado pelo autor.

Além disso, o sistema é responsável por converter a base de dados para os padrões de entrada da Caffe, bem como escolher a base que será utilizada no processo. A conversão é simplificada pelo sistema na segunda tela e a seleção da base utilizada é definida pela terceira tela do sistema.

A escolha da arquitetura de rede e configuração dos parâmetros da arquitetura são representados na quarta tela do sistema, que omite do usuário todas as alterações fundamentais em arquivos de configuração da Caffe, além de definir os caminhos corretos dos mesmos.

O início do treinamento é dado pela quinta tela do sistema. Nesta etapa, o sistema implementa uma chamada à biblioteca `LibOPT4Caffe`, que faz o intermédio entre a `LibOPT Plus` e a `Caffe`, com o algoritmo de otimização escolhido na primeira etapa.

O sistema, em sua sexta tela, faz a tarefa de exibir os resultados obtidos ao final do

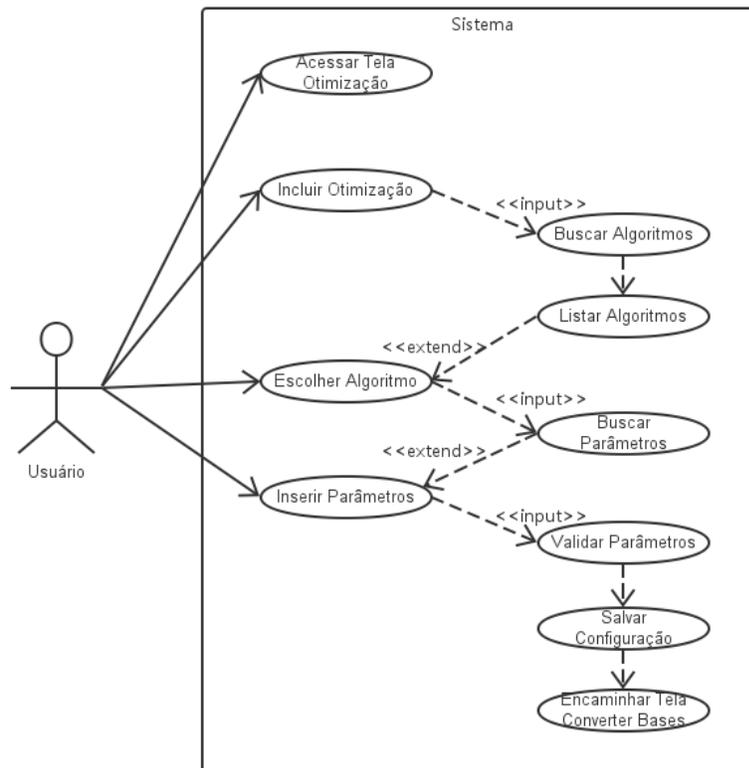
processo.

## 6.3 Diagramas de Caso de Uso

Os diagramas de Caso de Uso foram definidos por módulos, a fim de representar a interação do usuário com o sistema proposto e suas principais funcionalidades.

### 6.3.1 Selecionar Algoritmo de Otimização

Figura 11 – Caso de uso da seleção do algoritmo utilizado.



Fonte: Elaborado pelo autor.

Caso de uso: selecionarAlgoritmo

Atores: Usuário

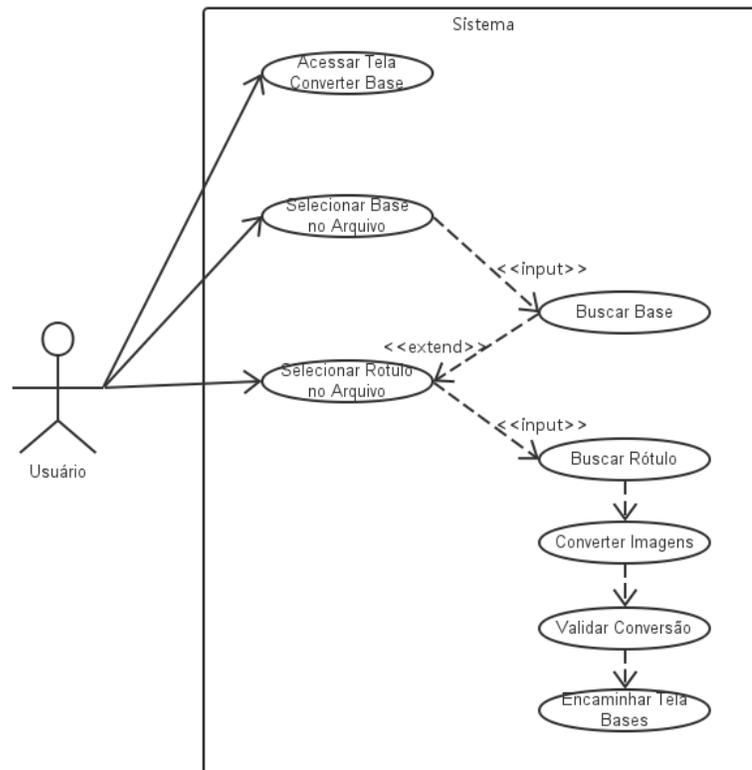
Propósito: Definir a configuração referente ao algoritmo de otimização utilizado, informando parâmetros como número de agentes no espaço de busca, dimensão do problema, número máximo de iterações, e outros parâmetros particulares de cada técnica.

Descrição: O usuário deverá informar os parâmetros solicitados e, após finalizar a configuração, para prosseguir, deverá salvar essa configuração.

Tipo: primário e essencial.

### 6.3.2 Converter Base de Dados

Figura 12 – Caso de uso da conversão da base de dados.



Fonte: Elaborado pelo autor.

Caso de uso: converterBase

Atores: Usuário

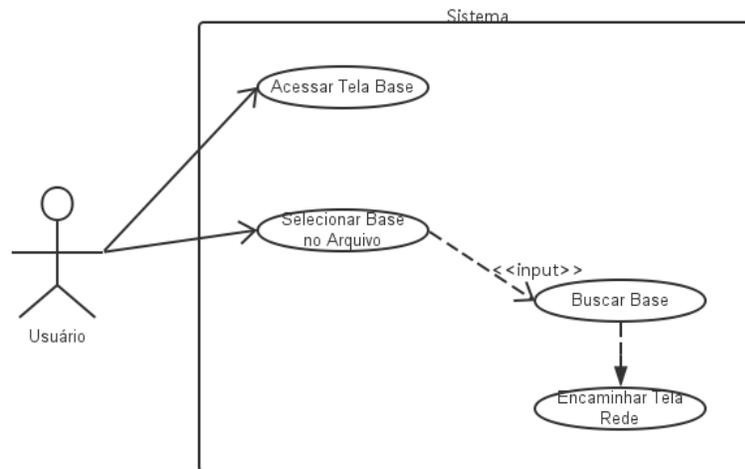
Propósito: Converter determinada base de dados proveniente do sistema de arquivos para os padrões de entrada de dados necessários, ou seja, dimensão e coloração da imagem, para a biblioteca de aprendizado de máquina em questão, a Caffe.

