

**GIOVANA OLIVEIRA DE LUCCA**

**RECONHECIMENTO DE EXPRESSÕES FACIAIS GRAMATICAS DA  
LÍNGUA BRASILEIRA DE SINAIS COM APRENDIZADO DE MÁQUINA**

Trabalho de Conclusão de Curso apresentado  
à banca avaliadora do Curso de Engenharia  
de Computação, da Escola Superior de  
Tecnologia, da Universidade do Estado do  
Amazonas, como pré-requisito para obtenção  
do título de Engenheiro de Computação.

Orientador(a): Profa. Dra. Elloá B. Guedes da Costa

Manaus – Novembro – 2018

***Universidade do Estado do Amazonas - UEA***  
***Escola Superior de Tecnologia - EST***

*Reitor:*

***Cleinaldo de Almeida Costa***

*Vice-Reitor:*

***Cleto Cavalcante Leal***

*Diretor da Escola Superior de Tecnologia:*

***Roberto Higino Pereira da Silva***

*Coordenador do Curso de Engenharia de Computação:*

***Salvador Ramos Bernardino da Silva***

*Coordenador da Disciplina Projeto Final:*

***Márcia Sampaio Lima***

*Banca Avaliadora composta por:*

*Data da Defesa: 30 /11/2018.*

***Profa. Elloá Barreto Guedes da Costa, D.Sc. (Orientador(a))***

***Prof. Luis Cuevas Rodriguez, D.Sc.***

***Prof. Rodrigo Tavares Teixeira, M.Sc.***

### **CIP – Catalogação na Publicação**

DE LUCCA, Giovana Oliveira

Reconhecimento de expressões faciais gramaticais da Língua Brasileira de Sinais com Aprendizado de Máquina [orientado por] Profa. Dra. Elloá Barreto Guedes da Costa – Manaus: UEA, 2018.

63 p.: 19 il.; 30cm

Inclui Bibliografia

Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação).  
Universidade do Estado do Amazonas, 2018.

CDU: \_\_\_\_\_

GIOVANA OLIVEIRA DE LUCCA

**RECONHECIMENTO DE EXPRESSÕES FACIAIS GRAMATICAIIS DA  
LÍNGUA BRASILEIRA DE SINAIS COM APRENDIZADO DE MÁQUINA**

Trabalho de Conclusão de Curso apresentado  
à banca avaliadora do Curso de Engenharia  
de Computação, da Escola Superior de  
Tecnologia, da Universidade do Estado do  
Amazonas, como pré-requisito para obtenção  
do título de Engenheiro de Computação.

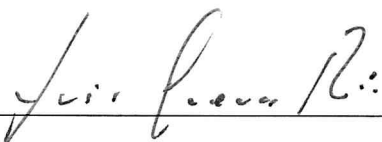
**Aprovado em: 30/11/2018.**

BANCA EXAMINADORA



**Profa. Dra. Elloá B. Guedes da Costa**

*UNIVERSIDADE DO ESTADO DO AMAZONAS*



**Prof. Prof. Luis Cuevas Rodriguez, D.Sc.**

*UNIVERSIDADE DO ESTADO DO AMAZONAS*



**Prof. Rodrigo Tavares Teixeira, M.Sc.**

*UNIVERSIDADE DO ESTADO DO AMAZONAS*

# Resumo

Este trabalho aborda o reconhecimento de expressões faciais gramaticais da Língua Brasileira de Sinais (Libras) com Aprendizado de Máquina. Para tanto, um conjunto de dados composto por expressões faciais gramaticais anotadas a partir de 225 vídeos nos quais dois indivíduos emitem sentenças em Libras foi utilizado em uma tarefa de classificação multi-classe, base de dados esta já existente na literatura. Nesta tarefa foram propostos diversos modelos de Aprendizado de Máquina afim de obter um modelo que se adequasse ao problema em questão. Nove algoritmos de Aprendizado de Máquina distintos foram submetidos cada um à uma busca em *grid* variando diversos parâmetros e hiperparâmetros, produzindo diferentes modelos. Os modelos foram treinados e testados segundo a validação cruzada  $k$ -fold, com  $k = 3$ , utilizando a métrica de avaliação  $F$ -score do tipo *micro-averaging*, em virtude das diferentes classes serem desbalanceadas. Ao final das buscas, foram elencados os melhores modelos, em que destacou-se uma Floresta Aleatória com desempenho igual a 0.903, resultado superior a todos os demais. Este modelo foi então submetido a outras perspectivas de análise para uma comparação com a literatura, as quais sugerem melhorias no estado da arte para este problema.

**Palavras-Chave:** Expressões Faciais Gramaticais, Língua Brasileira de Sinais, Aprendizado de Máquina.

# ***Abstract***

This work addresses the Brazilian Sign Language grammatical facial expressions recognition task by using Machine Learning. For this purpose, a dataset available in the literature composed of grammatical facial expressions collected from 225 videos was used. In each video, two individuals make phrases in the Brazilian Sign Language. This problem was tackled through a multi-class classification task. Available Machine Learning models were proposed to handle the referred problem. Nine different Machine Learning algorithms were submitted to a grid search, changing parameters and hyperparameters, producing different models. Such models were trained and tested with  $k$ -fold cross validation method, with  $k = 3$ , using a  $F$ -score metric of the micro-averaging type, since dataset classes were unbalanced. At the end of the grid searches, the best models were listed, in which the Random Forest was highlighted with a performance equals to 0.903, a result superior to all others. This model was then submitted to an analysis for a comparison with the literature, which suggest improvements in the state of the art for this problem.

**Keywords:** *Grammatical Facial Expressions, Brazilian Sign Language, Machine Learning.*

# Agradecimentos

Agradeço primeiramente à Deus por sempre me acompanhar e por me proporcionar ter chegado até aqui. Aos meus familiares, especialmente meus pais, agradeço por todo apoio desde o princípio em todos os momentos. Agradeço também aos meus amigos por sempre acreditarem em mim. À minha orientadora, um muito obrigado especial por todos os ensinamentos durante toda minha graduação.

Agradeço também à Fundação de Amparo à Pesquisa do Estado do Amazonas (FAPEAM) que, por meio do Projeto PROTI Pesquisa 11/2017, colaborou para a consolidação da infraestrutura física e tecnológica do Laboratório de Sistemas Inteligentes da Escola Superior de Tecnologia da Universidade do Estado do Amazonas. Este trabalho de conclusão de curso é um dos produtos deste projeto, pois foi desenvolvido no referido laboratório, fez uso dos recursos computacionais ali disponíveis e foi melhorado graças às discussões e interações com o grupo de pesquisa nele sediado.

## **Epígrafe**

*Nenhuma alta sabedoria pode ser atingida sem uma dose de sacrifício.*

C. S. Lewis

As Crônicas de Nárnia: O Sobrinho do Mago

# Sumário

|   |           |
|---|-----------|
| <b>Lista de Tabelas</b>   | <b>ix</b> |
| <b>Lista de Figuras</b>   | <b>xi</b> |
| <b>1 Introdução</b>   | <b>1</b>  |
| 1.1 Objetivos . . . . .   | 3         |
| 1.2 Justificativa . . . . .   | 3         |
| 1.3 Metodologia . . . . .   | 4         |
| 1.4 Organização do Documento . . . . .  | 5         |
| <b>2 Fundamentação Teórica</b>  | <b>6</b>  |
| 2.1 Língua Brasileira de Sinais . . . . .                                     | 6         |
| 2.1.1 Expressões Faciais Gramaticais na Língua Brasileira de Sinais . . . . . | 8         |
| 2.2 Aprendizado de Máquina . . . . .  | 10        |
| 2.2.1 Árvores de Decisão . . . . .  | 12        |
| 2.2.2 Redes Neurais Artificiais . . . . .                                     | 13        |
| 2.2.3 Métodos <i>Ensemble</i> . . . . .                                       | 17        |
| 2.2.4 Máquinas Vetores de Suporte . . . . .                                   | 19        |
| 2.2.5 Vizinhos mais Próximos . . . . .  | 20        |
| 2.3 Tecnologias Utilizadas . . . . .  | 21        |
| 2.4 Trabalhos Relacionados . . . . .  | 22        |
| <b>3 Solução Proposta</b>   | <b>27</b> |



|          |   |           |
|----------|---|-----------|
| 3.1      | Tarefa de Aprendizado de Máquina . . . . .      | 27        |
| 3.2      | Visão Geral do Conjunto de Dados . . . . .      | 28        |
| 3.3      | Modelos, Parâmetros e Hiperparâmetros . . . . . | 33        |
| 3.4      | Implementação da Solução Proposta . . . . .     | 36        |
| <b>4</b> | <b>Resultados e Discussão</b>                   | <b>38</b> |
| 4.1      | Métricas Obtidas . . . . .                      | 38        |
| 4.2      | Análise Comparativa com a Literatura . . . . .  | 41        |
| <b>5</b> | <b>Considerações Finais</b>                     | <b>46</b> |

# Lista de Tabelas

|     |   |    |
|-----|---|----|
| 2.1 | Tipos de estrutura de sentenças quanto à sintaxe das expressões faciais e exemplos correspondentes. Elaborado a partir de: (QUADROS; KARNOPP, 2004; RAMOS, 2004; SOUSA, 2010) . . . . .   | 9  |
| 3.1 | Estatística descritiva dos 17 pontos selecionados. Os símbolos $\bar{\mathbf{x}}, \tilde{\mathbf{x}}$ e $\sigma_{\mathbf{x}}$ denotam a média, mediana e desvio padrão da coordenada $x$ , respectivamente. A notação é análoga para a coordenada $y$ . . . . . | 32 |
| 4.1 | Melhor $F$ -score obtido em cada grupo de modelos. . . . .  | 40 |

# Lista de Figuras

|      |  |    |
|------|--|----|
| 2.1  | Alfabeto em Libras. Fonte: (RODRIGUES; VALENTE, 2011). . . . .   | 8  |
| 2.2  | Exemplo de sinal em Libras. Fonte: (RODRIGUES; VALENTE, 2011). . . . .   | 8  |
| 2.3  | Exemplos de Expressões Faciais Gramaticais. Fonte: (SOUSA, 2010). . . . .  | 9  |
| 2.4  | Hierarquia de aprendizado. Fonte: (FACELI et al., 2011). . . . .   | 11 |
| 2.5  | Neurônio artificial: a combinação linear das entradas $x$ ponderadas pelos pesos $w$ é transformada pela função de ativação $f$ na saída $y$ . . . . .                           | 14 |
| 2.6  | Exemplos de diferentes funções de ativação. . . . .  | 14 |
| 2.7  | Papel desempenhado pelos neurônios das diferentes camadas de uma rede MLP. Fonte: (FACELI et al., 2011). . . . .   | 16 |
| 2.8  | Método <i>Ensemble</i> : fornecer um limite de decisão mais complexo. Fonte: (MARS-LAND, 2015). . . . .  | 18 |
| 2.9  | Exemplo de SVM. . . . .  | 20 |
| 2.10 | Impacto do valor de $k$ no algoritmo $k$ -NN. Fonte: (FACELI et al., 2011). . . . .  | 22 |
| 2.11 | Histograma com os $F$ -scores obtidos por (FREITAS; BARBOSA; PERES, 2014b) e (SILVA et al., 2017) para cada expressão facial gramatical. . . . .                                 | 26 |
| 3.1  | Pontos da face marcados para construção do conjunto de dados. Fonte: (FREITAS; BARBOSA; PERES, 2014a). . . . .   | 29 |
| 3.2  | <i>Boxplot</i> da distribuição das coordenadas $z$ de profundidade referentes aos 17 pontos escolhidos. . . . .  | 31 |
| 3.3  | <i>Boxplot</i> da distribuição das coordenadas $z$ de profundidade referentes aos 17 pontos escolhidos e considerado apenas as expressões faciais gramaticais negativas. . . . . | 31 |

|     |   |    |
|-----|---|----|
| 3.4 | Histograma com a quantidade de exemplos por classe. . . . .   | 32 |
| 4.1 | $F$ -scores obtidos em cada modelo. . . . .   | 39 |
| 4.2 | $F$ -scores obtidos por (FREITAS; BARBOSA; PERES, 2014b) e (SILVA et al., 2017) para cada expressão facial gramatical e $F$ -score alcançado utilizando o método <i>holdout</i> no re-treino. . . . . | 42 |
| 4.3 | Matrizes de confusão por expressão facial gramatical na estratégia Um-Contra-Todos. . . . .   | 43 |
| 4.4 | $F$ -scores obtidos após realizar estratégia Um-Contra-Todos. . . . .   | 44 |
| 4.5 | $F$ -scores obtidos após refinamento por modelo. . . . .  | 45 |

# Capítulo 1

## Introdução

A *Língua Brasileira de Sinais* (Libras) é uma linguagem gesto-visual segundo a qual uma mensagem é emitida por meio de movimentos nas mãos, corpo e face e compreendida por meio da visão (BRECAILO, 2012). Esta língua é um caminho para a abertura social de pessoas surdas, surdo-cegas e com outros tipos de deficiência auditiva, tendo sido reconhecida pela Lei Federal 10.436 em 2002 (BRASIL, 2002; FLORES; BARBOSA; RIGO, 2012).

A Libras, na sua condição de língua natural, é tão complexa e sofisticada quanto qualquer língua oral-auditiva e, portanto, carrega níveis linguísticos diferentes, tais como fonologia, morfologia, sintaxe e semântica. Sendo assim, os elementos chamados sinais são equivalentes aos itens lexicais de uma língua oral-auditiva e, juntamente com as expressões faciais e corporais, corroboram para o entendimento da informação. As expressões faciais, em particular, podem ser afetivas ou gramaticais. Quando possuem caráter gramatical, as chamadas *expressões faciais gramaticais* podem ser de nove diferentes categorias e conferem informação gramatical a uma sentença expressa em sinais, complementando o seu sentido (QUADROS, 2017).

Atenuar os obstáculos impostos diariamente às pessoas surdas e/ou com deficiência auditiva e aumentar a integração destas na sociedade majoritariamente ouvinte é uma necessidade fundamental (TEODORO, 2015). Nesta perspectiva, as tecnologias atuais podem corroborar com este cenário, em que meios de tradução automática podem ser integrados e desenvolvidos para dispositivos tais como *tablets* e *smartphones*. De maneira simples, almeja-se que num futuro breve seja possível utilizar um *smartphone* para capturar gestos emitidos em Libras e

então traduzi-los para texto escrito, permitindo assim a captura da mensagem emitida por uma pessoa surda, sem a necessidade de intérpretes ou a utilização de outras linguagens.

Além disso, esta ideia também poderia vir a ser utilizada para verificar automaticamente o conteúdo de uma informação que está sendo transmitida em Libras. Eventualmente, durante um diálogo, poderia ajudar a corrigir sinais emitidos, o que seria útil para iniciantes em Libras, por exemplo, e em um cenário mais amplo, poderia evitar a adulteração da comunicação por malevolência. Um caso ocorrido em Setembro de 2018 no estado do Espírito Santo retrata este último cenário. Um intérprete de Libras dos programas eleitorais foi acusado de inventar sinais durante a tradução. Segundo a Associação dos Surdos, o intérprete utilizou sinais desconexos e sem contexto, desrespeitando a estrutura gramatical da língua de sinais (G1, 2018). Um outro caso similar ocorreu na Ucrânia em 2004 quando um intérprete adulterou as informações traduzidas sobre a eleição da presidência, deixando o governo da Ucrânia em total desordem (JOURNAL, 2004).

Considerando a abrangência e os desafios envolvidos na tradução automática de Libras, é essencial que se inicie endereçando esta tarefa grande e complexa a partir de tarefas mais elementares. Assim, no escopo deste trabalho, decidiu-se, portanto, considerar os diferentes tipos de expressões faciais gramaticais, as quais utilizam, dentre outros, movimentos de olhos, sobrancelhas e boca para complementar ou significar os gestos emitidos com o restante do corpo.

A solução proposta utilizou-se de dados reais oriundos de dois sujeitos emitindo diferentes expressões faciais gramaticais em Libras com vistas a classificar a expressão correspondente, dentre um conjunto de nove categorias. Este problema foi endereçado como uma tarefa de classificação no paradigma de aprendizado supervisionado para a qual foram utilizados diferentes modelos de Aprendizado de Máquina, configurados com diferentes parâmetros e hiperparâmetros segundo uma busca em *grid*. Procedimentos de validação dos resultados foram utilizados e buscou-se a generalização na classificação produzida por estes modelos, aferida com métricas de desempenho apropriadas.

Ao longo desta introdução serão mostrados os demais elementos que guiaram a condução de atividades deste trabalho. A Seção 1.1 contempla os objetivos inicialmente propostos para

o desenvolvimento do projeto. Na Seção 1.2 são apresentadas as justificativas que motivaram a realização do trabalho em questão. A metodologia adotada encontra-se detalhada na Seção 1.3 e, por fim, a Seção 1.4 dispõe da organização do restante do documento.

