ISEL
INSTITUTO SUPERIOR DE ENGENHARIA DE LISBOA


# Aplicação de Tradução de Língua Gestual Portuguesa (LGP) para Português com Recurso a Inteligência Artificial

**GONÇALO CARÉ FONSECA**
(Licenciado)

Trabalho de Projeto para obtenção do Grau de Mestre em
Engenharia Informática e Multimédia

**Orientadores:** Doutor Rui Jesus
Doutor Gonçalo Marques

**Júri:**

**Presidente:** Doutor Pedro Santos
**Vogais:** Doutor Pedro Jorge
Doutor Gonçalo Marques

**Janeiro 2025**


# Aplicação de Tradução de Língua Gestual Portuguesa (LGP) para Português com Recurso a Inteligência Artificial

**GONÇALO CARÉ FONSECA**

(Licenciado)

Trabalho de Projeto para obtenção do Grau de Mestre em
Engenharia Informática e Multimédia

**Orientadores:** Doutor Rui Jesus, DEETC/ISEL
Doutor Gonçalo Marques, DEETC/ISEL

**Júri:**

**Presidente:** Doutor Pedro Santos, DEETC/ISEL
**Vogais:** Doutor Pedro Jorge, DEETC/ISEL
Doutor Gonçalo Marques, DEETC/ISEL

**Janeiro 2025**

# Agradecimentos

Gostaria de expressar a minha gratidão a todos aqueles que, de uma forma ou de outra, contribuíram para a realização desta tese.

Em primeiro lugar, gostaria de agradecer aos meus orientadores, o Professor Rui Jesus e o Professor Gonçalo Marques, pela orientação, apoio e conselhos valiosos ao longo deste percurso. Mesmo com todos os percalços que aconteceram, o vosso apoio, compreensão e disponibilidade foram fundamentais para o desenvolvimento deste trabalho.

Quero também agradecer aos meus amigos que, de diversas formas, contribuíram para esta jornada. Um especial agradecimento à Madalena, Rúben, Gustavo, Afonso, Inês, Ana, Maria Inês, Miguel, Sofia e José, que dedicaram o seu tempo para participar na recolha de fotos necessárias para a obtenção das imagens das letras de Língua Gestual Portuguesa que fizeram parte deste trabalho. Para além disso, quero expressar a minha gratidão ao Rúben, Tiago e Sofia, que estiveram sempre disponíveis para discutir ideias e resolver dúvidas, proporcionando-me um apoio enorme.

Agradeço de forma especial à minha família, que me deu forças nos momentos mais desafiantes. Aos meus pais e avós, pelo apoio incondicional, e à minha namorada, Madalena, por todo o apoio e paciência ao longo deste processo.

Quero também reconhecer e agradecer aos meus colegas de trabalho da empresa 2Logical, especialmente ao meu chefe, Gabriel, por todo o suporte e flexibilidade, que me permitiram conciliar as minhas responsabilidades profissionais com o desenvolvimento desta tese.

Finalmente, agradeço a todas as pessoas envolvidas, especialmente àquelas que, sem hesitar, tiraram tempo do seu dia para ajudar com as fotografias dos gestos de Língua Gestual Portuguesa, mesmo sem conhecerem os gestos. A vossa paciência e dedicação foram fundamentais para a conclusão deste trabalho.

A todos, o meu sincero obrigado.

**Declaração de integridade**

Declaro que esta **dissertação** é o resultado da minha investigação pessoal e independente. O seu conteúdo é original e todas as fontes listadas nas referências bibliográficas foram consultadas e estão devidamente mencionadas no texto. Mais declaro que todas as referências científicas e técnicas relevantes para o desenvolvimento do trabalho estão devidamente citadas e constam das referências bibliográficas.

O autor

---

Lisboa, 05 de janeiro de 2025

# Resumo

Neste trabalho, propõe-se o desenvolvimento de um sistema para a tradução em tempo real de gestos da Língua Gestual Portuguesa (LGP) para texto, com o objetivo de facilitar a comunicação entre pessoas surdas e ouvintes. O reconhecimento automático de gestos é uma área fundamental para promover a inclusão social, especialmente pela disponibilidade crescente de ferramentas tecnológicas que melhorem a acessibilidade comunicacional.