Descrição: O usuário deverá selecionar a base de dados que será convertida no sistema de arquivos, definir um rótulo para a base, confirmar e validar a conversão.

Tipo: primário e opcional.

### 6.3.3 Selecionar Base de Dados

Figura 13 – Caso de uso da seleção da base de dados utilizado.



Fonte: Elaborado pelo autor.

Caso de uso: selecionarBase

Atores: Usuário

Propósito: Selecionar a base de dados, previamente convertida, dentro do sistema para que possa ser utilizada no processo de reconhecimento de expressões faciais otimizado.

Descrição: O usuário deverá selecionar a base de dados que será convertida no sistema de arquivos, definir um rótulo para a base, confirmar e validar a conversão.

Tipo: primário e essencial.

### 6.3.4 Configurar Arquitetura da Rede

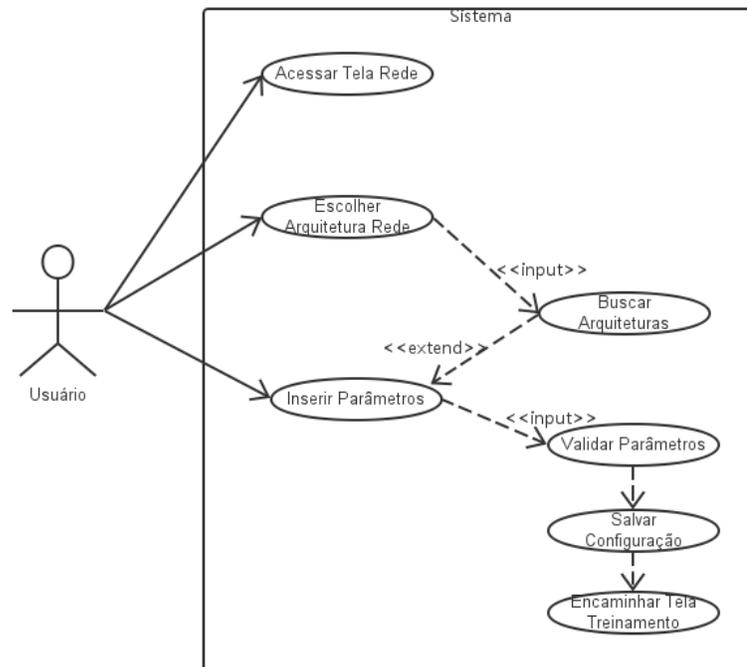
Caso de uso: configurarRede

Atores: Usuário

Propósito: Definir e configurar a arquitetura de rede e parâmetros que serão utilizados para a classificação pela técnica de aprendizado de máquina escolhida. Os parâmetros incluem taxa de aprendizado base, valor gamma, tamanho do passo, número máximo de iterações internas, momentum, decaimento e modo de processamento (GPU ou CPU).

Descrição: O usuário deverá selecionar a arquitetura que quer usar e, a partir dela, escolher os parâmetros de acordo com seu problema. Os parâmetros escolhidos são validados e salvos pelo sistema.

Figura 14 – Caso de uso da configuração da arquitetura de rede.



Fonte: Elaborado pelo autor.

Tipo: primário e essencial.

### 6.3.5 Treinar Rede

Caso de uso: treinarRede

Atores: Usuário

Propósito: Escolher nome e destino em que o arquivo com os resultados da rede serão salvos, bem como iniciar a classificação da imagens juntamente com o algoritmo de otimização escolhido. Os logs da operação deverão ser exibidos para o usuário.

Descrição: O usuário deverá escolher o destino do arquivo que será salvo e iniciar o processo de classificação das imagens.

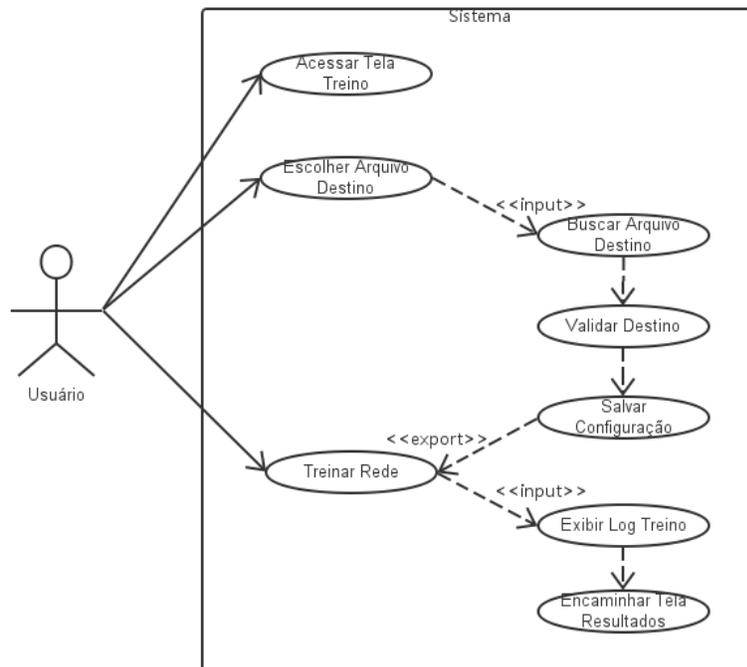
Tipo: primário e essencial.

### 6.3.6 Exibir Resultados

Caso de uso: exibirResultados

Atores: Usuário

Figura 15 – Caso de uso do treinamento da rede.



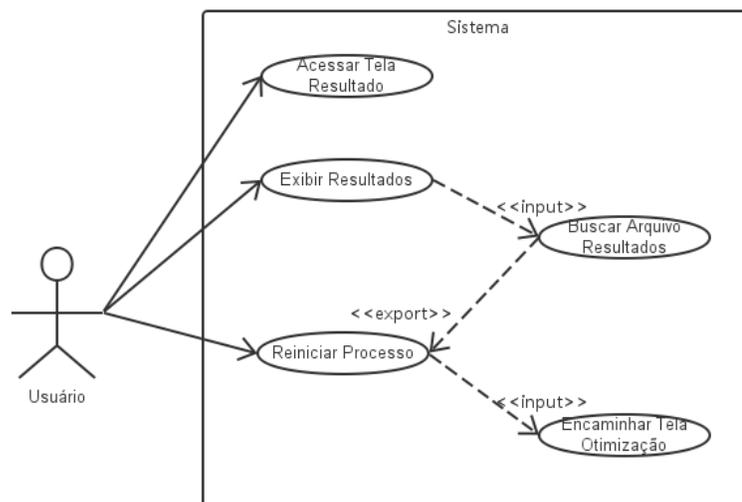
Fonte: Elaborado pelo autor.