## 1.1 Objetivos

O objetivo geral deste trabalho consistiu em classificar expressões faciais gramaticais da Língua Brasileira de Sinais com Aprendizado de Máquina. Para alcançar esta meta, alguns objetivos específicos precisaram ser alcançados, a citar:

1. Efetuar a fundamentação teórica dos conceitos relativos ao Aprendizado de Máquina;
2. Consolidar uma base de dados de expressões faciais gramaticais;
3. Descrever o problema da classificação de expressões faciais gramaticais segundo uma tarefa de Aprendizado de Máquina;
4. Propor, treinar e testar diferentes modelos para a tarefa considerada;
5. Avaliar os resultados obtidos.

## 1.2 Justificativa

O reconhecimento automático da língua de sinais é uma importante área de pesquisa que tem como objetivo atenuar os obstáculos impostos no dia a dia das pessoas surdas e/ou com deficiência auditiva e aumentar a integração destas pessoas na sociedade majoritariamente ouvinte (TEODORO, 2015). Para elaboração de soluções neste domínio, é essencial, portanto, considerar também a análise das expressões faciais gramaticais, perspectiva considerada no escopo deste trabalho.

Além do que foi exposto, a proposta considerada neste trabalho incentiva a prática de conceitos, técnicas e tecnologias de uma área emergente da Computação, contribuindo na formação profissional da aluna concluinte. No mais, esta proposta de trabalho de conclusão de curso está

alinhada com as atividades desenvolvidas pelo Grupo de Pesquisas em Sistemas Inteligentes e do Laboratório de Sistemas Inteligentes, iniciativas promovidas por docentes do Núcleo de Computação (NUCOMP) da Escola Superior de Tecnologia (EST) da Universidade do Estado do Amazonas (UEA), motivando o desenvolvimento de soluções inovadoras que utilizam técnicas emergentes do Aprendizado de Máquina.

## 1.3 Metodologia

Para atingir os objetivos propostos no escopo deste trabalho, a condução das atividades obedeceu à metodologia apresentada a seguir, composta dos seguintes passos:

1. Estudo dos conceitos relacionados à Aprendizado de Máquina;
2. Estudo do ferramental tecnológico para elaboração e execução de projetos de Aprendizado de Máquina, incluindo Python, Sci-kit Learn, Pandas, Google Cloud Platform, dentre outros;
3. Pesquisar uma base de dados representativa de expressões faciais gramaticais;
4. Descrever o problema como uma tarefa de Aprendizado de Máquina, estabelecendo os atributos, tipo de tarefa, forma de avaliação e métricas de desempenho;
5. Efetuar a limpeza e organização da base de dados;
6. Propor diferentes modelos para a tarefa considerada;
7. Propor uma busca em *grid* para os valores dos parâmetros e hiperparâmetros dos modelos considerados;
8. Treinar os modelos com os exemplos da base de dados;
9. Testar os modelos e coletar métricas de desempenho;
10. Analisar os resultados obtidos identificando os modelos mais adequados ao cenário considerado.



## 1.4 Organização do Documento

Para apresentar o presente trabalho de conclusão de curso, este documento está organizado como segue. O Capítulo 2 contempla os fundamentos teóricos que foram necessários para elaboração do projeto, incluindo conceitos de Libras, Aprendizado de Máquina e os modelos de Aprendizado de Máquinas utilizados. A solução proposta para o tema de classificação de expressões faciais gramaticais da Língua Brasileira de Sinais com Aprendizado de Máquina é abordada no Capítulo 3. Os resultados obtidos são explanados no Capítulo 4. Por fim, as considerações finais do trabalho encontram-se no Capítulo 5.

# Capítulo 2

## Fundamentação Teórica

Nesta seção serão apresentados os fundamentos teóricos que deram suporte à realização deste trabalho. A definição da Língua Brasileira de Sinais, seus principais conceitos, características e estatísticas são percorridas na Seção 2.1. Os conceitos elementares sobre as técnicas de Aprendizado de Máquina e os conceitos, métodos, aplicações e outros detalhes sobre diversos modelos de Aprendizado de Máquina os quais foram utilizados neste trabalho encontram-se na Seção 2.2. As tecnologias utilizadas na realização deste trabalho são apresentadas na Seção 2.3. Por fim, alguns trabalhos relacionados encontrados na literatura a respeito da classificação de expressões faciais gramaticais da Libras são explanados na Seção 2.4.

### 2.1 Língua Brasileira de Sinais

A Língua Brasileira de Sinais (Libras) é a língua de sinais reconhecida como meio legal de comunicação e expressão dos surdos no Brasil desde 2002 por meio da sanção da Lei nº 10436. A Libras, na sua condição de língua natural, é tão complexa e sofisticada quanto qualquer outra língua oral-auditiva, apresentando as mesmas propriedades linguísticas e diferindo apenas pela sua característica visuoespacial (RODRIGUES; VALENTE, 2011; QUADROS, 2017). Segundo o Censo de 2010 realizado pelo Instituto Brasileiro de Geografia e Estatística (IBGE), o Brasil possui 9,7 milhões de deficientes auditivos e cerca de 20% apresentam deficiência auditiva severa (IBGE, 2010).

Os registros históricos sobre Libras relatam que o imperador Dom Pedro II contratou o conde e professor surdo E. Huet para a primeira instituição educacional para surdos no Brasil no Rio de Janeiro. Huet era um professor surdo francês que trazia consigo a Língua de Sinais Francesa (LSF) e, portanto, a conclusão que se tem é que a LSF teve total influência na constituição da Libras, pois os registros indicam que além da presença de Huet como professor, alguns materiais foram traduzidos (ou adaptados) do francês para o português incluindo registros das línguas de sinais (QUADROS, 2017).

Desde 2002, ano do reconhecimento como a língua de sinais nacional, a Libras projetou-se como importante ferramenta na inclusão de pessoas surdas e com deficiências auditivas na sociedade. Em 2005, foi criado o Decreto 5.626, o qual regulamentou as políticas linguísticas relacionadas à Libras. Uma das primeiras ações regidas pelo decreto foi a criação do curso de Letras Libras em nível de graduação para formação de licenciados ou bacharéis. Outra ação importante foi a inclusão da Libras nos currículos de formação de professores de todas as licenciaturas e no curso de Fonoaudiologia. É interessante notar que essa inclusão tem um impacto direto no reconhecimento e na valorização da Libras uma vez que abrange todos os futuros professores que irão atuar no ensino de todas as áreas da educação básica (QUADROS, 2017).

O alfabeto em Libras, ilustrado na Figura 2.1, auxilia na comunicação mais básica com um indivíduo surdo ou deficiente auditivo. No entanto, como dito anteriormente, a Libras na sua condição de língua natural é tão complexa e sofisticada quanto qualquer outra língua oral e, portanto, não é baseada apenas na utilização do alfabeto.

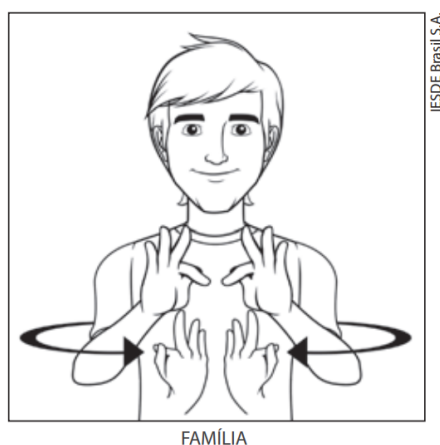
A composição da comunicação das línguas oral-auditivas utiliza os chamados itens lexicais os quais são equivalentes aos denominados *sinais* das línguas de sinais. Os sinais da Libras são formados a partir da combinação de diversos parâmetros, a citar: (1) configuração das mãos; (2) ponto de articulação; (3) movimento; (4) orientação e (5) expressão facial e/ou corporal, como pode ser visto no exemplo da Figura 2.2. Todos os componentes na formação de um sinal em Libras são importantes na compreensão do conteúdo do que está sendo transmitido. A alteração de um dos parâmetros na formação de um dado sinal resulta em um sinal diferente

Figura 2.1: Alfabeto em Libras. Fonte: (RODRIGUES; VALENTE, 2011).



ou até mesmo em um sinal inexistente, ou seja, pequenas alterações na formação de um sinal provocam diferentes significados (RODRIGUES; VALENTE, 2011).

Figura 2.2: Exemplo de sinal em Libras. Fonte: (RODRIGUES; VALENTE, 2011).



### 2.1.1 Expressões Faciais Gramaticais na Língua Brasileira de Sinais

No tocante às expressões faciais, em particular, as mesmas estão relacionadas à estruturas nos níveis da morfologia e sintaxe, sendo obrigatórias em determinados contextos (QUADROS; KARNOPP, 2004). No nível morfológico, as expressões faciais correspondem ao grau de inten-

sidade de um adjetivo ou ao grau de tamanho para um substantivo. Já a nível da sintaxe, as expressões faciais determinam o tipo da estrutura da sentença, a qual pode ser de nove tipos diferentes, conforme ilustrado na Tabela 2.1.

Tabela 2.1: Tipos de estrutura de sentenças quanto à sintaxe das expressões faciais e exemplos correspondentes. Elaborado a partir de: (QUADROS; KARNOPP, 2004; RAMOS, 2004; SOUSA, 2010)

| Tipo de Sentença                        | Exemplo  |
|---|--|
| Afirmativa                              | <i>Eu irei à escola.</i>                             |
| Negativa                                | <i>Não gosto de chocolate.</i>                       |
| Condicional                             | <i>Se chegarmos no horário iremos ao cinema.</i>     |
| Interrogativa com pronome interrogativo | <i>Quando é o seu aniversário?</i>                   |
| Interrogativa binária (sim/não)         | <i>Você vai ao shopping hoje?</i>                    |
| Interrogativa de dúvida                 | <i>Você tem certeza que esse lápis é seu?</i>        |
| Relativa                                | <i>O carro que quebrou está na oficina.</i>          |
| Tópico                                  | <i>Cores, eu gosto de vermelho.</i>                  |
| Foco                                    | <i>O almoço foi risoto. Não, o almoço foi arroz.</i> |

Para realizar estas expressões faciais devem ser utilizados o movimento da cabeça, a direção do olhar, a elevação das sobrancelhas, o franzir da testa e até mesmo o movimentos dos lábios, conforme ilustrado na Figura 2.3. Cada expressão facial possui características bem definidas e pode aparecer mais de uma vez em uma única sentença. É importante ressaltar que a omissão das expressões faciais pode retirar o sentido de uma determinada frase emitida na língua brasileira de sinais (QUADROS; KARNOPP, 2004).

Figura 2.3: Exemplos de Expressões Faciais Gramaticais. Fonte: (SOUSA, 2010).

(a) Negativa



(b) Afirmativa



(c) Interrogativa



## 2.2 Aprendizado de Máquina

As técnicas de Aprendizado de Máquina (AM) têm sido aplicadas com sucesso em um grande número de problemas reais em diversos domínios. A principal razão deste sucesso é decorrente da natureza inferencial e da boa capacidade de generalização dos métodos e técnicas desta área, cuja ideia central consiste em utilizar algoritmos capazes de aprender padrões por meio de exemplos, baseando-se apenas em dados previamente disponíveis (FACELI et al., 2011; GULLI; PAL, 2017).

Nas últimas décadas, com a crescente complexidade dos problemas a serem tratados computacionalmente e diante do grande volume de dados constantemente gerados por diferentes setores, tornou-se clara a necessidade de ferramentas, algoritmos, métodos e técnicas computacionais mais sofisticados para endereçar estas questões. Este cenário favoreceu a difusão das técnicas de AM que utilizam essa massa de dados para adquirir conhecimento (FACELI et al., 2011).

Os algoritmos de AM têm sido amplamente utilizados em diversas tarefas as quais podem ser organizadas de acordo com diferentes critérios. Um desses critérios diz respeito ao paradigma de aprendizado a ser adotado para lidar com a tarefa em questão. Os diferentes paradigmas de aprendizado variam de acordo a característica da tarefa que pode ter propriedades preditivas ou descritivas (MARSLAND, 2015).

Em tarefas de descrição, as técnicas de AM aprendem as informações contidas no conjunto de dados com a finalidade de descrever ou explorar estes dados. Nesse caso, a base de dados não possui atributos de saída e, portanto, seguem um paradigma de aprendizado chamado *aprendizado não supervisionado*. As técnicas utilizadas neste paradigma utilizam regras para, por exemplo, agrupar os dados ou encontrar associações que relacionam os atributos de entrada (BRINK; RICHARDS; FETHEROLF, 2017).

As tarefas preditivas utilizam modelos que submetem os dados de entrada à um treinamento que aprende a rotular uma determinada saída com base no conhecimento adquirido. Assim sendo, os conjuntos de dados utilizados neste tipo de tarefa possuem atributos de entrada e saída e, portanto, seguem um paradigma de aprendizado chamado *aprendizado supervisionado*.

Dentre as tarefas preditivas é possível ainda categorizá-las como tarefas de *classificação* ou tarefas de *regressão* (FACELI et al., 2011).

Os problemas de regressão utilizam os atributos de entrada para estimar um valor real de saída. Por outro lado, os problemas de classificação baseiam-se nos atributos de entrada para determinar qual a classe discreta uma dada entrada pertence. Quando os problemas de classificação possuem apenas duas classes, são chamados de problemas de classificação *binária* e quando possuem três ou mais classes, são chamados de problemas de classificação *multi-classe* (FACELI et al., 2011).

A Figura 2.4 apresenta um esquema com a hierarquia de aprendizado baseada nos tipos de tarefa especificados. Uma vez que a tarefa de AM é definida, o processo de aprendizado se inicia, o qual (MARSLAND, 2015, vide Seção 1.5) baseia-se em uma metodologia composta por seis etapas.

Figura 2.4: Hierarquia de aprendizado. Fonte: (FACELI et al., 2011).



A primeira etapa do processo de aprendizado consiste na preparação e estruturação do conjunto de dados. Em seguida, com os dados já consolidados, é feita a seleção das características mais relevantes ao problema em questão. Esta etapa demanda certo conhecimento do domínio do problema e eventualmente utiliza embasamento estatístico para auxiliar na análise do conjunto de dados. A terceira e quarta etapa consistem em especificar o algoritmo de AM, o modelo e os parâmetros a serem utilizados. Segue-se então para o treinamento do modelo no qual ocorre a realização da inferência. Por fim, tem-se a etapa de avaliação do modelo para examinar sua capacidade de generalização, além da geração de métricas (MARSLAND, 2015).

### 2.2.1 Árvores de Decisão

As Árvores são estruturas de dados muito utilizadas como modelos de AM. Esse tipo de estrutura emprega a estratégia de dividir para conquistar, na qual um problema complexo é dividido em problemas mais simples aos quais recursivamente é aplicada essa mesma estratégia (FACELI et al., 2011; MARSLAND, 2015).

No escopo de problemas de classificação utiliza-se, em particular, as Árvores de Decisão (AD). A ideia geral deste modelo consiste em dividir a classificação em um conjunto de escolhas sobre cada atributo, começando pela raiz da árvore e progredindo até as folhas por meio de estruturas denominadas nós (MARSLAND, 2015). Desse modo, a previsão de uma AD consiste em uma série de testes conduzida a partir do nó raiz em que o resultado é obtido quando um nó folha, o qual está associado a um rótulo, é alcançado (ZHOU, 2012). Os testes que coordenam as decisões são alocados nos chamados nós de divisão os quais são univariados, isto é, cada nó dispõe de um único teste que envolve um único atributo (FACELI et al., 2011).

Alguns conceitos baseados em técnicas matemáticas são aplicados para definir qual atributo utilizar nos testes de divisão para a construção de uma AD. Um destes conceitos é a *entropia*, a qual é usada para medir a aleatoriedade do atributo alvo, descrita pela Equação 2.1, em que  $A$  é uma variável aleatória discreta e a probabilidade de observar cada valor é  $p_1, p_2, \dots, p_v$ . A cada nó de divisão, o atributo que mais reduzir o valor de entropia será escolhido para dividir os dados. Para cada atributo, o *ganho de informação* mede a redução na entropia nas partições obtidas de acordo com os valores do atributo, isto é, a diferença entre a entropia do conjunto de exemplos e a soma ponderada da entropia das partições (FACELI et al., 2011).

$$E(A) = - \sum_{i=0} p_i \times \log_2 p_i, \quad (2.1)$$

em que  $E$  denota a entropia de Shannon. Pode-se também utilizar a entropia de Gini como métrica de entropia para cálculo do ganho de informação neste cenário.

Existem diversos algoritmos para indução de ADs e cada um deles é voltado para determinados tipos de problema. O algoritmo ID3, pioneiro nesta tarefa é de natureza gulosa e voltando



apenas para atributos categóricos (BOUZADA; RIBEIRO; PEIXE, 2013; ZHOU, 2012). Outro algoritmo bastante conhecido é o C4.5, inspirado no ID3 e que utiliza uma abordagem recursiva de particionamento do conjunto de dados (GOLDSHIMIDT; PASSOS, 2005). Além destes, o CART também é bastante utilizado, capaz de construir somente árvores do tipo binária (FACELI et al., 2011; MARSLAND, 2015).