A solução apresentada envolve a criação de uma aplicação móvel que reconhece em tempo real os gestos das letras da Língua Gestual Portuguesa (LGP), exibindo a tradução quase instantaneamente após o gesto ser realizado. Esta aplicação integra dois métodos principais: o primeiro utiliza o MediaPipe, onde a imagem capturada pela câmara do dispositivo é enviada para um servidor remoto. O servidor processa a imagem, deteta os pontos-chave (*landmarks*) das mãos e utiliza um *Multilayer Perceptron* (MLP) para classificar as letras. O segundo método é um modelo de Rede Neuronal Convolucional (CNN) que analisa diretamente as imagens capturadas pela câmara, sendo otimizado para funcionar em dispositivos móveis através do *TensorFlow Lite*. A aplicação foi desenvolvida em Flutter, oferecendo uma interface intuitiva e acessível.

Os testes realizados demonstraram que, com a utilização combinada dos dois modelos, ou seja, o modelo baseado no *MediaPipe* integrado com Multilayer Perceptron (MLP) e o modelo Rede Neuronal Convolucional (CNN), a aplicação é capaz de reconhecer gestos com boa precisão e rapidez, traduzindo gestos em tempo real de forma eficiente. A abordagem adotada permite flexibilidade na adaptação a diferentes cenários e condições de uso, sendo uma solução escalável que pode ser expandida para reconhecer outros gestos ou línguas gestuais. As implicações deste trabalho são significativas para a inclusão social, promovendo a comunicação entre diferentes grupos linguísticos e facilitando a integração das comunidades surdas na sociedade.

**Palavras-chave:** Língua Gestual Portuguesa, Reconhecimento de Gestos, Tradução de Gestos, Aplicação Móvel, Redes Neuronais, TensorFlow Lite, MediaPipe, Inclusão Social

# Abstract

This work proposes the development of a system for real-time translation of Portuguese Sign Language (LGP) gestures into text, with the aim of facilitating communication between deaf and hearing individuals. Automatic gesture recognition is a crucial field for promoting social inclusion, particularly given the growing need for technological tools that improve communication accessibility.

The proposed solution involves the creation of a mobile application capable of recognizing LGP letter gestures in real-time. The application integrates two primary methods: the first uses MediaPipe, where the image captured by the device's camera is sent to a remote server. The server processes the image, detects key hand points (*landmarks*), and uses a MLP to classify the letters. The second method is a Convolutional Neural Networks (CNN) model that directly analyzes the images captured by the camera, optimized for mobile devices using *TensorFlow Lite*. The application was developed in Flutter, offering an intuitive and accessible interface.

The tests conducted demonstrated that, through the combined use of both models, namely the *MediaPipe* integrated with an Multilayer Perceptron (MLP) and the Convolutional Neural Network (CNN) model, the application can recognize gestures with good accuracy and speed, translating them into text efficiently in real-time. This approach allows flexibility in adapting to different scenarios and conditions, providing a scalable solution that can be expanded to recognize other gestures or sign languages. The implications of this work are significant for social inclusion, fostering communication between different linguistic groups and aiding the integration of the deaf community into society.

**Keywords:** Portuguese Sign Language, Gesture Recognition, Gesture Translation, Mobile Application, Neural Networks, TensorFlow Lite, MediaPipe, Social Inclusion

# Índice

# Índice de Figuras

# Índice de Tabelas

# Glossário

| | | |
|---|---|---|
| data augmentation | Aumento de dados | 11 |
| deep learning | Aprendizagem profunda | 11 |
| dropout | Técnica de regularização utilizada em redes neuronais para prevenir o overfitting, desligando aleatoriamente neurónios durante o treino. | 37 |
| landmarks | Pontos-chave de referência que descrevem a posição de articulações ou partes específicas, como os dedos nas mãos, utilizados para a detecção de gestos. | xvii |
| MediaPipe | Framework de visão computacional desenvolvido pela Google, usado para detetar landmarks e realizar análise de gestos e movimentos em tempo real. | xix |
| Multilayer Perceptron (MLP) | Tipo de rede neuronal composta por múltiplas camadas densas, usada para classificação de dados com base em padrões não lineares. | ix |
| overfitting | Fenómeno onde o modelo treinado memoriza os dados de treino, mas perde a capacidade de generalizar para novos dados. | 33 |
| Rede Neuronal Convolucional (CNN) | Tipo de rede neuronal utilizada principalmente para processamento e reconhecimento de imagens, com grande aplicação em classificação de imagens e vídeos. | ix |
| TensorFlow Lite (tflite) | Versão leve do TensorFlow, otimizada para execução de modelos em dispositivos móveis e embutidos, com foco na eficiência de recursos e velocidade. | 56 |
| transfer learning | Aprendizagem por transferência | 11 |