Propósito: Exibir os resultados obtidos pelo classificador juntamente com o algoritmo de otimização escolhido, a fim de que se possa analisar o desempenho do processo para o conjunto de dados estabelecido. O usuário tem a opção de reiniciar o processo anteriormente iniciado com as mesmas configurações utilizadas em sua última inicialização.

Descrição: O usuário deverá selecionar a opção para que os resultados sejam exibidos e pode, se for de sua vontade, selecionar a opção que reinicia o processo.

Tipo: primário e opcional.

Figura 16 – Caso de uso da exibição dos resultados.



Fonte: Elaborado pelo autor.



## 7 Software para a Otimização de Redes Neurais por Convolução

O presente trabalho visou desenvolver um sistema que tornasse a tarefa de otimizar Redes Neurais por convolução mais simples aos usuários das bibliotecas Caffe e LibOPT. Cada tela do sistema é responsável por configurar uma etapa, ou módulo, do processo já descrito. Devido a necessidade encontrada no decorrer do trabalho, as bibliotecas foram instaladas em uma máquina com poder de processamento adequado no laboratório do grupo de pesquisa que a aluna participa. Desta forma, o sistema desenvolvido cria canais de comunicação remotos via servidor à máquina em questão.

Assim, a primeira tela é responsável pela organização das configurações no que diz respeito ao uso ou não dos algoritmos de otimização propostos, bem como a seleção do algoritmo escolhido e configurações de parâmetros individual. Na segunda tela, o usuário deve fornecer o nome da base que deseja converter segundo os padrões de entrada da biblioteca Caffe, caso queira utilizar uma base que ainda não foi previamente convertida. A base já convertida, que será utilizada no processo de treinamento da biblioteca Caffe, deve ser escolhida na terceira tela. A quarta tela refere-se a seleção da arquitetura de rede a ser usada no treinamento pelo usuário, bem como sua configuração de parâmetros. Na quinta tela, o usuário pode nomear o arquivo de saída com a acurácia final do treinamento da rede, além de definir o número máximo de rodadas do processo. É nesta mesma tela que o processo de treinamento pode ser iniciado. Os resultados obtidos com o treinamento da rede depois de configurados adequadamente todos os elementos necessários para o treino são exibidos na sexta tela.

### 7.1 Telas

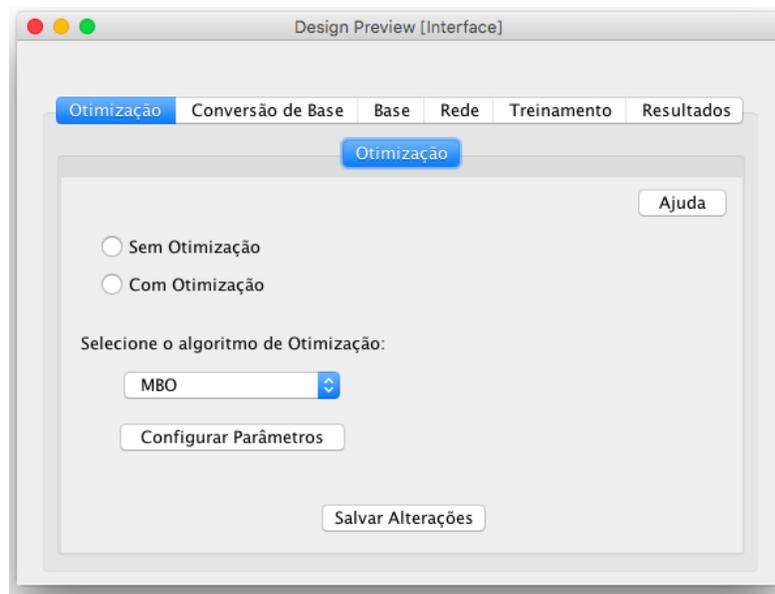
As telas que compõem o sistema desenvolvido neste trabalho de conclusão de curso são apresentadas e discutidas a seguir. Cada tela possui um botão “Ajuda” que, quando acionado, mostra as instruções e informações necessárias para o entendimento de cada etapa.

#### 7.1.1 Otimização

Na Figura 17, a primeira tela do sistema é apresentada. Nesta tela, o usuário pode escolher se no processo de treinamento utilizará otimização ou não. Esta opção é dada, caso o usuário queira obter os resultados originais da Caffe. O usuário também pode escolher qual o

algoritmo de otimização ele irá utilizar no processo de treinamento para avaliar os resultados. No presente trabalho, optamos por comparar o MBO ao algoritmo Busca Harmônica (*Harmony Search* - HS) (GEEM, 2009), uma técnica de otimização meta-heurística inspirado no processo de criação de músicas. uma técnica meta-heurística baseada em enxame e uma das mais utilizadas na literatura. Depois de escolhido o algoritmo que será utilizado no processo, o usuário tem a possibilidade de configurar os parâmetros do algoritmo selecionado. Para isto, abre-se uma nova janela com os parâmetros. Os parâmetros, depois de configurados, podem ser salvos. O usuário deve salvar as configurações selecionadas.

Figura 17 – Aba Otimização.

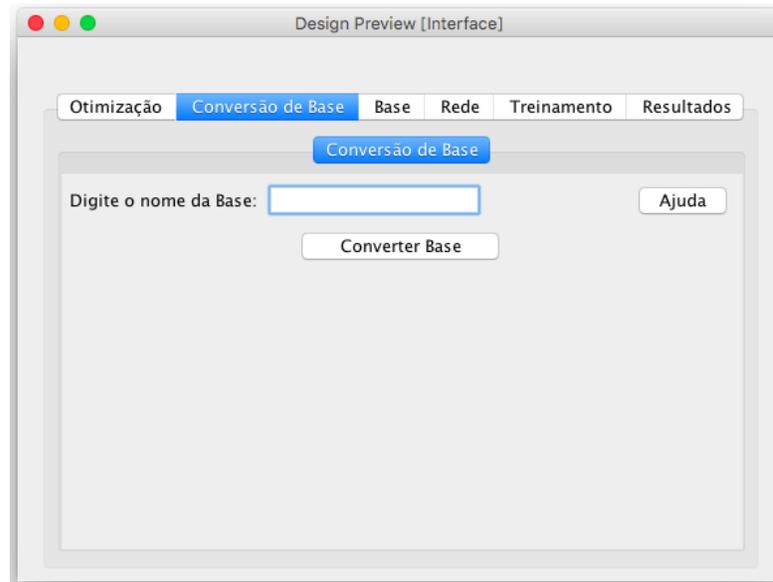


Fonte: Elaborado pelo autor.