### 2.2.2 Redes Neurais Artificiais

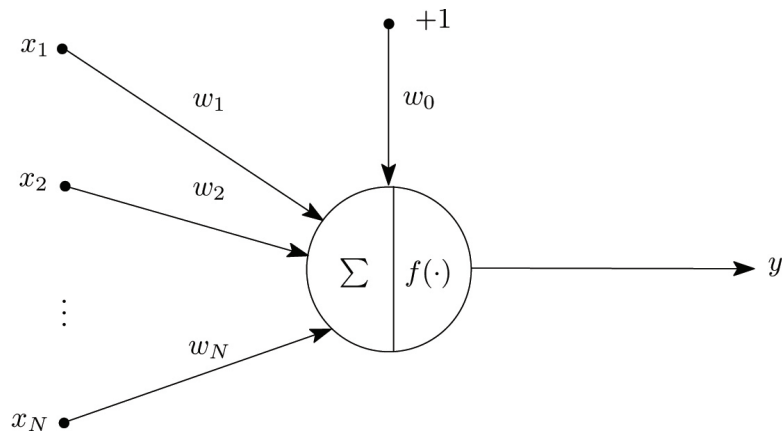
As Redes Neurais Artificiais (RNAs) são modelos computacionais inspirados na capacidade de processamento de informações do cérebro humano (ROJAS, 1996). De acordo com esta ideia, as RNAs possuem unidades de processamento simples, denominadas *neurônios artificiais*, dispostos em camadas interconectadas por ligações associadas a coeficientes numéricos, chamados *pesos* (FACELI et al., 2011). As RNAs são capazes de aprenderem padrões complexos a partir dos dados e prever resultados para exemplos não conhecidos, o que demonstra a sua capacidade de generalização (HAYKIN, 2009).

O neurônio artificial é a unidade fundamental na construção de RNAs, tendo sido inspirado no seu análogo biológico. Segundo Rosenblatt, existe um conjunto de  $m$  entradas, equivalentes aos dendritos de um neurônio biológico, por onde os sinais são introduzidos (ROSENBLAT, 1961). Associa-se um peso a cada entrada, representando a relevância referente a uma conexão sináptica. Há também o peso  $w_0$ , um termo de polarização criado com a intenção de estabelecer um limiar de ativação para cada neurônio. Este peso corresponde à entrada *bias*, cujo valor é sempre unitário. Pode-se então definir um vetor de entradas  $X = [+1, x_1, x_2, \dots, x_m]$  e um vetor de pesos  $W = [w_0, w_1, \dots, w_m]$ . As entradas e pesos são combinados por meio de uma função  $\phi : \mathbb{R}^{m+1} \rightarrow \mathbb{R}$ , que é geralmente a soma ponderada das entradas e pesos, conforme Equação 2.2. Este modelo de neurônio encontra-se ilustrado na Figura 2.5 (LIMA, 2016).

$$\phi(X, W) = \sum_{i=0}^m x_i \cdot w_i. \quad (2.2)$$

A função  $f$  é chamada de *função de ativação* e fornece a resposta de um neurônio para uma dada entrada. Esta função precisa ser monotônica e contínua, podendo comumente ser as

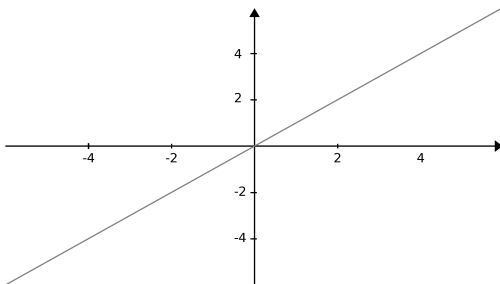
Figura 2.5: Neurônio artificial: a combinação linear das entradas  $x$  ponderadas pelos pesos  $w$  é transformada pela função de ativação  $f$  na saída  $y$ .



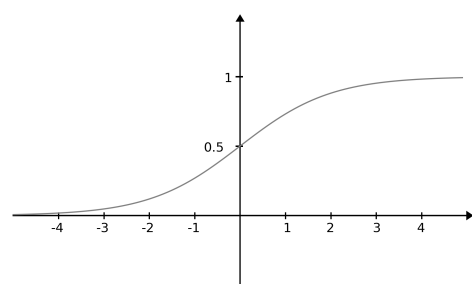
funções identidade, sigmoide, tangente hiperbólica, ou a retificada linear (ReLU) (DELICATO; PIRES; SILVEIRA, 2017). Estas funções encontram-se representadas na Figura 2.6.

Figura 2.6: Exemplos de diferentes funções de ativação.

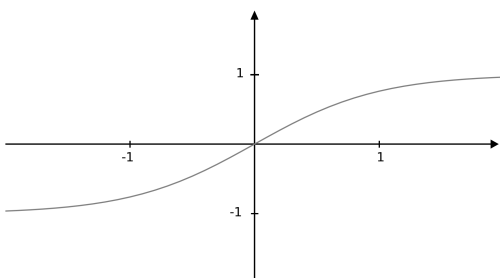
(a) Função identidade.



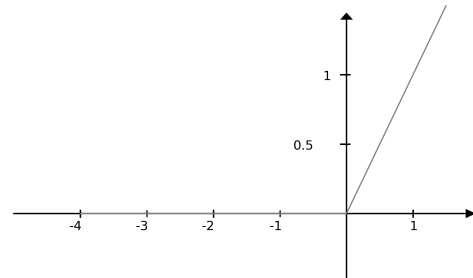
(b) Função sigmoide.



(c) Função tangente hiperbólica.



(d) Função retificada linear.



Neurônios artificiais individuais têm uma capacidade computacional limitada, independentemente da função de ativação escolhida, pois resolvem apenas problemas linearmente separáveis.

No entanto, um conjunto de neurônios artificiais conectados na forma de uma rede – *rede neural artificial* – adquirem a capacidade de resolver problemas de elevada complexidade (BRAGA; CARVALHO; LUDERMIR, 2007). A alternativa mais utilizada para resolver estes problemas é distribuir os neurônios em uma ou mais camadas conhecidas como camadas ocultas (FACELI et al., 2011). Segundo Cybenko, uma rede com uma camada oculta pode implementar qualquer função contínua e uma rede com duas camadas ocultas permite a aproximação de qualquer função (CYBENKO, 1989).

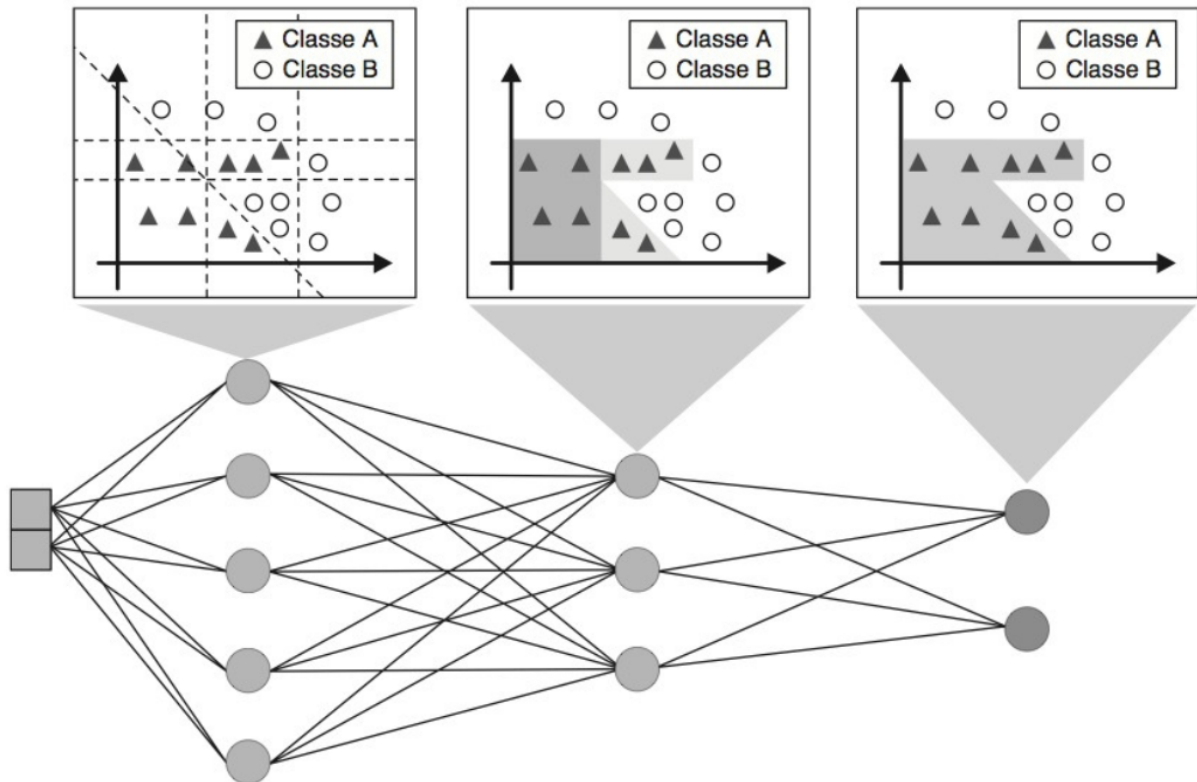
As RNAs do tipo Perceptron Multicamadas (MLP, do inglês *Multilayer Perceptron*) apresentam uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída. Uma das principais características de uma rede MLP é o seu alto grau de conectividade entre os neurônios, cuja intensidade está associada aos pesos da rede.

Cada neurônio em uma rede MLP atua ponderando as entradas recebidas dos neurônios de uma camada anterior a ele conectados, produzindo como saída um valor, resultante de sua função de ativação, que é propagado às camadas seguintes da rede neural. Conforme exemplificado na Figura 2.7, a combinação das atuações individuais desempenhadas por cada neurônio da rede que define a atuação associada à RNA como um todo (BRAGA; CARVALHO; LUDERMIR, 2007; FACELI et al., 2011; HAYKIN, 2009).

Uma importante característica das RNAs é a sua capacidade de aprender por meio de exemplos. O processo de *aprendizado* de uma rede neural consiste em sucessivos ajustes de pesos associados aos seus neurônios, de modo a aprimorar seu desempenho de acordo com um critério pré-estabelecido. Tais ajustes são realizados por algoritmos de treinamento formados por um conjunto de regras bem definidas que especificam quando e como deve ser alterado o valor de cada peso. Diversos algoritmos de aprendizado foram propostos, dentre os quais se destacam aqueles que seguem o paradigma de *aprendizado supervisionado* (FACELI et al., 2011; LIMA, 2016).

O aprendizado supervisionado ajusta os pesos aplicando um conjunto de exemplos de treinamento rotulados. Cada exemplo consiste em um sinal de entrada associado à sua resposta alvo desejada. A cada padrão de entrada submetido à rede, compara-se a resposta desejada

Figura 2.7: Papel desempenhado pelos neurônios das diferentes camadas de uma rede MLP. Fonte: (FACELI et al., 2011).



com a resposta calculada, ajustando-se os pesos das conexões para minimizar o erro (HAYKIN, 2009).

O algoritmo mais utilizado para o treinamento de redes MLP é o algoritmo *backpropagation*, também chamado de retropropagação do erro ou ainda regra delta generalizada. Este algoritmo respeita o aprendizado supervisionado em que os pesos são modificados e ajustados para reduzir a distância entre a resposta desejada e a resposta produzida pela rede (HAYKIN, 2009). O treinamento é constituído da iteração de duas fases, uma fase para frente (*forward*) e uma fase para trás (*backwards*) (FACELI et al., 2011). A fase *forward*, que compreende o fluxo da informação a partir da entrada até a saída da rede, é utilizada para produzir uma saída para um dado sinal de entrada. A fase *backwards*, com fluxo da informação da saída da rede em direção à entrada, utiliza a diferença entre as saídas desejada e produzida para atualizar os pesos das conexões entre os neurônios e assim minimizar o erro (BRAGA; CARVALHO; LUDERMIR, 2007). Os ciclos de apresentação dos dados de treinamento e eventuais ajustes de pesos no

*backpropagation* são iterados até que seja atingido um critério de parada como, por exemplo, um número máximo de ciclos ou uma taxa máxima de erro (FACELI et al., 2011).

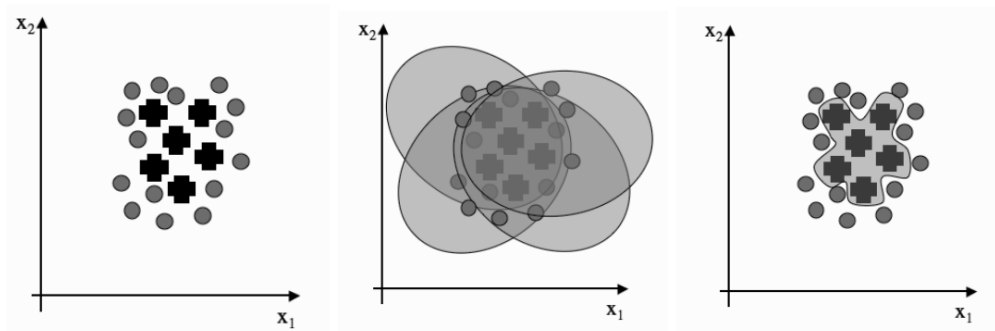
As RNAs são modelos computacionais com ampla aplicação na resolução de problemas de previsão. Algumas aplicações utilizam RNAs para predição de condições climáticas, como em (SOUSA et al., 2017) e (ARAUJO et al., 2017), em que se deseja prever precipitações de chuva de determinados locais. Diversas outras aplicações de RNAs dizem respeito à classificação de padrões. Dentre as aplicações na área de classificação financeira, uma das mais bem sucedidas é a análise de crédito (BRAGA; CARVALHO; LUDERMIR, 2007). Além disso, as RNAs também são muito utilizadas para diagnósticos médicos, como em (SILVA et al., 2016) e (PEREIRA et al., 2016). Outras aplicações tratam de reconhecimento de caracteres (CARVALHO, 2006), robótica (TINOS, 1999), jogos (CAEXETA, 2008), comunicação (SILVA et al., 2017), dentre outros.

### 2.2.3 Métodos *Ensemble*

Os métodos *Ensemble* são uma estratégia de AM na qual as previsões independentes de vários modelos são combinadas. A ideia deste método é considerar que múltiplos modelos combinados podem produzir melhor desempenho preditivo do que cada um desses mesmos modelos individualmente (BRINK; RICHARDS; FETHEROLF, 2017). A Figura 2.8 mostra a idéia básica do método de aprendizado *Ensemble*. Considerando um determinado problema de classificação binária relativamente simples e algum modelo de aprendizado que descreve uma elipse ao redor de um subconjunto dos dados, a combinação das elipses pode fornecer um limite de decisão consideravelmente mais complexo (MARSLAND, 2015).

A Floresta Aleatória (FA) é um método *Ensemble* que utiliza a combinação de ADs para obter um melhor desempenho. As ADs que crescem muito profundamente tendem a aprender padrões altamente irregulares. Assim sendo, as FAs procuram, por meio de voto majoritário, atenuar esse problema causado pela utilização de uma única AD, calculando a média de várias ADs profundas, treinadas em diferentes partes do mesmo conjunto de treinamento, com o objetivo de reduzir a variação. Para tanto, as FAs utilizam algoritmos como *Bagging* e *Boosting*

Figura 2.8: Método *Ensemble*: fornecer um limite de decisão mais complexo. Fonte: (MARSLAND, 2015).



(MARSLAND, 2015; BRINK; RICHARDS; FETHEROLF, 2017).

A técnica *Bagging* tem o propósito de combinar preditores gerados por um mesmo algoritmo base com o objetivo de reduzir a variância de funções preditivas. Esse modelo apresenta bons resultados em algoritmos base instáveis, principalmente nas ADs. A principal propriedade desta técnica é gerar conjuntos sucessivos e independentes de amostras, denominadas *bootstraps*, para serem treinados gerando modelos diferentes. O resultado deste treinamento resulta em uma distribuição de classes para cada subconjunto que não é exatamente a mesma distribuição da classe original, podendo gerar amostras mais representativas das classes que possuem menos exemplos. Desta forma, cada novo modelo pode ser adequado para capturar uma determinada característica do problema e aumentar seu desempenho (MARSLAND, 2015; BRINK; RICHARDS; FETHEROLF, 2017).

O algoritmo *Boosting* é um método *Ensemble* que reduz o erro de generalização ponderando o conjunto de dados para atribuir pesos maiores aos exemplos que foram classificados incorretamente (BRINK; RICHARDS; FETHEROLF, 2017). O *AdaBoost*, em particular, é baseado em *Boosting* e utiliza múltiplas árvores combinadas, conhecidas como “*weak learners*”, para constituir um modelo. Nesse algoritmo, em cada iteração um novo classificador é treinado com os pesos aplicados em cada subconjunto que será modificado nessa iteração de acordo com a performance deste mesmo subconjunto na iteração anterior (MARSLAND, 2015).