# Siglas

| | |
|---|---|
| API | Application Programming Interface (Interface de Programação de Aplicações) 44, 55 |
| ASL | *Americal Sign Language* (Língua Gestual Americana) 2, 7, 8, 9, 17, 60 |
| CNN | Rede Neuronal Convolucional xvii, xix, 2, 3, 4, 11, 12, 16, 17, 18, 19, 20, 31, 32, 33, 34, 36, 40, 43, 45, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 58, 59 |
| LGP | Língua Gestual Portuguesa ix, xvii, 1, 2, 3, 4, 9, 10, 11, 12, 14, 15, 16, 17, 18, 19, 20, 21, 23, 24, 25, 26, 27, 28, 29, 31, 32, 33, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 47, 49, 53, 55, 56, 58, 59, 60, 61 |
| MLP | *Multilayer Perceptron* ix, xi, xvii, 2, 12, 17, 31, 36, 37, 38, 39, 50, 55, 58 |
| ReLU | Rectified Linear Unit, uma função de ativação utilizada em redes neuronais 33, 38 |
| tflite | TensorFlow Lite 42, 43, 44, 45, 56, 57, 58, 59, 60 |

# 1 Introdução

A comunicação é uma necessidade fundamental para a integração social, no entanto, para as comunidades surdas, comunicar eficazmente pode ser um desafio, uma vez que a maioria das interações sociais dependem de línguas faladas e escritas. A Língua Gestual Portuguesa é a forma natural de comunicação para muitas pessoas surdas em Portugal, permitindo-lhes não só expressar pensamentos e emoções, mas também conversar, interagir e partilhar ideias de forma fluida no seu dia a dia. Contudo, a falta de ferramentas tecnológicas que facilitem a interação entre surdos e ouvintes continua a ser uma barreira para a inclusão social, dificultando o entendimento entre estas duas comunidades.

O desenvolvimento de soluções tecnológicas que permitam o reconhecimento em tempo real da LGP pode ter um impacto significativo na promoção da inclusão social. Ferramentas que possibilitem o reconhecimento automático dos gestos da LGP em tempo real podem facilitar a comunicação entre surdos e ouvintes, oferecendo uma forma de tradução acessível e eficiente. No entanto, construir um sistema capaz de realizar essa tarefa exige superar desafios técnicos, como reconhecer de forma consistente os gestos, mesmo em condições variáveis de iluminação, ângulos de captura ou movimentos rápidos, enquanto mantém tempos de resposta suficientemente curtos para garantir uma interação fluida entre o utilizador e a aplicação.

## 1.1 Contexto

A língua gestual é uma forma rica e complexa de comunicação, sendo o principal meio de interação para as comunidades surdas. Ao contrário das línguas faladas, que utilizam sons para transmitir significados, a língua gestual recorre a gestos, expressões faciais e movimentos corporais. Estes elementos desempenham um papel crucial na comunicação de conceitos e emoções, além de serem parte da identidade cultural das comunidades surdas.

A língua gestual tem vindo a ser reconhecida como língua oficial em diversos países, incluindo Portugal, o que representa um passo importante para a inclusão social. Este reconhecimento permite que as pessoas surdas acedam a serviços e oportunidades numa língua que lhes é natural. Contudo, a comunicação entre surdos e ouvintes continua a ser um desafio em contextos onde não há intérpretes disponíveis.

A tradução automática da LGP para texto surge como uma solução promissora para resolver este problema, facilitando a comunicação entre surdos e ouvintes e promovendo uma maior inclusão social. Contudo, a natureza visual e espacial da LGP, combinada com a necessidade de reconhecer e interpretar corretamente gestos subtis, apresenta desafios técnicos. Ferramentas como o MediaPipe, da Google, têm demonstrado potencial para capturar pontos-chave das mãos, mas ainda existem limitações a serem superadas para alcançar uma interpretação precisa da LGP em tempo real.