### 7.1.2 Conversão de Base

A Figura 18 apresenta a segunda tela do sistema, responsável pela conversão de base. A conversão de base é uma etapa opcional no processo, ou seja, ela só é necessária quando pretende-se utilizar uma base não convertida previamente nos padrões da biblioteca Caffe. Assim, depois de adicionar a pasta da base de imagens dentro do diretório “*data*” da biblioteca, o usuário deve apenas informar o nome da pasta da base de imagens em questão. Ao clicar no botão “Conversão de Base”, o sistema converte a base e a torna disponível ser escolhida na próxima tela.

Figura 18 – Aba Conversão de Base.



Fonte: Elaborado pelo autor.

### 7.1.3 Base

A terceira tela do sistema é representada pela Figura 19. Nesta tela, é possível selecionar a base que será utilizada no processo de treinamento da rede neural por convolução. Depois de escolhida, o usuário deve salvar as configurações selecionadas.

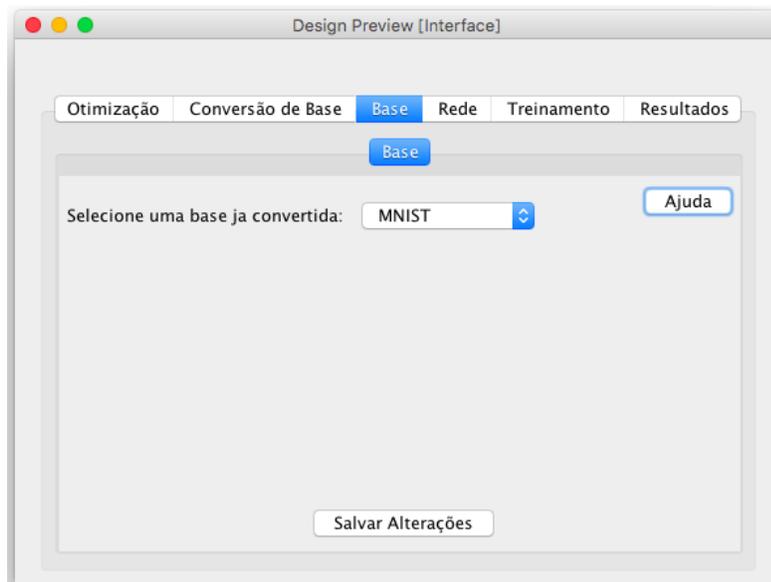
### 7.1.4 Rede

Na Figura 20, é apresentada a quarta tela do sistema. Nela, o usuário pode escolher entre as arquiteturas de rede previamente definidas pela biblioteca Caffe. Cada uma das arquiteturas são destinadas a conjuntos específicos de imagens. Depois de escolhida a arquitetura de rede que será utilizada no processo de treinamento, o usuário pode configurar os parâmetros individuais da arquitetura. Uma nova janela é então aberta para alterar tais parâmetros e salvá-los. Depois de configurados os parâmetros, o usuário deve, então, salvar as alterações escolhidas.

### 7.1.5 Treinamento

A Figura 21 exhibe a quinta tela do sistema, a tela de treinamento. O usuário deve, primeiramente, escolher o nome do arquivo de saída a ser gerado pela biblioteca, como também o número de rodadas que o processo deve conter. Depois de definidos o nome do arquivo e o número de rodadas, o usuário deve salvar as alterações feitas. É nesta mesma tela que

Figura 19 – Aba Base.



Fonte: Elaborado pelo autor.

Figura 20 – Aba Rede.



Fonte: Elaborado pelo autor.

o usuário deve iniciar o processo de treinamento da rede, depois de todas as configurações estabelecidas anteriormente em cada tela do sistema. Uma nova janela com o processamento da operação aberta a partir disso.

Figura 21 – Aba Treinamento.

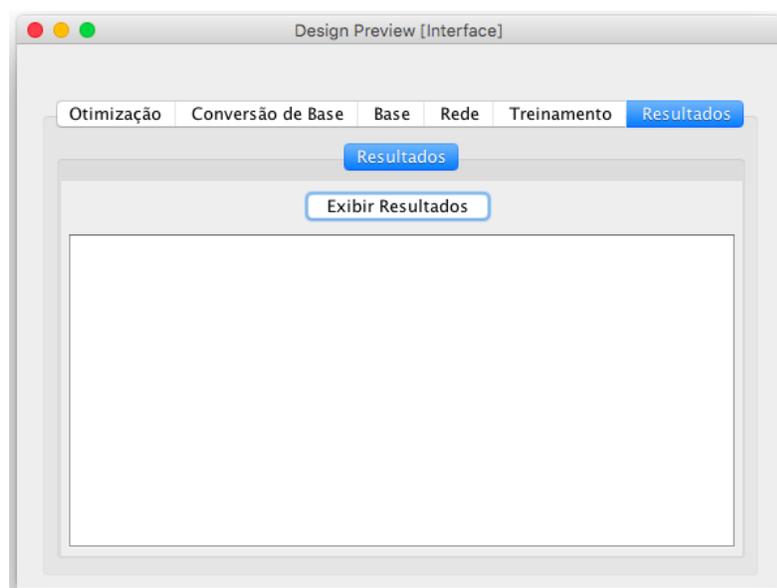


Fonte: Elaborado pelo autor.

### 7.1.6 Resultados

Depois que o treinamento estiver finalizado, o usuário poderá seguir para a sexta tela, apresentada na Figura 22. Nesta tela, o usuário é capaz de carregar os resultados obtidos por meio do processo de treinamento da rede com a otimização, base e arquitetura definidos e configurados anteriormente para avaliação e análise das taxas de acerto obtidas.

Figura 22 – Aba Resultados.



Fonte: Elaborado pelo autor.

## 8 Teste e Validação

Todas as possibilidades de escolha feitas pelo usuário foram testadas, a fim de garantir o funcionamento do sistema dentro da previsibilidade. O teste de caixa branca, que visa testar individualmente os módulos, foi utilizado para a validação particular de cada tela do sistema.