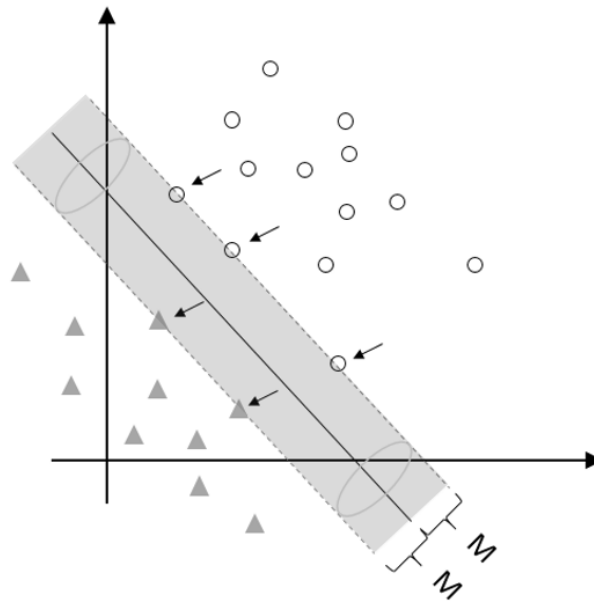
### 2.2.4 Máquinas Vetores de Suporte

As Máquinas Vetores de Suporte (SVM, do inglês *Support Vector Machines*) foram introduzidas em 1992 por Vapnik e se tornaram um dos algoritmos de AM mais populares, principalmente por produzirem resultados comparáveis e muitas vezes superiores aos obtidos por outros algoritmos tal como as RNAs (FACELI et al., 2011; MARSLAND, 2015).

A ideia principal das SVMs é encontrar a reta, ou o hiperplano equivalente em dimensões mais altas, que separa as classes de forma otimizada. Em vez de medir a distância entre todos os pontos, as SVMs tentam encontrar a maior margem somente entre os pontos mais próximos da reta de separação das classes, levando em consideração a irrelevância dos pontos mais distantes (BRINK; RICHARDS; FETHEROLF, 2017). Para tanto, o algoritmo das SVMs baseia-se na Teoria de Aprendizado Estatístico que estabelece condições matemáticas que auxiliam na escolha de um classificador a partir de um conjunto de treinamento, levando em conta o desempenho e a complexidade do classificador, com a finalidade de obter um bom desempenho também para novos dados do mesmo domínio (FACELI et al., 2011).

Um exemplo de separação entre duas classes produzido por uma SVM é apresentado na Figura 2.9 com os dados representados como pontos no espaço, mapeados de forma que se estabeleça uma fronteira de decisão entre as classes do conjunto de dados. Para entender o funcionamento deste modelo, considera-se uma região em torno da fronteira de decisão, conforme ilustrado de maneira sombreada na Figura 2.9, em que qualquer ponto que se encontra dentro desta região é considerado como sendo não pertencente à nenhuma das duas classes. Essa região é simétrica em relação à fronteira, produzindo um cilindro de três dimensões ou um hiper-cilindro em dimensões maiores. O maior raio que se pode obter permitindo que os pontos com classes definidas permaneçam fora ou tangente ao cilindro é chamado de *margem*, a qual pode ser observada na Figura 2.9 com valor igual a  $M$ . Os pontos mais próximos da linha de decisão são chamados de vetores de suporte e são considerados os mais importantes, pois após o treinamento as SVMs utilizam apenas esses pontos para determinar a classificação (MARSLAND, 2015). Assim, a classificação de um dado exemplo por uma SVM está sujeita aos vetores de suporte encontrados.

Figura 2.9: Exemplo de SVM.



O exemplo descrito anteriormente retrata uma SVM linear, adequada para problemas de classificação linearmente separáveis. Porém, considerando problemas reais, na maioria dos casos não é possível dividir satisfatoriamente os dados de treinamento por uma reta ou hiperplano. Assim sendo, para problemas não lineares, as SVMs mapeiam o conjunto de treinamento do espaço original para um novo espaço de maior dimensão, denominado *espaço de características*. A escolha apropriada deste mapeamento faz com que o conjunto de treinamento possa ser então separado por uma SVM linear. Este procedimento é baseado no Teorema de Cover, o qual relata que, dado um conjunto de dados não linear em um determinado espaço de entradas, este espaço pode ser transformado em um espaço de características no qual, com alta probabilidade, os objetos são linearmente separáveis. Para tanto, duas condições precisam ser satisfeitas: a transformação precisa ser não linear e a dimensão do espaço de características deve ser suficientemente alta (FACELI et al., 2011).

### 2.2.5 Vizinhos mais Próximos

O algoritmo Vizinhos mais Próximos, do inglês *Nearest Neighbours* (NN), é um método de AM baseado em distância o qual aplica a seguinte hipótese base: dados similares tendem a estar



concentrados em uma mesma região no espaço de entrada (FACELI et al., 2011). As instâncias do algoritmo NN são definidas pelo número de vizinhos que serão considerados, em que para  $k$  vizinhos tem-se o algoritmo  $k$ -NN. Nesta instância, a classificação de um determinado objeto baseia-se na classificação majoritária dos  $k$  exemplos mais próximos ao mesmo, em que  $k$  é um número ímpar a fim de diminuir os possíveis empates (MARSLAND, 2015).

Para determinar a proximidade entre os objetos do conjunto de treinamento, cada objeto é representado por um ponto no espaço de entrada. Uma vez representados, a distância entre todos os pares de pontos é dada comumente pela distância euclidiana, denotada por  $d$ , representada na Equação 2.3, em que  $x_i$  e  $x_j$  são dois objetos representados por vetores no espaço  $\mathbb{R}^d$ , e  $x_i^l$  e  $x_j^l$  são elementos desses vetores, que correspondem aos valores da coordenada  $l$  (FACELI et al., 2011).

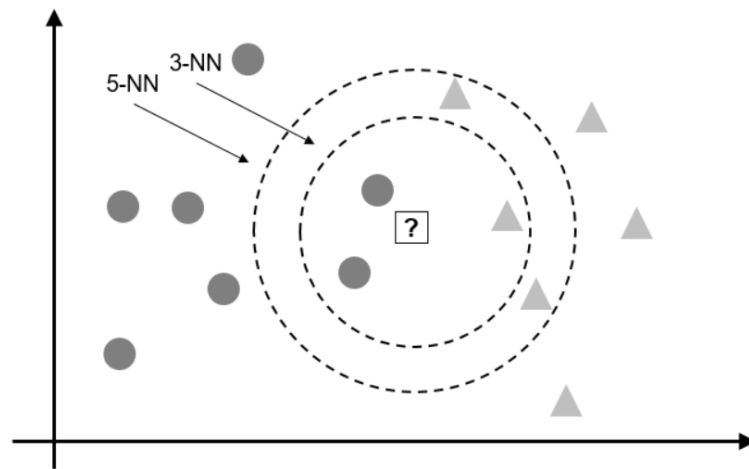
$$d(x_i, x_j) = \sqrt{\sum_{l=1}^d (x_i^l - x_j^l)^2}. \quad (2.3)$$

Após o cálculo das distâncias entre os pontos do espaço de entrada, é possível então apontar os  $k$  vizinhos mais próximos a um certo objeto. A escolha do valor de  $k$  pode não ser trivial, como é possível observar na Figura 2.10 em que são estabelecidas diferentes classificações para diferentes valores de  $k$ . Se o valor de  $k$  for muito pequeno, a classificação fica sensível a ruídos. Entretanto, se o valor de  $k$  for muito grande, a precisão diminui, pois pontos muito distantes serão considerados (MARSLAND, 2015). Frequentemente, o valor de  $k$  é pequeno e ímpar. A literatura estipula outras estratégias, como estimar  $k$  utilizando uma técnica chamada validação cruzada ou associar um peso à contribuição de cada vizinho (FACELI et al., 2011).

## 2.3 Tecnologias Utilizadas

As tecnologias e bibliotecas predominantemente utilizadas neste trabalho envolvem e são compatíveis com a linguagem de programação Python, pois esta tem se destacado amplamente em projetos de AM em diversos cenários. Esta é uma linguagem de programação interpretada, interativa, multi-paradigma, de alto nível, multi-plataforma, com uma sintaxe simples e código

Figura 2.10: Impacto do valor de  $k$  no algoritmo  $k$ -NN. Fonte: (FACELI et al., 2011).



aberto, idealizada por Guido van Rossum no início da década de 1990 (BORGES, 2010).

Para análise do conjunto de dados, as bibliotecas **pandas** e **numpy** tiveram um papel protagonista principalmente pelo desempenho e capacidade de simplificar tarefas complexas de manipulação de dados (NUMPY, 2018; PANDAS, 2018). Ademais, no que diz respeito ao treinamento e testes dos modelos de AM, a biblioteca **scikit-learn** foi considerada, tendo um papel importante para a execução de treino e testes de todos os modelos de AM propostos e nos cálculos automáticos das métricas de desempenho (SCIKIT-LEARN, 2018).

Por fim, a infra-estrutura de computação em nuvem provida pelo Google Cloud Platform (GCP) totalmente voltada para projetos de AM foi essencial para o desenvolvimento deste projeto. As máquinas virtuais oferecidas pela plataforma aumentaram o poder computacional acessível, possibilitando um processamento mais rápido para o cenário em questão. A manipulação e pré-processamento do conjunto de dados, o treinamento dos modelos de AM e a fase de análise desses modelos foram executados em instâncias disponíveis pelo GCP tendo em vista os recursos de memória e processamento necessários (GCLOUD, 2018).

## 2.4 Trabalhos Relacionados

Considerando o problema do reconhecimento automático de Libras com vistas a promover a inclusão social de pessoas surdas ou com deficiência auditiva, diversos trabalhos

na literatura foram desenvolvidos nesta perspectiva, inclusive utilizando diferentes métodos e técnicas e com resultados relevantes para o problema endereçado. Os conceitos de Aprendizado de Máquina, em geral, estão bastante presentes em muitos destes trabalhos para promover o reconhecimento de padrões e classificação das expressões faciais. Nesta seção serão apresentados e discutidos alguns destes trabalhos.

O trabalho Almeida et al. considerou a utilização de Visão Computacional e focou na detecção de gestos emitidos com as mãos em Libras (ALMEIDA et al., 2018). Para tanto, os autores utilizaram uma base dados de imagens estáticas e fizeram uso conjunto da biblioteca OpenCV e da linguagem de programação Python para capturar os contornos da mão. Os resultados obtidos neste experimento foram considerados satisfatórios, entretanto, os autores evidenciaram que a existência de ruído interferiu no processo, afetando a performance geral.

Alguns trabalhos utilizaram o sensor de movimento *Kinect* para obter informações dos gestos e expressões da Libras, como foi o caso de Barros Junior (JUNIOR, 2016) e de Briet et al. (BRIET; POVOA; SANTOS, 2016). Ambos os trabalhos empregaram a linguagem Python e os métodos de AM com os modelos de Redes Neurais Convolucionais para tarefas de classificação. A utilização do *Kinect* se dá pela possibilidade não só da captura de imagens, mas também pela capacidade de capturar a profundidade da imagem através do sensor infravermelho. No caso do primeiro autor, foram capturadas imagens por meio do sensor, as quais foram pré-processadas e então convertidas para a escrita de sinais SignWriting utilizando os modelos de redes neurais (JUNIOR, 2016). Briet et al., de maneira análoga, capturaram as imagens por meio do sensor para primeiramente consolidar uma base de dados. Os autores então aplicaram as técnicas de *Deep Learning* para classificar as expressões faciais existentes. No entanto, fatores como a arquitetura das redes e a baixa qualidade do conjunto de imagens corroboraram para resultados não expressivos com uma acurácia de aproximadamente 20%. (BRIET; POVOA; SANTOS, 2016).

Diferentemente da abordagem destes dois últimos trabalhos, Lazzarotto considerou a utilização de uma luva sensora pelo sujeito que emitia os sinais da Libras, fazendo com que a emissão de alguns gestos fosse utilizada para consolidar sua base de dados (LAZZAROTTO,

2016). Os gestos emitidos consistiam essencialmente nas letras do alfabeto em Libras, os quais foram capturados e processados por uma placa de sistemas microcontroladores em que é implementado um algoritmo de reconhecimento de padrões baseado em Redes Neurais Artificiais. As letras do alfabeto de Libras reconhecidas pelo modelo são apresentadas ao final do processo em um *display*. Os resultados obtidos por Lazzarotto com o reconhecimento dos padrões mostraram que o sistema, dentro do escopo de modelagem, apresentou viabilidade para ser utilizado em aplicações que possam desenvolver as habilidades em datilologia de palavras para alunos iniciantes em Libras.

Ainda no âmbito da emissão de sinais com as mãos, Porfírio constituiu sua base de dados por meio da captura de vídeos das configurações da mão, seleções das visões frontal e lateral, eliminação de ruídos e geração de uma malha 3D com o método da silhueta (PORFÍRIO, 2013). O autor também utilizou o sensor *Kinect* para este procedimento de processamento dos dados. Para classificar as configurações de mãos a partir das malhas 3D, foi utilizada a biblioteca LibSVM. O processo de treinamento utilizou a técnica *k-fold* com  $k = 10$  e a métrica para análise utilizada foi a acurácia, em que o melhor método obteve uma taxa 98.57% de acerto.

No tocante às expressões faciais em si, os trabalhos de Rezende e equipe e o de Freitas et al. endereçam problemas análogos ao que está sendo considerando no escopo deste trabalho (FREITAS; BARBOSA; PERES, 2014b; REZENDE et al., 2017; REZENDE, 2016; GUERRA et al., 2018). Rezende e equipe consideram essencialmente a utilização de técnicas de AM para reconhecimento de expressões faciais, com ênfase inicial nas Máquinas de Vetores de Suporte e, com posterior utilização de algoritmos de Vizinhos mais Próximos e Florestas Aleatórias (REZENDE et al., 2017; REZENDE, 2016; GUERRA et al., 2018). Porém, a base de dados utilizada por estes autores está em processo de formulação para posterior disponibilização pública e considera a emissão de frases específicas, o que restringe uma comparação equânime com outros trabalhos. Apesar disso, os resultados mostram-se animadores, com acurácia igual a 96,67% alcançada pelo modelo Vizinhos mais Próximos considerando a utilização dos dados de classificação deslocados.

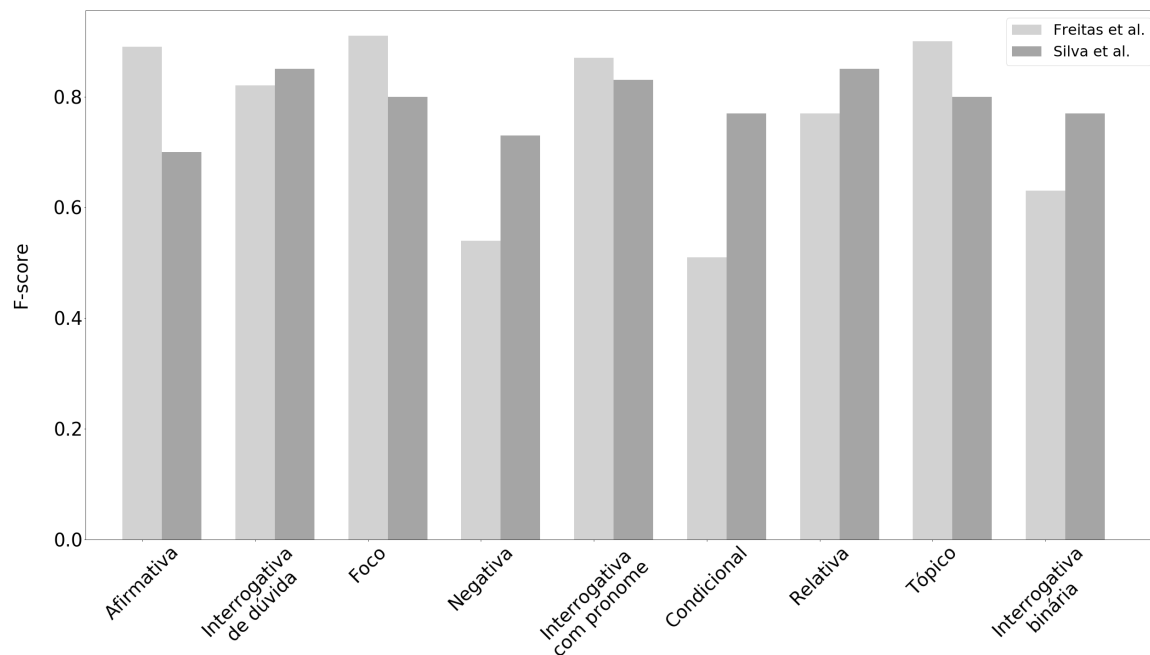
Freitas et al., por sua vez, inicialmente conceberam um conjunto de dados, intitulado *Gram-*

*matical Facial Expressions Data Set*, contendo 100 coordenadas de diferentes posições faciais obtidas a partir de 225 vídeos gravados com um sensor de movimentos enquanto um sujeito emitia diferentes sentenças em Libras (FREITAS; BARBOSA; PERES, 2014a). Utilizando apenas 17 coordenadas das diferentes posições faciais no *dataset* concebido, os autores treinaram e testaram uma única rede neural artificial para o problema de classificação da expressão facial gramatical correspondente. A rede proposta pelos autores, do tipo *multilayer perceptron* com 10 neurônios na camada oculta, foi treinada com diferentes taxas de aprendizado. De acordo com os resultados obtidos, a melhor expressão facial detectada foi a de foco, ao passo que as expressões negativas tiveram *F-score* de 45%, indicando uma baixa sensibilidade do modelo proposto, o qual classifica menos expressões negativas do que deveria (FREITAS; BARBOSA; PERES, 2014b).