## 1.2 O Problema

Apesar de a LGP ser reconhecida como uma língua oficial em Portugal, a barreira de comunicação entre surdos e ouvintes permanece significativa. Muitas instituições e serviços públicos não disponibilizam intérpretes de língua gestual, o que restringe o acesso de pessoas surdas a serviços essenciais e limita a sua participação plena na sociedade. Mesmo com o reconhecimento da LGP, a falta de tecnologias eficazes que possam auxiliar na tradução em tempo real dessa língua agrava o problema.

A tradução automática de línguas gestuais apresenta desafios técnicos distintos. Ao contrário das línguas faladas, que seguem uma estrutura linear baseada em sons, a LGP é visual e espacial, dependendo de gestos manuais, expressões faciais e movimentos corporais para transmitir significados. Esta complexidade requer que os sistemas de tradução sejam capazes de lidar com uma ampla variabilidade de gestos, rápida execução dos movimentos e contextos subtis que podem alterar o significado dos gestos.

Atualmente, as soluções existentes ainda não são suficientemente robustas para capturar todas as nuances da LGP de forma precisa. Além disso, a maioria das soluções foca-se em línguas gestuais mais amplamente utilizadas, como a *Americal Sign Language* (Língua Gestual Americana) (ASL), o que deixa a LGP com recursos tecnológicos limitados. Assim, há uma necessidade de desenvolver uma aplicação específica para o reconhecimento das letras da LGP, utilizando inteligência artificial para facilitar a comunicação entre surdos e ouvintes em Portugal.

O código-fonte completo deste projeto está disponível publicamente no repositório GitHub através do link `https://github.com/souocare/thesis_portuguese_sign_language`

## 1.3 Objetivos

O principal objetivo desta tese é desenvolver uma aplicação que, utilizando a câmara do telemóvel permita a tradução, em tempo real, das letras da LGP, facilitando uma maior integração entre as comunidades surda e ouvinte. Este objetivo será alcançado através da criação de dois modelos de reconhecimento: um modelo de CNN para o reconhecimento direto das imagens das letras gestuais e um segundo modelo utilizando o MediaPipe, que deteta os pontos-chave das mãos e utiliza um MLP para classificar as letras.

A aplicação será implementada para funcionar em dispositivos móveis, permitindo que, ao apontar a câmara para uma pessoa que está a realizar gestos da LGP, o sistema seja capaz

de identificar automaticamente qual a letra que está a ser gesticulada, convertendo-a em texto.

### 1.3.1 Objetivos Específicos

1. **Desenvolver um modelo de CNN** baseado num *dataset* exclusivo de letras da LGP, com o objetivo de reconhecer com alta precisão as diferentes variações das letras gestuais, mesmo com diversidades nos gestos.

2. **Explorar o uso do MediaPipe para a deteção de pontos-chave das mãos**, criando um segundo modelo que utilize esses pontos para identificar as letras gestuais, abordando as limitações de variabilidade e precisão na deteção dos marcos.

3. **Integrar os modelos desenvolvidos numa aplicação**, que permita o reconhecimento em tempo real das letras gestuais. A integração será realizada em dois cenários: (a) execução do modelo localmente, com desafios de processamento e compatibilidade; e (b) execução via servidor, lidando com latência e conectividade.

4. **Validar os modelos desenvolvidos** através de vários testes com dados não presentes no *dataset* de treino, garantindo a robustez do sistema em condições reais, bem como testar a aplicação com utilizadores reais para avaliar a usabilidade e eficiência.

5. **Contribuir para a comunidade** ao criar uma aplicação que facilita a aprendizagem e comunicação em LGP, abordando uma necessidade significativa no contexto português, onde os recursos tecnológicos focados na LGP são limitados.

### 1.3.2 Impacto Esperado

O impacto esperado deste trabalho é tanto social quanto tecnológico. Do ponto de vista social, espera-se que a aplicação promova uma maior inclusão entre as comunidades surda e ouvinte em Portugal, facilitando a comunicação e reduzindo barreiras linguísticas.A aplicação poderá também servir como uma ferramenta educativa para aqueles que desejam aprender a LGP, oferecendo funcionalidades como uma visualização detalhada de cada gesto associado às letras do alfabeto, instruções visuais passo a passo e a possibilidade de praticar e validar os gestos em tempo real. Estas características não só facilitam o processo de aprendizagem, mas também promovem a disseminação e aceitação da LGP na sociedade, ao permitir que pessoas ouvintes adquiram uma compreensão prática da língua.