As principais preocupações com os módulos tiveram relação com a conversão das imagens dos bancos de dados escolhidos para o formato adequado sem perda de informação. O sistema faz esta conversão disparando os comandos necessários para o sistema operacional remoto do usuário e receber todas as respostas das operações, sem que haja perda de informação. Os arquivos modelos de parâmetros, tanto referente ao algoritmo de otimização, quanto da rede, foram acessados de forma adequada e tiveram as alterações feitas pelo usuário salvas. O sistema garante que o processo seja iniciado e em tempo real, ou seja, no mesmo momento que o usuário requisitou a tarefa. A taxa de reconhecimento final é importada do terminal para a aplicação de forma consistente.

A partir dos testes individuais dos módulos, o teste de caixa preta foi utilizado para garantir a integridade da comunicação e informação transmitida entre eles. A eficácia com que os dados são fornecidos para o sistema foi verificado, assim como a funcionalidade dos componentes do sistema. Alguns testes relacionados a diferentes entradas de dados, por exemplo, também foram realizados neste contexto.

As imagens a seguir demonstram casos em que alguma configuração não foi realizada ou salva adequadamente pelo usuário. O processo de treinamento da rede não é permitido em nenhum desses casos, ou seja, após o surgimento da mensagem de erro, o processo de treinamento não avança até que o erro seja reparado.

A Figura 23 representa o caso em que, ao tentar realizar o treinamento da rede na tela “Treinamento”, uma mensagem de erro informa que a configuração de otimização não foi realizada.

Ja na Figura 24, a mensagem de erro na tela “Treinamento” indica que a base para realizar o treinamento da rede não foi selecionada ou que esta informação não foi armazenada adequadamente.

Uma mensagem de erro, informando que a arquitetura de rede não foi escolhida, surge na quinta tela para indicar que não houve arquitetura de rede escolhida e salva para fazer o treinamento nesta etapa. Esta mensagem de erro é exemplificada pela Figura 25.

Por fim, a Figura 26 representa o caso em que um nome para o arquivo de saída e/ou o número máximo de rodadas para o processo, não foi informado ou teve a alteração salva pelo o usuário.

Figura 23 – Mensagem exibida pela falta de configuração de Otimização.



Fonte: Elaborado pelo autor.

Figura 24 – Mensagem exibida pela falta de seleção de Base.



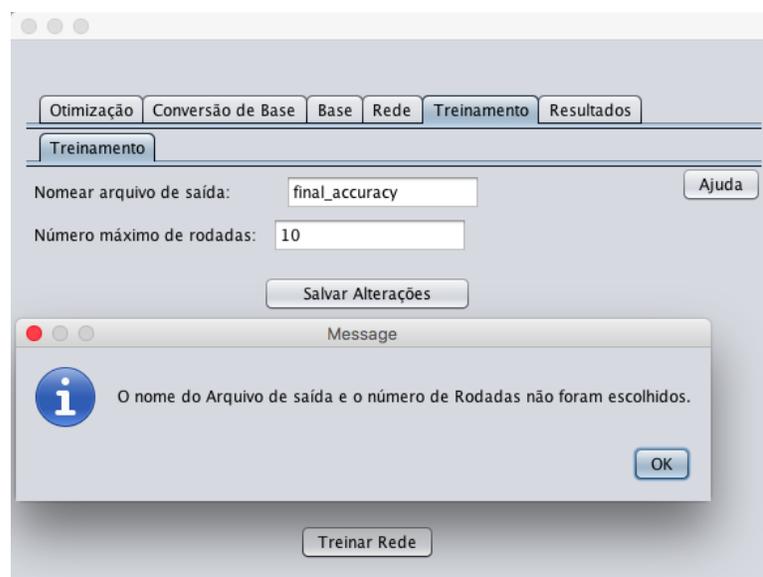
Fonte: Elaborado pelo autor.

Figura 25 – Mensagem exibida pela falta de configuração de Arquitetura.



Fonte: Elaborado pelo autor.

Figura 26 – Mensagem exibida pela falta de Arquivo de saída e número de Rodadas.



Fonte: Elaborado pelo autor.



# 9 Utilização do Sistema para o Reconhecimento de Expressões Faciais e Resultados Obtidos

## 9.1 Metodologia

Esta seção apresenta a metodologia empregada no presente trabalho visando a otimização dos hiper-parâmetros de uma CNN pelo algoritmo MBO, bem alguns detalhes de sua implementação e as configurações para sua experimentação. As bases de dados utilizadas foram descritas previamente no Capítulo 8.

### 9.1.1 Aplicação

Os trabalhos que utilizam CNNs na tarefa de reconhecimento de expressões faciais não se preocupam muito com o fato de escolher os parâmetros da arquitetura neural a ser utilizada. Muito embora vários deles argumentem que uma escolha empírica possa sido realizada, uma heurística automática poderia ser mais eficaz, dado que uma gama maior de valores poderia ser investigada por meio de algoritmos de otimização.

A ideia consistiu, basicamente, em modelar o problema de encontrar um conjunto de parâmetros de uma CNN como sendo uma tarefa de otimização evolucionista. Para tal, foi necessário a definição de uma função objetivo, ou seja, a função que será empregada para guiar os agentes evolucionistas no processo de busca pela melhor solução global. No presente trabalho, essa função foi a taxa de acerto de reconhecimento da expressão facial sobre um conjunto de validação. Desta forma, a base de dados foi então particionada em conjuntos de treinamento, validação e teste: os dois primeiros conjuntos foram utilizados para o aprendizado dos parâmetros da arquitetura neural, sendo o conjunto de teste utilizado apenas para a verificação da taxa de reconhecimento final.

Conforme apresentado na Seção 3.1, uma CNN possui inúmeros parâmetros, tornando a sua escolha um problema de otimização combinatória inviável. No caso de uma otimização evolucionista, cada agente é posicionado aleatoriamente no espaço de busca, que possui  $N$  dimensões, sendo  $N$  o número de parâmetros a serem otimizados. A ideia consistiu em, para cada agente, executar uma CNN para extração de características das imagens dos conjuntos de treinamento e validação, para posterior utilização de uma técnica de classificação de padrões com o intuito de se obter a taxa de acerto na tarefa de reconhecimento de expressões no conjunto de validação. Esses passos são realizados para todas os agentes, que irão então modificar a sua posição (valores a serem otimizados) no espaço de busca seguindo a sua

heurística apropriada. Uma nova iteração é então realizada, com uma nova execução da CNN nos conjuntos de treinamento e validação. Após o processo de convergência da técnica evolucionista, o melhor conjunto de parâmetros, isto é, aquele que maximiza a taxa de acerto do classificador no conjunto de validação, será então utilizado para extração e classificação das imagens do conjunto de teste. A Figura 27 apresenta um diagrama simplista para o procedimento acima descrito.