Dadas as características particulares das expressões negativas e a complexidade envolvida na emissão das mesmas, o reconhecimento automático deste tipo de expressão facial tem se mostrado um desafio. Além do que foi exposto, é essencial considerar também as variações nas expressões faciais emitidas por diferentes pessoas e ainda a co-ocorrência destas expressões com outros elementos da língua brasileira de sinais, que podem resultar em oclusão da face. Estes dois aspectos acentuam as dificuldades na concepção de métodos automáticos de reconhecimento (FREITAS; BARBOSA; PERES, 2014b). Desse modo, um desenvolvimento anterior da autora deste trabalho juntamente com colaboradores sujeitou à esta base de dados o treinamento de mais redes neurais, considerando uma variação mais ampla de parâmetros e hiperparâmetros a fim de obter melhores resultados (SILVA et al., 2017). Ao final do treinamento as métricas de *F-score* acusaram uma melhoria de mais de 20% na expressão facial gramatical do tipo negativa em comparação com (FREITAS; BARBOSA; PERES, 2014b). A Figura 2.11 apresenta um histograma dos *F-scores* obtidos por (FREITAS; BARBOSA; PERES, 2014b) e (SILVA et al., 2017) para expressão facial gramatical.

É interessante ressaltar que a contribuição de Freitas et al. (FREITAS; BARBOSA; PERES, 2014b) e o desenvolvimento anterior (SILVA et al., 2017) foram de suma importância para uma análise mais profunda deste problema, a qual está sendo conduzida neste trabalho e

Figura 2.11: Histograma com os  $F$ -scores obtidos por (FREITAS; BARBOSA; PERES, 2014b) e (SILVA et al., 2017) para cada expressão facial gramatical.



que considera o mesmo conjunto de dados, mesma tarefa de classificação de expressões faciais gramaticais, mas, em contraponto à limitação de ambos os trabalhos em só utilizar redes neurais, considera diversos modelos de AM para obtenção dos resultados, bem como a realização de uma busca em *grid* e a utilização de técnicas de validação mais robustas.

# Capítulo 3

## Solução Proposta

### 3.1 Tarefa de Aprendizado de Máquina

No contexto deste trabalho, o problema de classificação de expressões faciais gramaticais da Libras foi abordado como uma *tarefa de classificação multiclasse*, cujo objetivo era classificar qual a expressão facial gramatical correspondente dentre um conjunto de 9 categorias discretas, a citar: (1) Afirmativa, (2) Negativa, (3) Condicional, (4) Interrogativa com Pronome Interrogativo, (5) Interrogativa Binária (sim/não), (6) Interrogativa de Dúvida, (7) Relativa, (8) Tópico e (9) Foco. Para tanto, primeiramente, houve uma etapa de pré-processamento, em que os exemplos organizados segundo as diferentes classes, foram agrupados em um conjunto de dados único com os rótulos correspondentes. Com isso, estabeleceu-se o atributo-alvo do problema.

A etapa seguinte consistiu na especificação dos diversos modelos de Aprendizado de Máquina variando os valores de seus parâmetros e hiperparâmetros. Foram propostos modelos de Árvores de Decisão, Florestas Aleatórias, Métodos *Ensemble*, Máquinas Vetores de Suporte, Redes Neurais e Vizinhos mais Próximos.

Para que os modelos propostos para classificação produzissem resultados factíveis, fez-se necessária uma etapa de treinamento, em que o dados foram fornecidos sucessivamente aos modelos propostos seguindo a metodologia de validação cruzada *k-fold* com  $k = 3$ , com vistas a prevenir *overfitting* e a capturar a qualidade da generalização proposta pelos modelos

considerados (BRINK; RICHARDS; FETHEROLF, 2017). O detalhamento acerca dos dados disponíveis para treino e teste dos modelos será apresentado na seção a seguir.

Nesta tarefa de classificação multi-classe, realizada de acordo com o paradigma supervisionado, a métrica de desempenho utilizada foi o *F-Score* do tipo *micro-averaging*, em virtude das diferentes classes serem desbalanceadas. A avaliação inicia-se com os resultados da matriz de confusão para cada classe individual, sendo *tp* os verdadeiros positivos, *fp* os falsos positivos e *fn* os falsos negativos. Em seguida, é preciso calcular a *Precisão* e a *Revocação*, conforme mostram as Eqs. 3.1 e 3.2,

$$\text{Precisão} = \frac{\sum_{i=1}^L tp_i}{\sum_{i=1}^L tp_i + fp_i}, \quad (3.1)$$

$$\text{Revocação} = \frac{\sum_{i=1}^L tp_i}{\sum_{i=1}^L tp_i + fn_i}, \quad (3.2)$$

em que *i* denota uma classe dentre *L* classes existentes. Parte-se então para o cálculo do *F-Score*, da seguinte forma:

$$\text{F-Score} = \frac{2 \times \text{Precisão} \times \text{Revocação}}{\text{Precisão} + \text{Revocação}}. \quad (3.3)$$

Esta métrica foi escolhida por considerar a precisão e a revocação bem como computar adequadamente as diferentes quantidades de exemplos por classe disponíveis (KUBAT, 2015, Capítulo 11).

## 3.2 Visão Geral do Conjunto de Dados

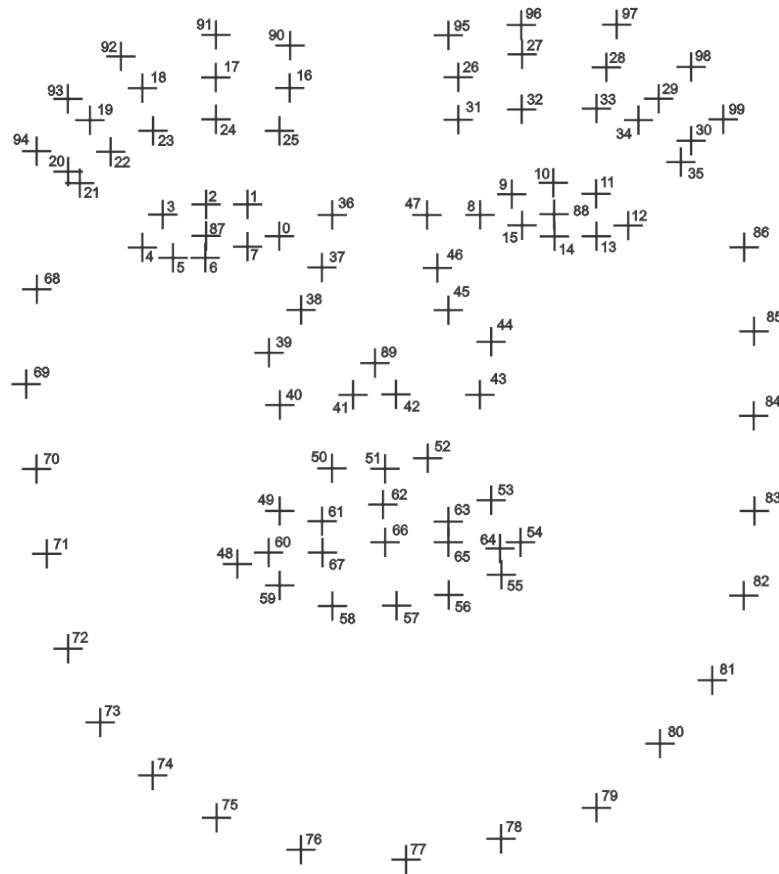
O conjunto de dados adotado para o desenvolvimento deste trabalho de conclusão de curso foi o *Grammatical Facial Expressions Data Set*, disponibilizado pela Universidade da Califórnia em Irvine (FREITAS; BARBOSA; PERES, 2014a). Este *dataset* é composto por exemplos de expressões faciais gramaticais anotadas de maneira supervisionada a partir de 225 vídeos nos quais dois indivíduos emitem sentenças em Libras. O *dataset* é disponibilizado gratuitamente



como uma iniciativa dos autores para estimular novos trabalhos que enderecem a detecção de expressões faciais gramaticais e favoreçam o desenvolvimento de métodos automatizados para o reconhecimento de Libras.

Para prover experiência acerca de uma dada expressão facial gramatical, os autores fizeram uma amostragem em *frames* nos quais foram marcadas 100 coordenadas  $(x, y, z)$  indicando altura, largura e profundidade de diferentes pontos de referência na face do indivíduo, como ilustra a Figura 3.1. O atributo alvo denota se a respectiva expressão gramatical está ou não sendo emitida no respectivo *frame*. Em resumo, cada exemplo do *dataset* correspondente a um tipo de expressão é composto de 300 atributos preditores, *timestamp* e um atributo alvo.

Figura 3.1: Pontos da face marcados para construção do conjunto de dados. Fonte: (FREITAS; BARBOSA; PERES, 2014a).



No escopo deste trabalho, decidiu-se utilizar apenas 17 pontos na face, os mesmos adotados no trabalho de Freitas et al. (FREITAS; BARBOSA; PERES, 2014b), localizados nas seguintes regiões da face: olhos esquerdo e direito (2 pontos em cada), sobrancelhas esquerda e direita,

íris esquerda e direita, linhas acima da sobrancelha esquerda e direita, nariz, boca (2 pontos), ponta do nariz e contorno da face. A decisão por utilizar apenas estes 17 pontos foi levada em consideração por permitir uma comparação em termos equivalentes com os resultados já existente na literatura, por diminuir o conjunto de atributos preditores, que impacta diretamente no tempo de treinamento e teste dos modelos, e para minimizar redundância dos atributos preditores, pois os pontos originais possuem distância de ordem milimétrica.

Um pré-processamento realizado teve como objetivo eliminar os demais pontos, ignorar os dados pessoais do sujeito que emitia os sinais (que poderia ser de dois tipos distintos, Sujeito *A* ou Sujeito *B*), descartar os dados do *timestamp* por não influenciarem na tarefa de classificação e, por fim, agrupar os dados em um conjunto de dados único indicando a expressão facial gramatical correspondente, o atributo alvo. Este pré-processamento culminou na estruturação do *dataset* com as bibliotecas da linguagem Python, promovendo praticidade na manipulação das informações.

Em seguida, uma análise preliminar do conjunto de dados permitiu identificar que as coordenadas  $z$ , referentes à profundidade, possuíam uma variação muito pequena entre si, além de haver *outliers* com valor zero indicando possíveis erros de medição, conforme ilustrado na Figura 3.2. Esta observação é particularmente reforçada ao considerar a análise das coordenadas  $z$  em função de certas expressões faciais gramaticais, especialmente a negativa, conforme ilustrado na Figura 3.3. Em decorrência deste fato, optou-se então por excluir tais valores do conjunto de dados, pois eventualmente poderiam atuar como fatores de confusão. Um outro fator que reforça essa exclusão é a possibilidade de trabalhos futuros extraírem características faciais em imagens e vídeos de apenas duas dimensões, nos quais a coordenada  $z$  naturalmente não existiria.

Considerando então os atributos preditores remanescentes, foi analisada a estatística descritiva dos mesmos, detalhada na Tabela 3.1. É interessante notar que os atributos preditores possuem distribuições oblíquas, o que pode ser evidenciado pela não coincidência entre média e mediana, e que o desvio padrão é significativo, indicando dispersão.

Ao final do pré-processamento, o conjunto dados passou a ter um total de 9.877 exemplos,

Figura 3.2: *Boxplot* da distribuição das coordenadas  $z$  de profundidade referentes aos 17 pontos escolhidos.

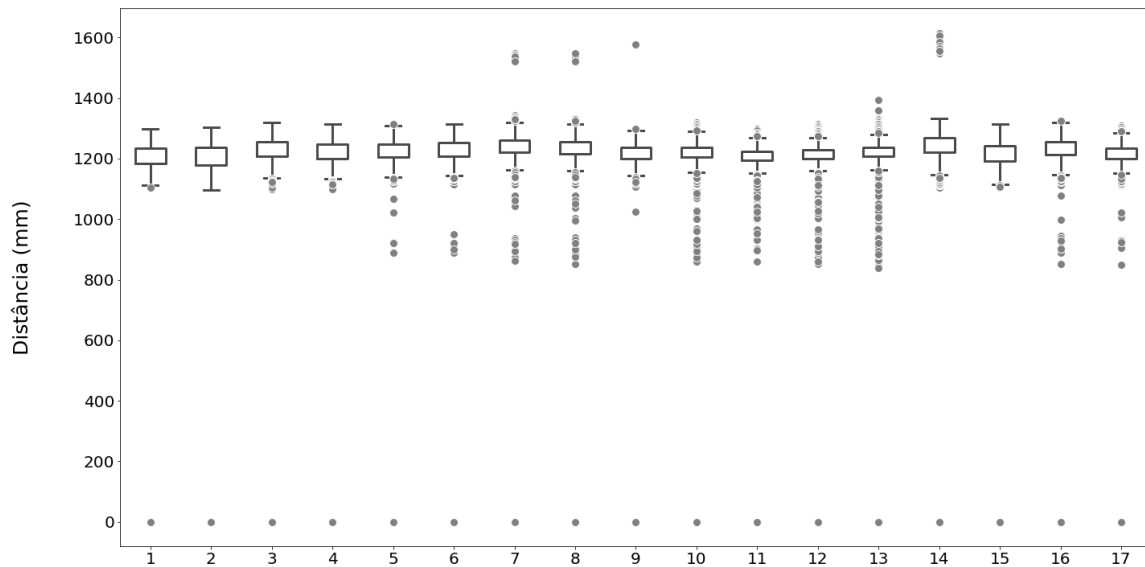
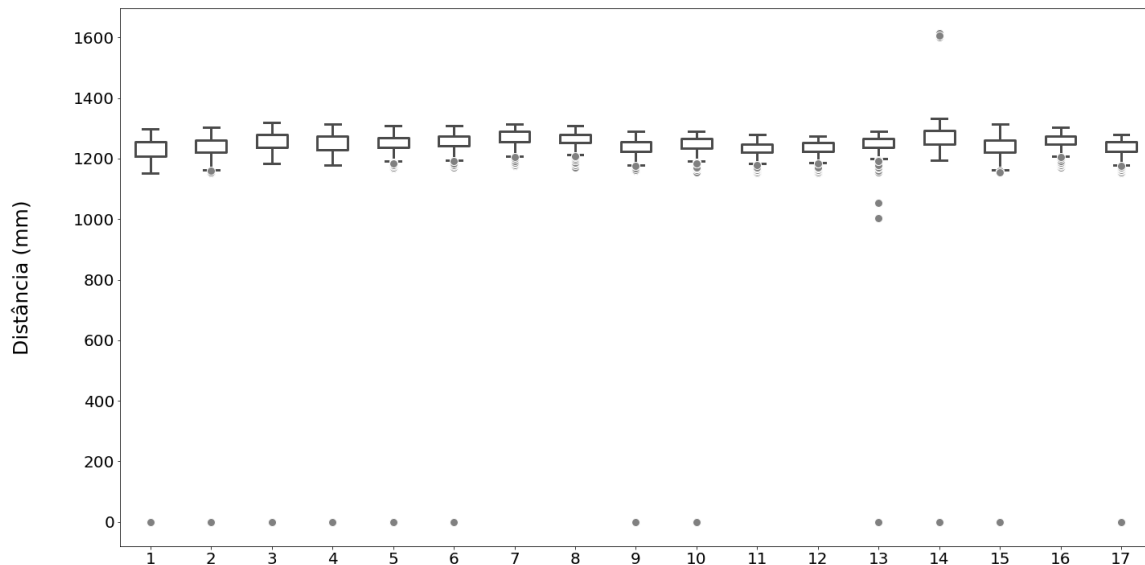


Figura 3.3: *Boxplot* da distribuição das coordenadas  $z$  de profundidade referentes aos 17 pontos escolhidos e considerado apenas as expressões faciais gramaticais negativas.



com 34 atributos preditores (17 pontos com 2 coordenadas cada) e um atributo alvo correspondente à uma das 9 expressões faciais gramaticais.