Do ponto de vista tecnológico, o desenvolvimento de modelos de inteligência artificial específicos para o reconhecimento de letras da LGP contribuirá para o avanço da tradução automática de línguas gestuais. A criação de uma solução eficaz e escalável poderá servir de base para futuras pesquisas e aplicações, tanto para expandir a cobertura para outras línguas gestuais, como para melhorar a precisão e funcionalidade do reconhecimento de gestos em tempo real.

## 1.4 Estrutura

Esta tese está organizada em capítulos que descrevem, de forma sistemática, todas as etapas do desenvolvimento deste projeto, desde a conceção inicial até à avaliação final dos resultados e propostas futuras.

O **Capítulo 1, Introdução**, apresenta o contexto do estudo, destacando a relevância da tradução automática de gestos da LGP para texto e os principais desafios envolvidos. Este capítulo também expõe os objetivos gerais e específicos da investigação e o impacto esperado nas áreas de tecnologia e inclusão social.

O **Capítulo 2, Fundamentos e Trabalho Relacionado**, discute os principais conceitos que sustentam o projeto, como *machine learning*, *deep learning*, e *transfer learning*, explicando como estes se aplicam ao problema de reconhecimento de gestos. Além disso, é apresentada uma revisão abrangente de trabalhos anteriores, destacando soluções desenvolvidas em contextos semelhantes. As limitações identificadas nas soluções existentes ajudam a moldar o foco deste projeto.

O **Capítulo 3, Metodologia**, descreve em detalhe as abordagens metodológicas adotadas ao longo do desenvolvimento do projeto. O capítulo começa com a apresentação da arquitetura do sistema, seguida pela recolha e preparação dos dados, onde são discutidos os métodos de recolha de imagens e vídeos, assim como o pré-processamento dos dados para garantir a robustez do modelo. São também descritos os modelos de aprendizagem criados, nomeadamente a Rede Neuronal Convolucional (CNN) e o uso de uma Framework de visão computacional para deteção de gestos e *landmarks* (??) para a deteção de pontos das mãos com a integração de um modelo Multilayer Perceptron (MLP). Finalmente, o desenvolvimento da aplicação em Flutter é explicado, destacando as suas principais funcionalidades e como foram integrados os modelos.

O **Capítulo 4, Aquisição e Pré-processamento de Dados**, detalha todo o processo de recolha de dados e pré-processamento. O capítulo cobre a captura de imagens e vídeos dos gestos da LGP, o uso do MediaPipe para identificar as regiões das mãos, e a aplicação de técnicas de *data augmentation*. Além disso, é abordada a organização dos dados em conjuntos de treino, validação e teste.

No **Capítulo 5, Implementação**, são apresentadas as etapas de desenvolvimento da aplicação e a integração dos modelos de classificação. Este capítulo inclui a explicação do desenvolvimento da interface utilizador em Flutter, a integração dos modelos CNN e MediaPipe, e a arquitetura do servidor Flask e, por fim, os principais desafios técnicos existentes.

O **Capítulo 6, Avaliação**, concentra-se na análise e avaliação dos resultados obtidos pelos modelos desenvolvidos e pela aplicação. São discutidos os testes realizados para medir o desempenho do sistema em diferentes cenários, incluindo a análise da sua capacidade de generalização. Também são apresentados os resultados da avaliação da aplicação em termos de eficiência e experiência de utilizador, realizada em diversos dispositivos e condições de uso.

Por fim, o **Capítulo 7, Conclusão e Trabalhos Futuros**, sintetiza os principais contributos deste trabalho. São discutidos possíveis melhoramentos e expansões futuras.

# 2 Fundamentos e Trabalho Relacionado

## 2.1 A Lingua Gestual

A língua gestual serve como uma ferramenta de comunicação vital para as comunidades surdas e com deficiência auditiva, abrangendo uma rica variedade de gestos, expressões faciais e movimentos corporais para transmitir significado. A documentação histórica sugere que formas estruturadas de língua gestual existem há séculos, embora muitas vezes tenham se desenvolvido espontaneamente dentro de comunidades isoladas. Exemplos notáveis incluem as línguas gestuais britânica e francesa, que surgiram organicamente e foram posteriormente refinadas através de instituições educacionais para surdos [17].