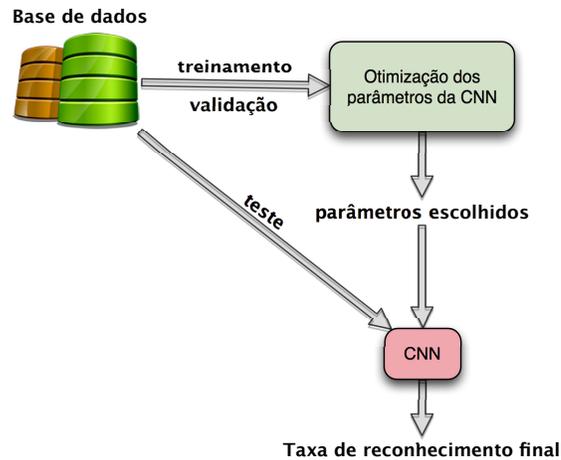


Figura 27 – Arquitetura proposta para os experimentos.

### 9.1.2 Implementação

O algoritmo MBO foi implementado e integrado à LibOPT<sup>1</sup>, uma biblioteca para a otimização de técnicas meta-heurísticas mantida pelo grupo de pesquisa que a aluna pertence. Abaixo, segue a estrutura principal do algoritmo implementado em linguagem C com o auxílio da biblioteca científica GSL<sup>2</sup>, isto é, *GNU Scientific Library*.

```

typedef struct _BirdFlock{
    int m;
    int n;
    int max_iterations;
    int k;
    int X;
    int M;
    int leftSide;
}
  
```

<sup>1</sup> <<https://github.com/jpppsi/LibOPT>>

<sup>2</sup> <http://www.gnu.org/software/gsl/>

```

    gsl_vector **left;
    gsl_vector **right;
    gsl_matrix *x;
    gsl_vector *fitness;
    gsl_matrix *nb_left;
    gsl_vector *nb_fitness_left;
    gsl_matrix *nb_right;
    gsl_vector *nb_fitness_right;
    gsl_vector *LB;
    gsl_vector *UB;
}BirdFlock;

```

Abaixo, é apresentada a explicação de cada elemento da estrutura principal do algoritmo.

- m: número de aves no bando
- n: número de variáveis de decisão (dimensão do problema)
- max\_iterations: número máximo de iterações
- k: números de soluções vizinhas a serem consideradas
- X: número de soluções vizinhas a serem compartilhadas com a iteração seguinte
- M: número de *tours*, ou seja, número de substituições do líder
- leftSide: um identificador para saber qual lado do bando precisa ser mudado
- left: índices das aves do lado esquerdo da ave líder
- right: índices das aves do lado direito da ave líder
- x: soluções possíveis, ou seja, as posições das aves no espaço de busca
- fitness: valor de aptidão
- nb\_left: conjunto de vizinhos da iteração atual utilizado pelas aves da esquerda
- nb\_fitness left: aptidão dos vizinhos da iteração atual das aves da esquerda
- nb\_right: conjunto de vizinhos da iteração atual utilizado pelas aves da direita
- nb\_fitness right: aptidão dos vizinhos da iteração atual das aves da direita birds

- LB: vetor que armazena o limite inferior para cada variável de decisão
- UB: vetor que armazena o limite superior para cada variável de decisão

Apresentam-se, também, algumas funções principais que compõem o algoritmo implementado,

```
void ImproveLeaderSolution(BirdFlock *B, prtFun Evaluate,
int FUNCTION_ID, ...);
void ImproveOtherSolutions(BirdFlock *B, prtFun Evaluate,
int FUNCTION_ID, ...);
void ReplaceLeader(BirdFlock *B);
```

onde:

- ImproveLeaderSolution: objetiva melhorar a ave líder avaliando seus vizinhos
- ImproveOtherSolutions: objetiva melhorar as outras aves do bando avaliando seus vizinhos
- ReplaceLeader: substitui a ave líder pela próxima ave do bando

### 9.1.3 Configuração Experimental

O presente trabalho refere-se à otimização dos hiper-parâmetros de CNNs pelo algoritmo MBO a fim de evitar a escolha aleatória destes e, por conseguinte, objetivar uma melhor taxa de acerto final para o reconhecimento de expressões faciais. Contudo, a escolha dos parâmetros do algoritmo de otimização é naturalmente empírica. Para fins de comparação, foi utilizado o algoritmo Busca Harmônica (*Harmony Search* - HS) (GEEM, 2009), o qual foi implementado pelo aluno de graduação e também membro do grupo de pesquisa *Recogna*, Gustavo Henrique de Rosa (Processo FAPESP #2014/09125-3). Foi realizada a validação cruzada com 10 rodadas. Para as técnicas de otimização, 15 soluções no espaço de busca foram consideradas (aves e harmonias) em 50 iterações para o HS e 25 iterações para o MBO como critério de convergência. Essa diferença é compensada pelo número de *tours* (2) que o MBO realiza. Sendo assim, mais chamadas ao algoritmo da CNN são realizadas. Uma análise estatística foi obtida a partir do teste de Wilcoxon (WILCOXON, 1945). A Tabela 1 apresenta as configurações dos parâmetros de cada uma das técnicas de otimização empregadas neste trabalho.

Tais configurações foram escolhidas levando em conta o tempo que o algoritmo de otimização HS levou para ser executado, sendo que este é computacionalmente mais simples,

Tabela 1 – Configuração dos parâmetros.

Técnica	Parâmetros
MBO	$k = 5, X = 2, M = 2$
HS	$HMCR = 0.7, PAR = 0.5, \rho = 0.1$

do que o algoritmo do presente projeto, o MBO. Acredita-se que uma melhor escolha de parâmetros do MBO aumentaria a taxa de acerto final dos resultados, mas seria inviável dentro período de tempo disponível para tal etapa de experimentação. Vale destacar que os parâmetros HMCR (*Harmony Memory Considering Rate*) e PAR (*Pitch Adjusting Rate*) são utilizados pela técnica HS com o objetivo de evitar ótimos locais.

Para as técnicas de aprendizado de máquina, foi utilizada a implementação da biblioteca Caffe, descrita na Seção ??, que se mostrou eficiente por ser desenvolvida sob a plataforma GPGPU (*General Purpose computing on Graphics Processor Units*).