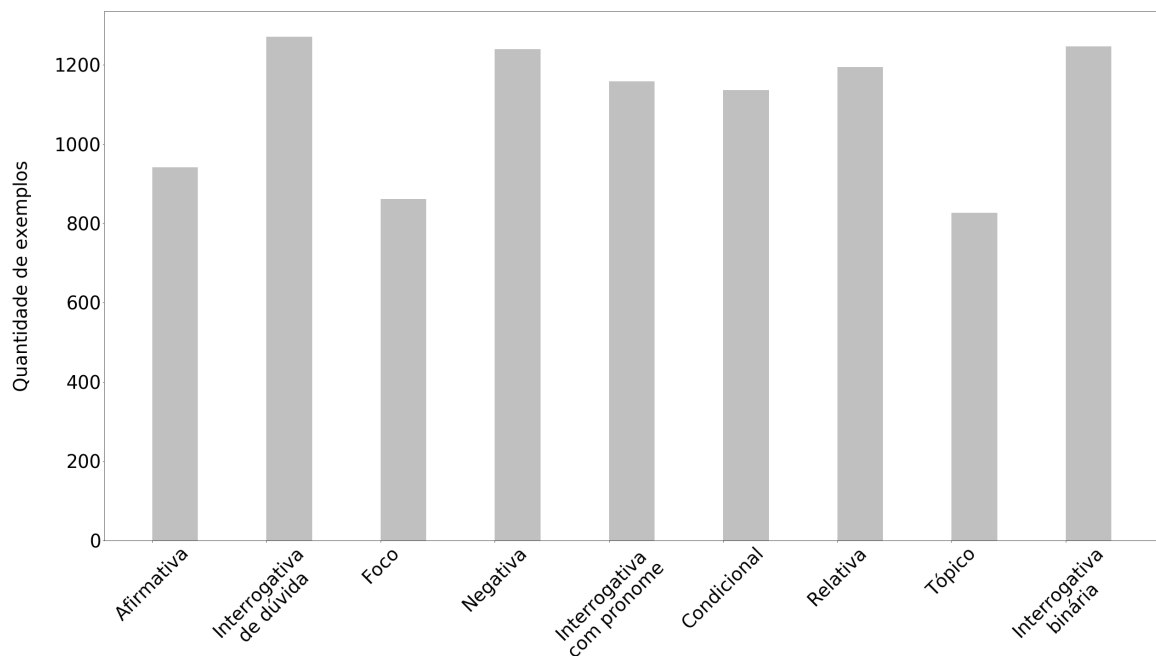
O gráfico da Figura 3.4 mostra a quantidade de exemplos por classe, em que é possível observar uma amostragem desbalanceada, a qual precisa ser levada em consideração para posterior realização de uma avaliação apropriada. Concluída a caracterização e análise do conjunto de dados, partiu-se então para especificação dos modelos de AM, seus parâmetros e hiperparâmetros,

Tabela 3.1: Estatística descritiva dos 17 pontos seleccionados. Os símbolos  $\bar{x}$ ,  $\tilde{x}$  e  $\sigma_x$  denotam a média, mediana e desvio padrão da coordenada  $x$ , respectivamente. A notação é análoga para a coordenada  $y$ .

| Posição na Face                     | $\bar{x}$ | $\tilde{x}$ | $\sigma_x$ | $\bar{y}$ | $\tilde{y}$ | $\sigma_y$ |
|-------------------------------------|-----------|-------------|------------|-----------|-------------|------------|
| Olho esquerdo                       | 303,44    | 304,93      | 11,33      | 218,93    | 229,06      | 15,13      |
| Olho direito                        | 332,18    | 333,15      | 11,23      | 218,49    | 227,43      | 13,83      |
| Sobrancelha esquerda                | 300,66    | 301,33      | 12,53      | 211,23    | 221,23      | 15,11      |
| Sobrancelha direita                 | 334,37    | 333,63      | 11,62      | 210,90    | 218,82      | 13,59      |
| Nariz                               | 317,62    | 317,82      | 12,76      | 228,28    | 232,96      | 17,57      |
| Boca                                | 318,39    | 319,05      | 14,18      | 245,03    | 255,32      | 18,67      |
| Contorno da face                    | 319,83    | 319,82      | 27,16      | 245,97    | 243,08      | 22,02      |
| Íris esquerda                       | 303,44    | 306,49      | 10,69      | 218,93    | 229,05      | 15,07      |
| Íris direita                        | 332,18    | 334,14      | 10,56      | 218,49    | 228,07      | 13,76      |
| Ponta do nariz                      | 317,64    | 319,62      | 11,49      | 231,87    | 241,99      | 17,23      |
| Linha acima da sobrancelha esquerda | 229,75    | 300,41      | 13,46      | 207,75    | 217,39      | 14,79      |
| Linha acima da sobrancelha direita  | 335,13    | 335,04      | 12,08      | 207,07    | 214,91      | 13,17      |

cuja metodologia de obtenção encontra-se descrita na seção a seguir.

Figura 3.4: Histograma com a quantidade de exemplos por classe.



### 3.3 Modelos, Parâmetros e Hiperparâmetros

Considerando a necessidade de otimizar a escolha dos parâmetros e hiperparâmetros dos modelos de AM, optou-se por realizar uma *busca em grid* dos possíveis valores que estes podem assumir. Nesta estratégia, baseada em força bruta, define-se uma lista de possíveis valores que cada parâmetro ou hiperparâmetro pode vir a assumir por tipo de modelo considerado. Em seguida, considera-se uma combinação via produto cartesiano de cada um dos possíveis valores estabelecidos para cada parâmetro, que são então utilizados para treinar e testar o modelo. Ao final, os resultados são comparados e a melhor escolha de parâmetros é elencada (BRINK; RICHARDS; FETHEROLF, 2017).

É importante ressaltar que, quanto maior o refinamento dos valores estabelecidos no *grid*, maior será a quantidade de treinamentos e testes a serem efetuados, aumentando o custo computacional. Em contrapartida, tem-se como vantagem aumentar as possibilidades de encontrar um modelo mais adequado ajustado ao problema, com melhor performance e boa capacidade de generalização. De fato, a busca em *grid* nada mais é que uma maneira exaustiva de procurar todas as possíveis combinações de valores para cada parâmetro ajustável em um espaço de busca arbitrariamente grande (BRINK; RICHARDS; FETHEROLF, 2017).

No escopo deste trabalho, foram analisadas a utilização de determinadas variações dos parâmetros e hiperparâmetros de cada modelo com vistas a balancear os seguintes aspectos: (i) as especificações disponíveis na biblioteca *skit-learn*; (ii) o alto custo computacional que uma grande quantidade de combinações poderia causar; e (iii) os parâmetros e hiperparâmetros de maior impacto em cada modelo considerando o problema em questão (SCIKIT-LEARN, 2018). Como resultado, os parâmetros e hiperparâmetros definidos para cada modelo bem como suas variações são discutidos nos parágrafos a seguir. As variações não especificadas foram mantidas em seus valores padrões da biblioteca utilizada.

**Árvores de Decisão.** Um dos conceitos mais importantes na configuração de uma Árvore de Decisão é a entropia e, assim sendo, as duas medidas de entropias disponíveis pela biblioteca *skit-learn* foram acrescentadas à busca em *grid*: entropia de Gini e de Shannon. Além da

entropia, um parâmetro que especifica a estratégia utilizada nos testes de decisão foi variado entre melhor decisão e decisão aleatória. Um outro parâmetro foi modificado de seu valor padrão para pré-ordenar os dados e acelerar a descoberta das melhores divisões da árvore. Ao final, a busca em *grid* resultou em 4 diferentes modelos de Árvore de Decisão.

**Florestas Aleatórias.** De forma análoga às Árvores de Decisão, neste modelo também foram consideradas as duas entropias disponíveis. Entretanto, as Florestas Aleatórias dispõem de outro parâmetro extremamente importante: a quantidade de árvores da floresta, a qual sofreu variações de 5, 9, 10 e 15 árvores. Além destes parâmetros, foi considerada também a utilização ou não das amostras de *bootstrap* para construir as árvores, totalizando então 16 modelos distintos de Florestas Aleatórias.

**Vizinhos mais Próximos.** Neste modelo, a classificação de um determinado objeto baseia-se na classificação majoritária dos  $k$  exemplos (vizinhos) mais próximos ao mesmo, em que  $k$  é um número ímpar. Desse modo, foram consideradas variações dos valores de  $k$  em 3, 5, 9, 11 e 19 vizinhos. O algoritmo utilizado pelo modelo sofreu todas as 4 variações disponíveis pela biblioteca. A busca *grid* contemplou então um total de 20 modelos de Vizinhos mais Próximos.

**AdaBoost, Boosting e Bagging.** Os três modelos em questão não possuem muitos parâmetros livres para serem personalizados e, portanto, não foi possível elencar muitas variações. O modelo *AdaBoost* variou em dois parâmetros, o número de estimadores, com valores iguais a 10, 25 e 50 estimadores, e nos algoritmos, que variaram nas duas possibilidades disponíveis na biblioteca utilizada, obtendo assim um total de 6 modelos. No modelo *Boosting*, apenas a taxa de aprendizado sofreu variações, com valores igual a 0.01, 0.0055 e 0.001, resultando em 3 modelos *Boosting*. Apenas um parâmetro variou também no modelo *Bagging*, o número de estimadores, que recebeu valores iguais a 10, 15 e 20, resultando também em um total de 3 modelos.

**Máquinas Vetores de Suporte.** Este modelo possui diversos parâmetros importantes para serem considerados e variados na busca em *grid*, entretanto, o custo computacional de treina-

mento deste modelo é altíssimo e, portanto, o escopo de combinações da busca foi reduzido. Um dos parâmetros considerados foi o valor de penalidade  $C$  do termo do erro, o qual sofreu variações de 0.5 e 1.0. Outro parâmetro variado foi a função utilizada no algoritmo, em que foram consideradas as funções de base radial e polinomial, sendo que a função polinomial foi mantida com grau 2. Testes empíricos com polinômios de grau 3 e 4 até chegaram a ser realizados, mas o tempo de execução do treinamento mostrou-se intratável. Ao final, obteve-se uma busca em *grid* com 4 modelos de Máquinas Vetores de Suporte.

**Redes Neurais.** A maior quantidade de parâmetros variados ocorreu no modelo de Redes Neurais, pois os diferentes parâmetros deste modelo possuem grande influência no resultado que pode ser alcançado. Foram variadas as funções de ativação, *solvers*, taxa de aprendizado e a quantidade de neurônios nas camadas ocultas. As funções de ativação sigmoide, logística, tangente hiperbólica e retificada linear são bastante conhecidas e utilizadas e, portanto, todas elas foram consideradas. Foram utilizados os *solvers* **sgd** e **adam** e a taxa de aprendizado sofreu variações com valores igual a 0.01, 0.0055 e 0.001. Considerando as ilimitadas possibilidades para definir a quantidade de neurônios das camadas ocultas, as escolhas foram baseadas na Regra de Kolmogorov e na Regra da Pirâmide Geométrica as quais serão descritas a seguir. Considerando as duas regras, 34 neurônios na camada de entrada e 4 neurônios na camada de saída, foram obtidas 325 diferentes arquiteturas com uma e duas camadas ocultas para redes. Todas essas variações resultaram em 5380 Redes Neurais pela a busca em *grid*.

A Regra de Kolmogorov define a quantidade de neurônios em uma camada oculta conforme a Equação 3.4, em que  $N_{hidden}$  é a quantidade de neurônios na camada oculta e  $N_{in}$  é a quantidade de atributos preditores do conjunto de dados, ou seja, a quantidade de neurônios na camada de entrada (PALIT; POPOVIC, 2005). De acordo com esta regra, obteve-se o número de neurônios na única camada oculta como sendo igual a 70.

$$N_{hidden} = 2 \cdot (N_{in} + 1). \quad (3.4)$$

A Regra da Pirâmide Geométrica, por sua vez, define um intervalo de quantidades de

neurônios conforme a Equação 3.5, em que  $N_{hidden}$  é a quantidade de neurônios na camada oculta,  $N_{in}$  é a quantidade de neurônios na camada de entrada,  $N_{out}$  é a quantidade de neurônios na camada de saída e  $0.5 \leq \alpha \leq 2$ . Este intervalo deve ser aplicado em toda sua extensão para uma camada oculta e todas as combinações entre os números desse intervalo constituem as quantidades de neurônios para duas camadas ocultas (PALIT; POPOVIC, 2005).

$$N_{hidden} = \alpha \sqrt{N_{in} \cdot N_{out}}. \quad (3.5)$$

Como resultado deste intervalo, resultou-se que a quantidade de neurônios nas camadas ocultas deve permanecer entre 6 e 23. É interessante ressaltar que, em ambas as regras, o número de neurônios obtidos deveria ser distribuído em uma ou duas camadas ocultas, conforme Teorema da Aproximação Universal das redes neurais *multilayer perceptron* (BRAGA; CARVALHO; LUDERMIR, 2007), o que ensejou na combinação da distribuição do número de neurônios disponíveis às camadas consideradas, resultando em 324 valores distintos para este parâmetro.

### 3.4 Implementação da Solução Proposta

Para a implementação da solução proposta, descrita de forma detalhada nas seções anteriores deste capítulo, foi utilizada a linguagem de programação Python, uma linguagem interpretada, de alto nível e desenvolvida por Guido Van Rossum (BORGES, 2010).

Esta linguagem possui diversas bibliotecas as quais auxiliaram no desenvolvimento deste trabalho. As bibliotecas **pandas** e **numpy** foram essenciais para manipulação e estruturação do conjunto de dados. No que diz respeito ao Aprendizado de Máquina, a biblioteca utilizada em todo o processo foi a **sckit-learn**. Esta biblioteca possui classes para implementar os modelos de Aprendizado de Máquina bem como a busca em *grid* e a avaliação dos modelos.

A busca em *grid* é feita por meio da classe **GridSearchCV** que também realiza, de forma automática, o método de validação cruzada *k-fold*. Esta classe, além das classes dos modelos, possui funções padrão como **fit** e **predict** que realizam o treinamento e a previsão dos modelos, respectivamente. Vale ressaltar que ao final do treinamento, o **GridSearchCV** também



disponibiliza os resultados obtidos durante treino e testes de todos os *folds*, o resultado do melhor modelo, tempo de execução, dentre outros.

Para avaliação dos modelos, utilizou-se o submódulo `sklearn.metrics`, que dispõe de métricas para problemas de diversos tipos, de classificação e regressão no Paradigma Supervisionado, bem como métricas para o Paradigma Não-Supervisionado. No tocante à tarefa considerada, utilizou-se a métrica `F1_score` e também a matriz de confusão via `confusion_matrix`.

# Capítulo 4

## Resultados e Discussão

Em concordância com a metodologia descrita anteriormente, os procedimentos de instanciação dos modelos, treinamentos e testes foram executados utilizando a linguagem de programação Python e suas bibliotecas. Conforme mencionado, os modelos de Aprendizado de Máquina adotados foram submetidos ao processo de busca em *grid*, uma estratégia baseada em força bruta que gera combinações de parâmetros e hiperparâmetros pré-definidos de cada modelo.

Cada combinação fornecida pela busca, utilizando o conjunto de dados descrito na Seção 3.2, foi treinada e testada utilizando o método de validação cruzada *k-fold*, com  $k = 3$ . A métrica aplicada para avaliação dos treinamentos foi o *F-score* do tipo *micro-averaging*, em virtude das diferentes classes encontrarem-se desbalanceadas. As métricas obtidas e também uma comparação destes resultados com a literatura são apresentadas nas seções a seguir.

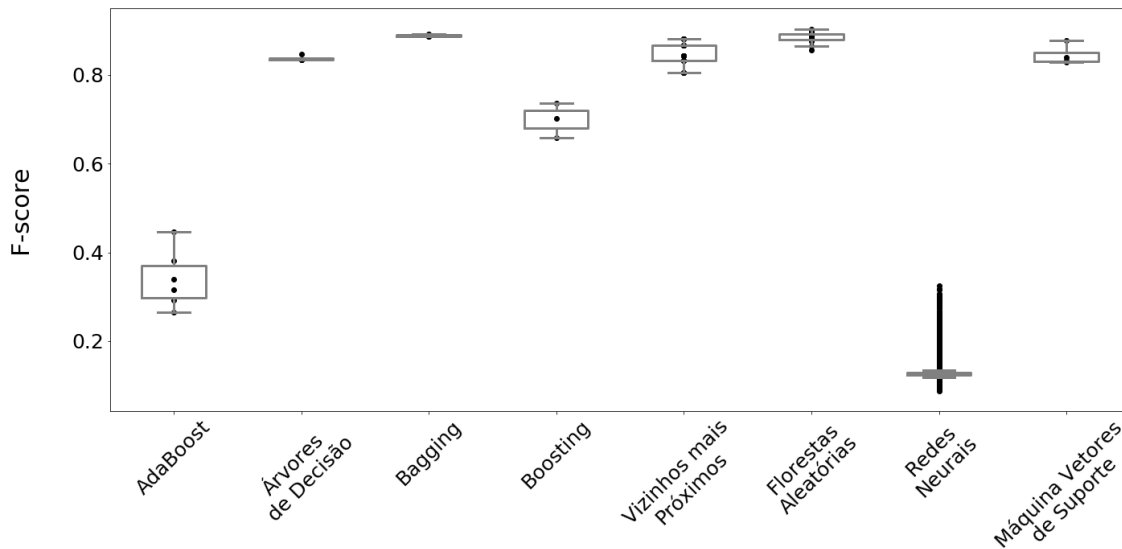
### 4.1 Métricas Obtidas

Conforme mencionado, em um problema de Aprendizado de Máquina há um número virtualmente ilimitado de soluções possíveis para um problema, especialmente quando se leva em conta o ajuste de todos os possíveis parâmetros e hiperparâmetros dos modelos adotados. Assim, optou-se por uma busca em *grid* para procurar ajustes adequados para alguns destes diferentes valores, impondo uma metodologia neste processo. Levando isto em consideração, a busca em *grid* mais abrangente foi segmentada em diferentes buscas em *grid* cada qual consi-

derando um tipo de modelo adotado, facilitando a automatização desta atividade. Assim, cada busca em *grid* foi realizada, ocasionando treinamento e teste dos vários modelos propostos. Com os resultados obtidos, partiu-se então para a análise destes valores, que é descrita a seguir.