No século XVIII, Charles Michel de l'Épée em França fundou a primeira escola pública para surdos, institucionalizando a Língua Gestual Francesa e influenciando outras línguas gestuais, como a ASL [15]. Contudo, o século XX viu um retrocesso devido ao movimento do oralismo, que promoveu a língua falada em detrimento da gestual. A perceção só começou a mudar na segunda metade do século, quando estudos como os de William Stokoe na década de 1960 confirmaram a riqueza e complexidade da ASL, ajudando a restabelecer o uso e estudo das línguas gestuais como verdadeiros sistemas linguísticos [31]. A ASL é uma das línguas gestuais mais estudadas e utilizadas, especialmente nas comunidades surdas dos Estados Unidos e Canadá. O estudo pioneiro de Edward Klima e Ursula Bellugi na década de 1970 foi crucial para a compreensão científica da ASL como uma língua completa e complexa. Eles demonstraram que a ASL possui uma gramática rica e profundamente estruturada, com capacidades equivalentes às das línguas faladas para expressar abstrações e conceitos complexos. Este trabalho ajudou a estabelecer a ASL como um campo legítimo de estudo linguístico e contribuiu significativamente para a mudança de perceções sobre a língua gestual em geral [14].

Estes desenvolvimentos sublinham a evolução dinâmica da língua gestual, refletindo mudanças sociais mais amplas na abordagem à diversidade e inclusão. À medida que estas continuam a ser estudadas e utilizadas, ganham maior reconhecimento, não apenas como ferramentas de comunicação, mas como elementos integrantes da identidade cultural da comunidade surda.

### 2.1.1 Estrutura da Língua

A estrutura das línguas gestuais é complexa e detalhada, apresentando várias camadas de significado e função. Estas línguas utilizam parâmetros como configurações de mão, orientações, movimentos, localizações (relativas ao corpo do utilizador) e expressões faciais para formar os sinais. Cada um destes componentes pode alterar significativamente o significado de um gesto, semelhante à maneira como os sons são usados nas línguas faladas para formar palavras e frases com diferentes significados [26].

A gramática das línguas gestuais é complexa e variada, organizando os sinais de modo a permitir uma comunicação clara e eficiente. Aqui estão alguns aspetos específicos da gramática das línguas gestuais, exemplificados com o uso da ASL:

1. **Concordância Verbal**: As línguas gestuais exibem sistemas de concordância verbal complexos. Os verbos podem concordar com o sujeito e o objeto, indicados pela direção e movimento das mãos que sinalizam. Por exemplo, se o transmissor usa o verbo "dar" e move a mão de si mesmo para outra pessoa, isso indica que a ação de dar é do transmissor para aquela pessoa;

2. **Uso do Espaço**: O espaço de sinalização é usado para representar referências específicos como pessoas, lugares ou coisas. A direção da sinalização indica ações ou relações entre essas referências. Um exemplo é um transmissor pode estabelecer um espaço à sua esquerda para representar um amigo e outro espaço à sua direita para uma loja de conveniência. Ele pode usar o sinal para "ir", direcionado do espaço representando o amigo para o espaço representando a loja, para indicar que o amigo foi à loja;

3. **Classificadores**: São formatos de mão que representam categorias de objetos, ações ou características. Os classificadores são integrados aos verbos para descrever ações detalhadamente. Um classificador comum na ASL é a forma de mão plana (B-handshape) usada para representar veículos ou superfícies planas. Se descrevendo um carro em movimento, o transmissor pode usar este classificador num movimento de um lado para o outro para mostrar o carro a viajar através do seu campo de visão;

4. **Sinais Não-manuais**: Elementos como expressões faciais e posturas corporais são essenciais para expressar funções gramaticais, incluindo perguntas de sim/não, perguntas informativas, negações, entre outros. Na ASL, elevar as sobrancelhas muitas vezes acompanha perguntas de sim/não. Por exemplo, ao sinalizar "VAIS À LOJA?", o transmissor elevaria as sobrancelhas para indicar que se trata de uma pergunta de sim/não, esperando uma resposta de sim ou não;

5. **Aspetos Temporais**: Diferentes aspetos do tempo são expressos através de modulações no verbo ou pelo uso de sinais específicos de tempo no início ou final das frases. Estes aspetos podem ser mostrados por sinais como "ONTEM" ou "AMANHÃ" colocados no início de uma frase para estabelecer o contexto de tempo. Por exemplo,

sinalizar “ONTEM EU FUI À LOJA” em ASL coloca a ação de ir à loja no passado [25].