## 9.2 Resultados Experimentais

Nesta Seção, são apresentados os resultados experimentais obtidos com a otimização por técnicas meta-heurísticas dos hiper-parâmetros de uma Rede Neural por Convolução. Foram comparados os resultados obtidos pela Caffe e suas configurações de treinamento normais, ou seja, sem otimização, com a Caffe sendo otimizada pelos algoritmos MBO e HS. Para todas as bases de dados utilizadas neste projeto, empregou-se a arquitetura proposta da base ImageNET, ou seja, a CaffeNet<sup>3</sup>. A arquitetura ImageNET se faz necessária, uma vez que as bases escolhidas não estão no modelo aceito como entrada para a Caffe. Portanto, as bases foram convertidas para o formato necessário com o auxílio de ferramentas fornecidas pela própria arquitetura. Nas próximas subseções, são detalhados os resultados experimentais para cada conjunto de dados escolhido.

### 9.2.1 Base de Dados JAFFE

Nesta seção, são apresentados e discutidos os resultados obtidos sobre a base de dados *Japanese Female Facial Expression*. A Tabela 2 apresenta a acurácia média e o desvio padrão sobre o conjunto de teste utilizando os melhores parâmetros encontrados pelos algoritmos de otimização utilizados, bem como o número de chamadas para o procedimento de aprendizado da CNN. Foi destacado em negrito a técnica com melhor acurácia.

A Tabela 3 apresenta os resultados do Teste de Wilcoxon (WILCOXON, 1945) considerando a acurácia final média sobre o conjunto de testes, onde qual o símbolo “=” denota as

<sup>3</sup> <http://caffe.berkeleyvision.org/gathered/examples/imagenet.html>

Tabela 2 – Resultados experimentais sobre a base de dados JAFFE.

Técnica	Acurácia Final (conjunto de teste)	#chamadas
Caffe	29,29% $\pm$ 0.03	1
<b>MBO</b>	<b>34,05% <math>\pm</math> 0.09</b>	<b>65</b>
HS	24,74% $\pm$ 0.04	65

técnicas que foram consideradas similares, e “ $\neq$ ” denota as técnicas que foram distintas em relação às outras. A partir dos resultados obtidos experimentalmente nesta base, é possível notar que enquanto a acurácia do HS ficou abaixo da acurácia atingida sem otimização pela Caffe, a técnica MBO, por outro lado, apresentou desempenho melhor quando comparada à Caffe e HS. Conclui-se, então, que a otimização de parâmetros de uma Rede Neural por Convolução se faz válida para este conjunto de dados. Nota-se também que, apesar de o MBO ter atingido a melhor taxa de acerto, o algoritmo apresentou um desvio padrão alto, fator que caracterizou os resultados da Caffe e do MBO como sendo semelhantes na Tabela 3. De maneira geral, MBO pode ser considerada uma técnica de otimização adequada para este conjunto de dados.

Tabela 3 – Resultados do Teste de Wilcoxon sobre a base de dados JAFFE.

Técnica	Caffe	MBO	HS
Caffe			
MBO	=		
HS	$\neq$	$\neq$	

### 9.2.2 Base de Dados Cohn-Kanade AU-Coded Expression

Nesta Seção, os resultados obtidos sobre a base de dados *Cohn-Kanade AU-Coded Expression* são apresentados e discutidos. A Tabela 4 exibe a acurácia média e o desvio padrão obtidos pela análise dos dados da base utilizando os melhores parâmetros encontrados pelos algoritmos de otimização, bem como o número de chamadas de aprendizado da CNN. As melhores taxas de acerto foram destacadas em negrito.

Com as configurações experimentais utilizadas, o HS obteve o melhor resultado para esse conjunto de dados. Embora as técnicas tenham sido dadas como semelhantes pelo Teste de Wilcoxon, verifica-se que há uma necessidade de dar uma melhor atenção e empregar um número maior de iterações no MBO, visto que este apresentou resultados inferiores aos obtidos pelo treinamento da Caffe, que internamente executa algumas operações de otimização.<sup>4</sup> O

<sup>4</sup> <http://caffe.berkeleyvision.org/tutorial/solver.html>

Tabela 4 – Resultados experimentais obtidos sobre a base de dados Cohn-Kanade AU-Coded Expression.

Técnica	Acurácia Final (conjunto de teste)	#chamadas
Caffe	67,18% $\pm$ 0.06	1
MBO	65,35% $\pm$ 0.05	65
<b>HS</b>	<b>68,51% <math>\pm</math> 0.05</b>	<b>65</b>

Tabela 5 – Resultados do Teste de Wilcoxon sobre a base de dados Cohn-Kanade AU-Coded Expression.

Técnica	Caffe	MBO	HS
Caffe			
MBO	=		
HS	=	=	

destaque obtido pelo HS é um indicativo de que algoritmos otimização, de um modo geral, são técnicas promissoras para melhorar a taxa de acerto de métodos de reconhecimento de padrões.

### 9.2.3 Base de Imagens Taiwanese Facial Expression

Os resultados do experimentos obtidos com a base de imagens *Taiwanese Facial Expression* são apresentados e discutidos nesta Seção. A Tabela 6 apresenta a acurácia média, o desvio padrão e o número de chamadas para o procedimento de aprendizado da CNN para este conjunto de dados, considerando que foram utilizados os melhores parâmetros encontrados pelos algoritmos de otimização. As técnicas com melhor acurácia são apresentadas em negrito.

Tabela 6 – Resultados experimentais obtidos sobre a base de dados Taiwanese Facial Expression.

Técnica	Acurácia Final (conjunto de teste)	#chamadas
Caffe	91,17% $\pm$ 0.01	1
<b>MBO</b>	<b>93,40% <math>\pm</math> 0.02</b>	<b>65</b>
HS	92,11% $\pm$ 0.01	65

Baseado no Teste de Wilcoxon, pode-se observar que o MBO foi estatisticamente mais eficiente do que os métodos de otimização implementados pela biblioteca Caffe e o HS, mesmo com um número reduzido de iterações. Pode-se concluir, portanto, que o método de

Tabela 7 – Resultados do Teste de Wilcoxon sobre a base de dados Taiwanese Facial Expression.

Técnica	Caffe	MBO	HS
Caffe			
MBO	$\neq$		
HS	$=$	$=$	

otimização utilizado, o MBO, é um método promissor para a otimização dos hiperparâmetros de uma CNN.

## 10 Conclusões

Redes Neurais por Convolução têm obtido bons resultados quanto ao reconhecimento de dígitos, objetos e processamento de linguagem natural. O empecilho para sua aplicação consiste na escolha adequada dos seus parâmetros para que sua eficácia seja significativa no contexto de aprendizado em profundidade. Paralelamente, técnicas evolucionistas têm sido empregadas para a otimização e seleção de parâmetros em geral, mas muito pouco utilizadas em arquiteturas de aprendizado em profundidade.