Em termos de desempenho no conjunto de testes, considerando a média do  $F$ -scores nos  $k = 3$  *folds*, foi possível primeiramente obter uma visão geral do desempenho por grupo de modelos considerados. O *boxplot* da Figura 4.1 apresenta, para cada tipo de modelo, os  $F$ -scores obtidos. É possível notar que os modelos *AdaBoost* e Redes Neurais não conseguiram, em nenhuma das combinações de parâmetros e hiperparâmetros, alcançar um bom desempenho, ficando abaixo da média dos demais para este mesmo problema. De maneira geral, as combinações dos demais modelos atingiram um  $F$ -score acima de 60%, demonstrando bom rendimento para a tarefa endereçada.

Figura 4.1:  $F$ -scores obtidos em cada modelo.



Dentre os  $F$ -scores observados no gráfico da Figura 4.1, foram elencadas as melhores métricas obtidas em cada busca em *grid* por tipo de modelo, as quais são apresentadas na Tabela 4.1 em ordem decrescente. O modelo com melhor desempenho foi do grupo de Florestas Aleatórias, com  $F$ -score de 0.90351. Entretanto, é interessante notar que a diferença de  $F$ -score entre os quatro primeiros modelos é pequena, sugerindo que a busca em *grid* foi capaz de identificar precisamente o modelo mais adequado para o cenário e que há outros modelos que também

podem se adequar bem para esta tarefa. Em contrapartida, como já observado anteriormente, os modelos *AdaBoost* e Redes Neurais, que tiveram um mal desempenho médio, pontuaram, no máximo, com *F-score* de 50%.

Tabela 4.1: Melhor *F-score* obtido em cada grupo de modelos.

| Modelo                      | Melhor <i>F-score</i> |
|-----------------------------|-----------------------|
| Florestas Aleatórias        | 0.90351321251         |
| Bagging                     | 0.89247747291         |
| Vizinhos mais Próximos      | 0.88184671458         |
| Máquinas Vetores de Suporte | 0.87804075333         |
| Árvores de Decisão          | 0.84661334413         |
| Boosting                    | 0.73574972157         |
| AdaBoost                    | 0.44578313253         |
| Redes Neurais               | 0.32469373291         |

O melhor modelo de Florestas Aleatórias, que atingiu o maior *F-score* dentre todos os valores observados, possuía 15 árvores na floresta, entropia de Shannon e não utilizou *bootstrap*. Os *F-scores* de cada um dos três *folds* foram, respectivamente, 0.9049, 0.8979 e 0.9075. De maneira geral, o grupo das Florestas Aleatórias teve um desempenho superior, o que ocorre em decorrência deste modelo lidar bem com conjuntos desbalanceados. Estes modelos procuram, por meio do voto majoritário, calcular a média de várias Árvores de Decisão profundas, treinadas em diferentes partes do conjunto de dados, com o objetivo de reduzir a variação entre as classes. Outros modelos como *Bagging* também possuem essa característica e também alcançaram bons resultados, o que reforça a hipótese aqui elencada.

De maneira intuitiva, a Floresta Aleatória pode ter tido um bom desempenho neste cenário por conseguir capturar uma analogia com o que um fluente em Libras discrimina ao examinar alguém emitindo uma expressão facial gramatical, em que responde mentalmente a um conjunto de perguntas binárias para determinar a expressão, tais como, “a boca está contraída?”, “a cabeça está inclinada?”, “as sobrancelhas estão franzidas?”, e assim por diante. Considerando o número de árvores na Floresta Aleatória e a heterogeneidade das mesmas, tem-se então um desempenho razoável para a tarefa, superando as Árvores de Decisão individuais.

O método de validação cruzada *k-fold* adotado ajudou a identificar modelos que generalizam

bem evitando problemas como *overfitting*. Durante o procedimento de busca de em *grid*, o melhor modelo foi persistido para ser utilizado em análises posteriores. Adiante, foi realizado um re-treinamento com o modelo utilizando o mesmo conjunto de dados e a mesma métrica, porém com o método de validação cruzada *holdout*, com 70% do conjunto de dados para treinamento e o restante para testes. O resultado obtido foi ligeiramente superior ao obtido por meio do *k-fold*, com *F-score* de 0.90890, o que se justifica também em relação à maior quantidade de exemplos fornecida ao modelo no treinamento. Este procedimento de validação cruzada *holdout* foi conduzido para propiciar uma melhor análise em termos comparativos com a literatura, como será discutido na seção a seguir.

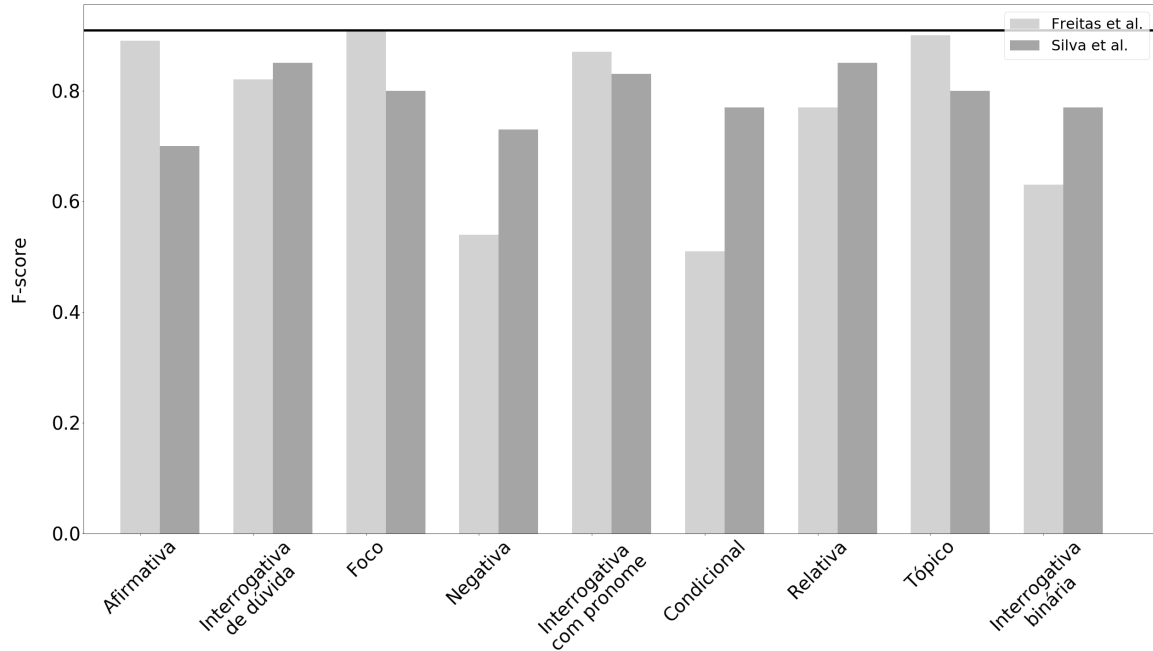
## 4.2 Análise Comparativa com a Literatura

Como apresentado anteriormente na Seção 2.4, há vários trabalhos na literatura que abordam o problema do reconhecimento de Libras. Para posicionar comparativamente os resultados obtidos em relação às contribuições já existentes, permitiu-se destacar os trabalhos de Freitas et al. (FREITAS; BARBOSA; PERES, 2014b) e Silva et al. (SILVA et al., 2017), pois ambos utilizaram o mesmo conjunto de dados para a mesma tarefa de Aprendizado de Máquina.

O histograma da Figura 4.2 apresenta os *F-scores* obtidos pelos dois trabalhos para cada expressão facial gramatical e, na parte superior do gráfico, há uma reta que representa o *F-score* alcançado utilizando o método *holdout* no re-treino do melhor modelo de Florestas Aleatórias, *benchmark* obtido como resultado deste trabalho. É interessante notar que este *F-Score* superou os resultados propostos por Silva et al., e ultrapassou a maioria dos resultados por expressão propostos por Freitas et al., exceto no tocante à expressão facial gramatical Foco, a qual possui valor bastante próximo, igual a 0,91, ou seja, uma diferença de 0,0011 nesta métrica.

No entanto, apesar do conjunto de dados utilizado ser o mesmo, neste trabalho foi realizado um pré-processamento no conjunto de dados com a finalidade de unificar os exemplos e endereçar a tarefa como sendo de classificação multi-classe, enquanto que os demais trabalhos relacionados consideram classificações binárias classe a classe. Para permitir uma comparação mais justa, realizou-se então um processamento dos dados do conjunto de testes e um re-treino

Figura 4.2:  $F$ -scores obtidos por (FREITAS; BARBOSA; PERES, 2014b) e (SILVA et al., 2017) para cada expressão facial gramatical e  $F$ -score alcançado utilizando o método *holdout* no re-treino.



do modelo, com vistas a capturar a efetividade deste último enquanto reconhecedor de cada classe, aplicando uma estratégia do tipo Um-Contra-Todos.

Segundo esta estratégia, o conjunto de dados foi modificado de forma que uma única classe seja evidenciada em cada turno. Para tanto, escolheu-se uma classe de referência, a qual foi mantida com o seu rótulo, enquanto todas as outras classes foram modificadas para ‘0’, adaptando o problema à uma classificação binária. O modelo é então treinado e testado para esta nova entrada e os resultados são aferidos, obtendo-se o  $F$ -Score de cada classe. Segue-se então para a realização do mesmo procedimento com a classe seguinte e assim por diante, até que se tenha abrangido todas as classes do problema.

Os resultados obtidos neste cenário são apresentados nas matrizes de confusão por expressão da Figura 4.3, onde nota-se a prevaência de dados na diagonal principal, sugerindo bom desempenho. Para uma comparação com a literatura, os resultados apresentados na Figura 4.4 sintetizam as métricas coletadas nesta abordagem. É possível notar que o desempenho verificado por meio de todos os  $F$ -scores alcançados superaram o resultado obtido neste trabalho antes da aplicação da estratégia Um-Contra-Todos, representada pela reta neste histograma, e

superaram também os resultados dos trabalhos relacionados em cada respectiva expressão. É interessante notar que este resultado reflete apenas uma nova perspectiva de análise do modelo treinado anteriormente para generalizar todas as expressões faciais.

Figura 4.3: Matrizes de confusão por expressão facial gramatical na estratégia Um-Contra-Todos.

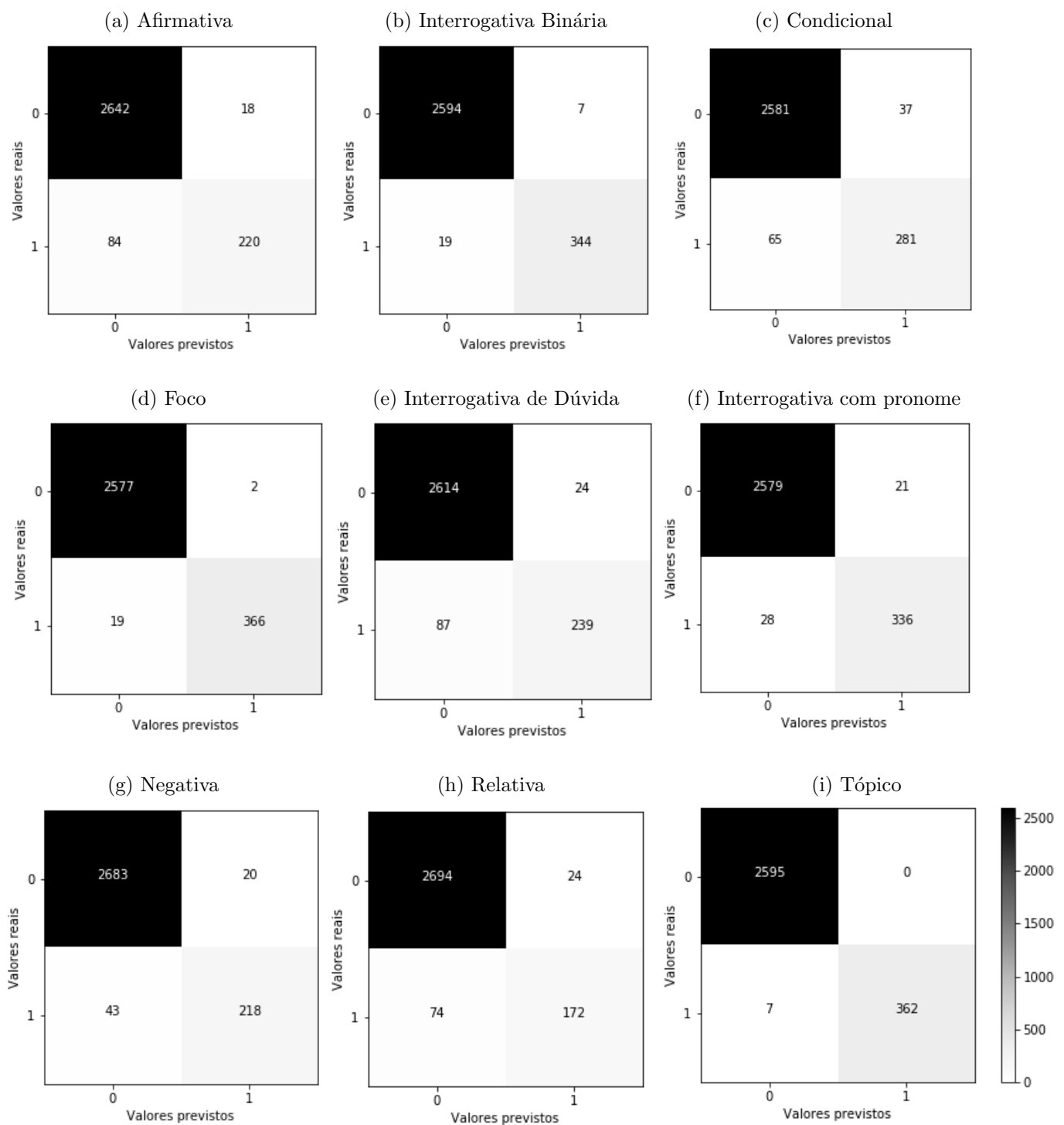
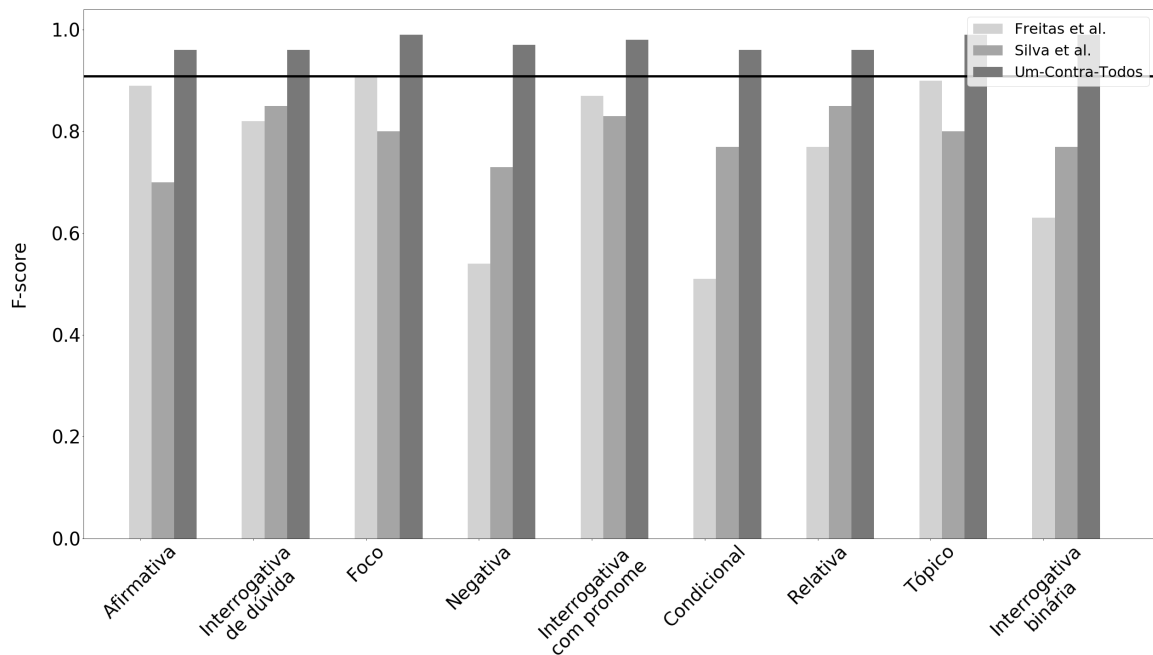


Figura 4.4:  $F$ -scores obtidos após realizar estratégia Um-Contra-Todos.