Dentre as inúmeras técnicas que poderiam ser empregadas para otimização de parâmetros em aprendizado de máquina, escolheu-se investir no algoritmo de Otimização das Aves Migratórias por este utilizar um mecanismo de beneficiamento entre as possíveis soluções do problema baseado no comportamento de um bando de aves migratórias. Não havia notícia ainda da aplicação de tal técnica no contexto de otimização de CNNs.

Desta forma, o presente projeto propôs o desenvolvimento de um sistema que viabilizasse a aplicação da técnica MBO para a otimização dos hiper-parâmetros das CNNs. O processo, até então feito por linha de comando, foi substituído por um sistema de interface amigável, que foi responsável por coordenar a metodologia empregada pelo usuário. Foram validados os resultados no contexto de reconhecimento de expressões faciais em 3 bases de dados públicas. Adicionalmente, esses resultados foram comparados com o algoritmo Busca Harmônica.

Os resultados experimentais obtidos confirmam a proposta deste trabalho, ou seja, indicam que a elaboração de um sistema que possa auxiliar na otimização de redes neurais por convolução, bem como a técnica escolhida, MBO, é passível de ser empregada para a otimização dos parâmetros de uma CNN. Acredita-se que o presente trabalho de conclusão de curso contribui para os estudos a cerca da técnica de aprendizado de máquina aqui explorada, além de simplificar e abstrair um processo trabalhoso de configuração e integração manual das várias bibliotecas por parte dos usuários para cada algoritmos de otimização evolucionista aplicado a bases de dados distintas. Em trabalhos futuros, a interface visual do sistema pode ser aperfeiçoada de forma que novas técnicas de otimização sejam adicionadas à interface, bem como um possível armazenamento das bases convertidas em um banco de dados, para que o sistema esteja sempre atualizado em sua interface. Assim, a intenção é de que o sistema facilite ainda mais a compreensão dos processos relacionados à otimização e rede neural utilizada, e que o usuário consiga obter um maior aproveitamento do sistema, no que diz respeito ao uso regular do mesmo no processo de otimização de Redes Neurais por Convolução.

Cabe destacar, ainda, que o projeto intitulado “Otimização de Redes Neurais por Convolação e sua Aplicação para o Reconhecimento de Expressões Faciais” referente ao processo FAPESP #2014/12593-9 foi de grande importância para o estudo e embasamento teórico necessários para o desenvolvimento e motivação deste trabalho de conclusão de curso.

# Referências

- B., L. P.; A., S. C. Formation flight of birds. *Science*, v. 3934, n. 168, p. 1003–1005, 1970.
- COX, D.; PINTO, N. Beyond simple features: A large-scale feature search approach to unconstrained face recognition. In: *Proceedings of the IEEE International Conference on Automatic Face Gesture Recognition and Workshops*. [S.l.: s.n.], 2011. p. 8–15.
- DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification (2nd Edition)*. [S.l.]: Wiley-Interscience, 2000. ISBN 0471056693.
- DUMAN, E.; UYSAL, M.; ALKAYA, A. Migrating birds optimization: A new metaheuristic approach and its performance on quadratic assignment problem. *Information Sciences*, Elsevier Science Inc., New York, NY, USA, v. 217, p. 65–77, 2012.
- FEDOROVICI, L. et al. Embedding gravitational search algorithms in convolutional neural networks for OCR applications. In: *7th IEEE SACI*. [S.l.: s.n.], 2012. p. 125–130.
- FUKUSHIMA, K.; MIYAKE, S. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, v. 15, n. 6, p. 455–469, 1982.
- GEEM, Z. W. *Music-Inspired Harmony Search Algorithm: Theory and Applications*. 1st. ed. [S.l.]: Springer Publishing Company, Incorporated, 2009. ISBN 364200184X, 9783642001840.
- GEISLER, W.; ALBRECHT, D. Cortical neurons: isolation of contrast gain control. *Vision Research*, v. 32, n. 8, p. 1409–1410, 1992.
- HINTON, G. Training products of experts by minimizing contrastive divergence. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 14, n. 8, p. 1771–1800, 2002. ISSN 0899-7667.
- JIA, Y. et al. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.
- LECUN, Y. et al. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, MIT Press, Cambridge, MA, USA, v. 1, n. 4, p. 541–551, 1989.
- LECUN, Y. et al. Advances in neural information processing systems 2. In: TOURETZKY, D. S. (Ed.). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1990. cap. Handwritten Digit Recognition with a Back-propagation Network, p. 396–404.
- LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, n. 11, p. 2278–2324, 1998.
- LTD, B. D. I. T. C. *Processon*. [S.l.]: Processon.com, 2014. <<https://www.processon.com>>.
- MARANA, A.; PAPA, J. *Recogna*. [S.l.]: FC.Unesp, 2010. <<http://www2.fc.unesp.br/~papa/recogna/index.html>>.

- OPENGROUP, T. *Unix*. [S.l.]: opengroup.org, 1995. <<http://opengroup.org/unix>>.
- ORACLE. *NetBeans IDE*. [S.l.]: NetBeans.org, 2016. <[https://netbeans.org/index\\_pt\\_BR.html](https://netbeans.org/index_pt_BR.html)>.
- PAPA, J. *LibOPT-plus*. [S.l.]: GitHub, 2016. <<https://github.com/jppbsi/LibOPT-plus>>.
- ROSA, G. *LibOPT4Caffe*. [S.l.]: GitHub, 2016. <<https://github.com/gugarosa/LibOPT4Caffe>>.
- ROSARIO, S. *Facial Emoticons: Reprodu de Informa Associada a Expresses Faciais por Via do seu Reconhecimento*. [S.l.]: Universidade Tica de Lisboa, 2018. <<https://fenix.tecnico.ulisboa.pt/cursos/meic-a/dissertacao/2353642189428>>.
- TEIXEIRA, A. *Desenvolvimento de uma Interface Grca para Classificadores de Imagem*. [S.l.]: DocPlayer, 2011. <<http://docplayer.com.br/712170-Desenvolvimento-de-uma-interface-grfica-para-classificadores-de-imagem.html>>.
- TEIXEIRA, J. P. et al. *Identifica de Caracteres com Rede Neuronal Artificial com Interface Grca*. [S.l.]: Instituto Politico de Bragan 2009. <<http://hdl.handle.net/10198/1868>>.
- WILCOXON, F. Individual comparisons by ranking methods. *Biometrics Bulletin*, v. 1, n. 6, p. 80–83, 1945.