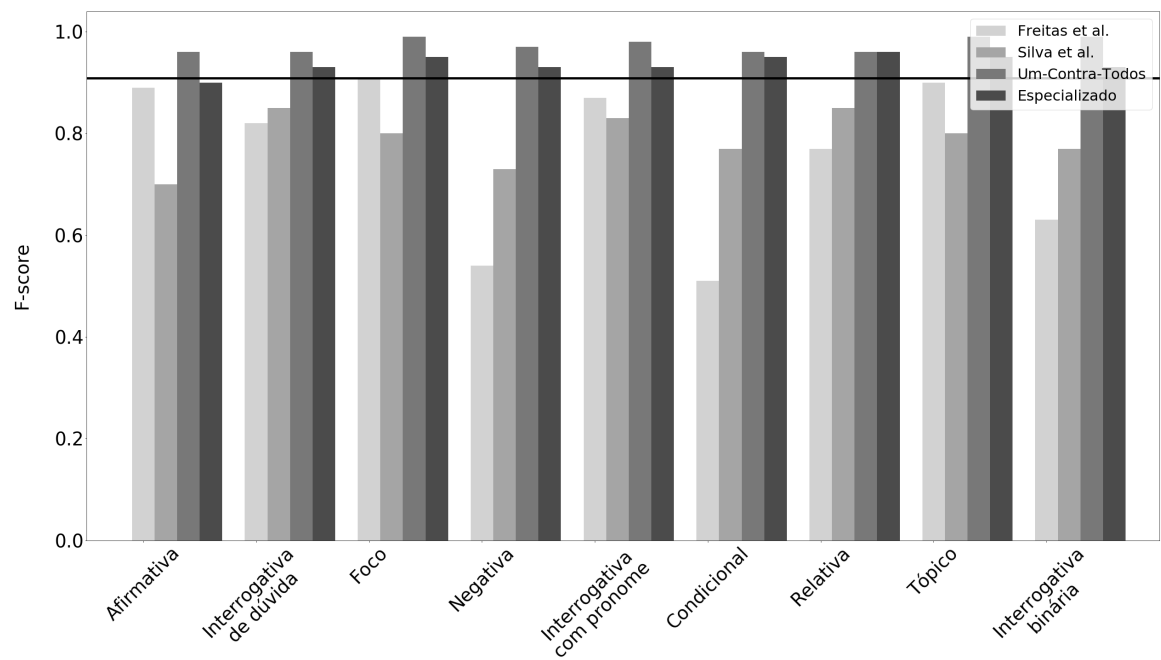
Apesar da aplicação da estratégia Um-Contra-Todos revelar bons resultados, o mesmo procedimento realizado por Freitas et al. e Silva et al. foi reproduzido para viabilizar uma comparação equânime entre os trabalhos. Neste cenário, foram utilizados os conjuntos de dados de cada expressão, em que há exemplos da expressão em si e também exemplos sem expressão alguma. Isto ocorre porque na base de dados original há *frames* em que, embora haja uma sentença em Libras sendo emitida, esta não envolve a utilização da face para fins gramaticais. Assim, ao invés de diferenciar as expressões, foi dada a chance do modelo tornar-se especialista em cada expressão, uma expressão por vez, distinguindo-a do ruído e das demais situações.

Os resultados foram aferidos e podem ser observados na Figura 4.5. Mais uma vez, é possível notar que os resultados desta nova situação superaram os resultados dos trabalhos relacionados em cada respectiva expressão. Entretanto, não superam a estratégia Um-Contra-Todos.

Este resultado verificado, embora pareça contra-intuitivo, é uma contrapartida da especialização do modelo em uma determinada expressão, o que acaba por enfatizar as características da mesma em detrimento do que não são características de interesse, que podem se manifestar de maneira mais diversa, o que impacta negativamente na métrica de desempenho observada.



Figura 4.5: *F-scores* obtidos após refinamento por modelo.



# Capítulo 5

## Considerações Finais

Este trabalho consistiu no reconhecimento de Expressões Faciais Gramaticais da Língua Brasileira de Sinais utilizando as técnicas de Aprendizado de Máquina. O problema foi tratado como uma tarefa de aprendizado supervisionado de classificação, a qual utiliza um conjunto de dados com exemplos de Expressões Faciais Gramaticais que foi submetido a diversos modelos com a finalidade de encontrar resultados satisfatórios.

O conjunto de dados utilizado nos treinamentos dos modelos é realístico e foi elaborado a partir de 225 vídeos onde foram emitidas sentenças de Libras por dois indivíduos, em que foram anotados exemplos de expressões faciais gramaticais de forma supervisionada (FREITAS; BARBOSA; PERES, 2014a). Estes dados passaram por um pré-processamento no qual foram eliminados os atributos irrelevantes. Ao final, o conjunto de dados resultante passou a ter 34 atributos preditores, 9 classes e 9.877 exemplos. É interessante ressaltar que este conjunto de dados foi utilizado em outros trabalhos os quais realizaram tarefas de Aprendizado de Máquina semelhantes, o que enseja comparações.

Considerando o problema em questão, para o treinamento do conjunto de dados foram utilizados modelos de Árvores de Decisão, Métodos *Ensemble*, Vizinhos mais Próximos, Redes Neurais e Máquinas Vetores de Suporte. Estes modelos foram adotados para treinamento a fim de prever uma expressão facial gramatical específica e passaram pelo processo de busca em *grid*, em que uma série de parâmetros e hiperparâmetros foram elencados e então utilizados para treinar os modelos. O treinamento utilizou o método de validação cruzada *k-fold* com

$k = 3$ , considerado para evitar problemas de generalização, como *overfitting*.

Para avaliar os modelos treinados na busca em *grid* foi utilizada a métrica *F-score* do tipo *micro-averaging*, em virtude do desbalanceamento dos dados. O modelo que obteve melhor desempenho foram as Florestas Aleatórias alcançando um *F-score* de 0.9031. Este resultado superou as métricas dos trabalhos de Freitas et al. e Silva et al., os quais utilizaram o mesmo conjunto de dados e trataram também de uma tarefa de classificação (FREITAS; BARBOSA; PERES, 2014b; SILVA et al., 2017).

Entretanto, o tratamento do conjunto de dados deste trabalho foi diferenciado em relação aos trabalhos relacionados e, portanto, foram utilizadas outras estratégias para adequar o problema para uma comparação equânime os mesmos. Para tanto, o modelo foi treinado e testado diversas outras vezes com algumas modificações no conjunto de dados e utilizando o método de validação cruzada *holdout*, com 70% dos dados para treino e o restante para testes. Os resultados obtidos superaram o estado da arte em todos os diferentes cenários, considerando as diversas expressões faciais gramaticais.

Em trabalhos futuros, pretende-se analisar mais pontos da face a acrescentá-los aos 17 pontos considerados. Além disso, será considerada a utilização de mais usuários para consolidar um conjunto de dados mais abrangente e robusto, incluindo, por exemplo, as expressões faciais afetivas.

# Referências Bibliográficas

ALMEIDA, M. A. de et al. Utilização de visão computacional para auxílio ao reconhecimento de sinais de libras. In: *Anais do VII Seminário de Iniciação Científica do IFNMG*. Montes Claros, Minas Gerais: SIC, 2018.

ARAUJO, N. et al. Previsão do volume mensal de precipitações em manaus, amazonas com redes neurais aritificiais. In: *Anais do IV Encontro Regional de Informática da Região Norte I*. Manaus, Amazonas: ERIN, 2017. p. 108–116.

BORGES, L. E. *Python para Desenvolvedores*. Rio de Janeiro, Rio de Janeiro: Novatec, 2010.

BOUZADA, M.; RIBEIRO, L.; PEIXE, J. *Métodos Quantitativos Aplicados a Casos Reais*. Rio de Janeiro, Rio de Janeiro: Elsevier Editora Ltda., 2013.

BRAGA, A. de P.; CARVALHO, A. P. de Leon F. de; LUDERMIR, T. B. *Redes Neurais Artificiais: Teoria e Aplicações*. 2. ed. Rio de Janeiro, Rio de Janeiro: LTC, 2007.

BRASIL. *Lei N°. 10.436, de 24 de Abril de 2002*. 2002. Disponível em <[http://www.planalto.gov.br/ccivil\\_03/LEIS/2002/L10436.htm](http://www.planalto.gov.br/ccivil_03/LEIS/2002/L10436.htm)>. Acesso em 3 de dezembro de 2018.

BRECAILO, S. de F. *Expressão Facial e Corporal na comunicação em Libras*. Curitiba: IMAP, 2012. Disponível em <[goo.gl/Lw5KYD](http://goo.gl/Lw5KYD)>. Acessado em 3 de dezembro de 2018.

BRIET, R.; POVOA, L. V.; SANTOS, L. B. R. dos. Classificação de expressões faciais utilizando deep learning para interpretação da língua brasileira de sinais. In: *Anais do 7º Congresso de Inovação, Ciência e Tecnologia do IFSP*. São Paulo, São Paulo: Conict, 2016.

BRINK, H.; RICHARDS, J. W.; FETHEROLF, M. *Real-World Machine Learning*. New York, Estados Unidos: Manning Publications, 2017.

CAEXETA, G. S. *VisionDraughts - Um Sistema de Aprendizagem de Jogos de Damas em Redes Neurais, Diferenças Temporais, Algoritmos Eficientes de Busca em Árvore e Informações Perfeitas Contidas em Bases de Dados*. Uberlândia, Minas Gerais: Universidade Federal de Uberlândia, 2008.

CARVALHO, C. E. S. de. *Reconhecimento de Caracteres Manuscritos Utilizando Redes Neurais Artificiais*. Brasília, Distrito Federal: Centro Universitário de Brasília, 2006.

CYBENKO, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, v. 2, p. 303–314, December 1989.

DELICATO, F.; PIRES, P.; SILVEIRA, I. Jornada de Atualização em Informática 2017. In: CSBC. *Congresso da Sociedade Brasileira de Computação*. Porto Alegre, Rio Grande do Sul: SBC, 2017. cap. 6, p. 212–260.

FACELI, K. et al. *Inteligência Artificial: Uma Abordagem de Aprendizado de Máquina*. 1. ed. Rio de Janeiro, Rio de Janeiro: LTC, 2011.

FLORES, E. M.; BARBOSA, J. L. V.; RIGO, S. J. Um estudo de técnicas aplicadas ao reconhecimento da língua de sinais: novas possibilidades de inclusão digital. *Revista Novas Tecnologias na Educação*, v. 10, n. 3, 2012.

FREITAS, F. de A.; BARBOSA, F. V.; PERES, S. M. *Grammatical Facial Expressions Data Set*. 2014. Disponível em <<https://archive.ics.uci.edu/ml/datasets/Grammatical+Facial+Expressions>>. Acesso em 3 de dezembro de 2018.

FREITAS, F. de A.; BARBOSA, F. V.; PERES, S. M. Grammatical facial expressions recognition with machine learning. In: *International Florida Artificial Intelligence Research Society Conference*. Flórida, Estados Unidos: AAAI Publications, 2014. p. 180–185.

G1. *Eleições 2018 no Espírito Santo*. 2018. Disponível em <<https://g1.globo.com/es/espírito-santo/eleicoes/2018/noticia/2018/09/07/tre-suspende-programas-eleitorais-com-interprete-de-libras-acusado-de-inventar-sinais-no-espírito-santo.ghtml>>. Acesso em 3 de dezembro de 2018.

GCLOUD. *Google Cloud*. 2018. Disponível em <<https://cloud.google.com/>>. Acesso em 3 de dezembro de 2018.

GOLDSHIMIDT, R.; PASSOS, E. *Data mining: um guia Prático*. Rio de Janeiro, Rio de Janeiro: Elsevier Editora, 2005.

GUERRA, R. R. et al. Facial expressions analysis in brazilian sign language for sign recognition. In: *Encontro Nacional de Inteligência Artificial e Computacional*. São Paulo, São Paulo: Eniac, 2018. p. 216–227.

GULLI, A.; PAL, S. *Deep Learning with Keras*. 1. ed. Birmingham, West Midlands: Packt, 2017.

HAYKIN, S. S. *Neural Networks and Learning Machines*. 3. ed. Upper Saddle River, New Jersey: Perason, 2009.

IBGE. *Censo de 2010*. 2010. Disponível em <<https://censo2010.ibge.gov.br/>>. Acesso em 3 de dezembro de 2018.

JOURNAL, T. W. S. *In a Sign of the Times, Ukrainian TV Interpreter Makes Bold On-Air Move*. 2004. Disponível em <[InaSignoftheTimes, UkrainianTVInterpreterMakesBoldOn-AirMove](#)>. Acesso em 3 de dezembro de 2018.

JUNIOR, J. D. B. *Tradução Automática de Línguas de Sinais: do Sinal para a Escrita*. 2016. Trabalho de Conclusão de Curso da Universidade Federal do Pampa. Universidade Federal do Pampa.

- KUBAT, M. *An Introduction to Machine Learning*. Estados Unidos: Springer, 2015.
- LAZZAROTTO, R. *Sistema de Reconhecimento de Padrões do Alfabeto da Língua Brasileira de Sinais Utilizando Microcontrolador*. 2016. Trabalho de Conclusão de Curso da Universidade Tecnológica Federal do Paraná. Universidade Tecnológica Federal do Paraná.
- LIMA, P. M. de. *Redes Neurais para Predição de Séries Temporais de Precipitação em Manaus, Amazonas*. 2016. Trabalho de Conclusão de Curso da Universidade do Estado do Amazonas. Universidade do Estado do Amazonas.
- MARSLAND, S. *Machine Learning An Algorithmic Perspective*. Florida, Estados Unidos: Taylor and Francis, 2015.
- NUMPY. *NumPy*. 2018. Disponível em <<http://www.numpy.org/>>. Acesso em 3 de dezembro de 2018.
- PALIT, A. K.; POPOVIC, D. *Computational Intelligence in Time Series Forecasting - Theory and Engineering Applications*. 1. ed. Londres: Springer, 2005.
- PANDAS. *Pandas*. 2018. Disponível em <<https://pandas.pydata.org/>>. Acesso em 3 de dezembro de 2018.
- PEREIRA, R. et al. Abordagem deep learning para classificação de lesões mamárias. In: *Anais do XXXVI Congresso da Sociedade Brasileira de Computação*. Porto Alegre, Rio Grande do Sul: CSBC, 2016. p. 2597–2600.
- PORFÍRIO, A. J. *Reconhecimento das Configurações de Mão da Libras a partir de Malhas 3D*. Curitiba, Paraná: Universidade Federal do Paraná, 2013.
- QUADROS, R. M.; KARNOPP, L. B. *Língua de sinais brasileira: estudos lingüísticos*. Porto Alegre: Artmed Editora, 2004. Capítulo 4.
- QUADROS, R. M. de. *Língua de Herança: Língua Brasileira de Sinais*. Porto Alegre, Rio Grande do Sul: Penso, 2017.
- RAMOS, C. R. Libras: a língua de sinais dos surdos brasileiros. *Revista Virtual de Cultura Surda e diversidade*, Editora Arara Azul Ltda, Petrópolis, Rio de Janeiro, n. 4, p. 1–15, 2004.
- REZENDE, T. M. *Aplicações de Técnicas de Inteligência Computacional para Análise da Expressão Facial em Reconhecimento de Sinais de Libras*. Belo Horizonte, Minas Gerais: Universidade Federal de Minas Gerais, 2016.
- REZENDE, T. M. et al. Análise da expressão facial em reconhecimento de sinais de libras. In: *Anais do XIII Simpósio Brasileiro de Automação Inteligente*. Porto Alegre, Rio Grande do Sul: SBAI, 2017. p. 465–470.
- RODRIGUES, C. S.; VALENTE, F. *Aspectos Linguísticos da Libras*. Curitiba, Paraná: IESDE, 2011.
- ROJAS, R. *Neural Networks: A Systematic Introduction*. 1. ed. Heidelberg, Berlin: Springer, 1996.

ROSENBLAT, F. *Principles of Neurodynamics: Perceptrons and the theory of brain mechanisms*. Washington, DC, 1961. 626 p.

SCIKIT-LEARN. *Scikit-learn*. 2018. Disponível em <<http://scikit-learn.org/stable/index.html>>. Acesso em 3 de dezembro de 2018.

SILVA, A. G. et al. Classificação de expressões faciais negativas na língua brasileira de sinais. In: *Anais do IV Encontro Regional de Informática da Região Norte I*. Manaus, Amazonas: ERIN, 2017. p. 80–88.

SILVA, G. da et al. Classification of malignancy of lung nodules in ct images using convolutional neural network. In: *Anais do XXXVI Congresso da Sociedade Brasileira de Computação*. Porto Alegre, Rio Grande do Sul: CSBC, 2016. p. 2481–2489.

SOUSA, D. V. C. Um olhar sobre os aspectos linguísticos da língua brasileira de sinais. *Littera Online*, v. 1, n. 2, p. 88–100, 2010.

SOUSA, R. et al. Redes neurais artificiais aplicadas à previsão antecipada de precipitações na região central de manaus. In: *Anais do IV Encontro Regional de Informática da Região Norte I*. Manaus, Amazonas: ERIN, 2017. p. 89–97.

TEODORO, B. T. *Sistema de Reconhecimento Automático de Língua Brasileira de Sinais*. Dissertação (Mestrado) — Escola de Artes, Ciências e Humanidades, Universidade de São Paulo, São Paulo, 2015.

TINOS, R. *Detecção e Diagnóstico de Falhas em Robôs Manipuladores via Redes Neurais Artificiais*. São Carlos, São Paulo: Universidade de São Paulo, 1999.

ZHOU, Z. *Ensemble Methods: Foundations and Algorithms*. Boca Raton, Florida: CRC Press, 2012. (Chapman & Hall/CRC machine learning & pattern recognition series).