Na Língua Gestual, os sinais são frequentemente categorizados em três grupos principais baseados na relação entre a forma do sinal e o seu significado: icónicos, referenciais e arbitrários. Cada grupo desempenha um papel fundamental na forma como a informação é comunicada visualmente.

1. **Sinais Icónicos**: Estes sinais têm uma relação visual direta com o objeto ou conceito que representam. Por exemplo, o sinal para “árvore” pode imitar a forma de um tronco com ramos. Estes sinais são frequentemente mais fáceis de aprender e recordar devido à sua natureza intuitiva. Em muitos casos, os sinais icónicos ajudam a tornar a língua gestual acessível e visualmente ligada ao mundo natural.

2. **Sinais Referenciais**: Estes sinais estabelecem uma conexão baseada na associação ou contexto, em vez de uma correspondência visual direta com o objeto ou conceito. Por exemplo, o sinal para “tempo” pode fazer referência ao movimento de um ponteiro de relógio, não porque a forma do sinal seja idêntica ao objeto, mas porque o contexto sugere uma relação interpretativa. Os sinais referenciais dependem de um entendimento partilhado do contexto cultural ou situacional.

3. **Sinais Arbitrários**: Estes sinais não têm uma relação visual óbvia com o seu significado. Eles precisam ser aprendidos como qualquer palavra numa língua falada porque a sua forma não revela diretamente o seu significado. Por exemplo, muitos sinais para conceitos abstratos caem nesta categoria, como o sinal para “liberdade” [20].

### 2.1.2 Língua Gestual em Portugal

A LGP é umas das três línguas oficiais de Portugal, foi reconhecida enquanto língua da comunidade surda portuguesa pela Constituição da República 1997, especificamente no artigo 74, n.º 2, alínea h [27], que determina como tarefa do Estado “proteger e valorizar a LGP, enquanto expressão cultural e instrumento de acesso à educação e da igualdade de oportunidades.” [21].

A LGP tem raízes que remontam ao século XIX, inspirando-se na experiência de Per Aron Borg que trabalhava com surdos na Suécia. Convidado pelo Rei D. João VI, Borg visitou Portugal para fornecer assessoria técnica nesta área, contribuindo para a fundação, em 1823, da primeira escola para surdos em Portugal. Embora os vocabulários da LGP e da Língua Gestual Sueca sejam distintos, o alfabeto manual das duas línguas reflete a origem comum devido à influência de Borg no ensino e na formação dos surdos [9].

Tal como outras línguas gestuais, a LGP possui um sistema de dactilologia, que é o alfabeto manual utilizado para apoiar a comunicação de conceitos que não têm sinais específicos. Em relação à ordem das palavras nas frases, esta segue uma estrutura diferente da língua

portuguesa. Não há consenso sobre a ordem predominante: alguns linguistas defendem uma ordem "sujeito-objeto-verbo" (SOV), enquanto outros argumentam que a ordem é "objeto-sujeito-verbo" (OSV) [2]. Nas perguntas, a LGP recorre a expressões faciais e pronomes interrogativos que normalmente aparecem no final da frase. Para negar, os sinais negativos são expressos de várias formas: utilizando o gesto NÃO, alterando a postura corporal (como movimentos de cabeça) ou executando verbos na forma negativa, como "NÃO QUERER" [18].

A comunidade surda utiliza gestos que simbolizam os conceitos representados. Parâmetros como a expressão facial, os movimentos corporais e os gestos criam uma imagem clara da ideia transmitida. No entanto, existem termos raros ou muito específicos que não estão no vocabulário e precisam ser soletrados. O alfabeto é parte essencial da LGP, composto por símbolos feitos com a mão dominante (direita ou esquerda). A maioria dos sinais é estática e representa letras, exceto "d", "k" e "z", que possuem movimento. A datilologia (soletrar palavras) é usada quando um símbolo não existe ou não é conhecido, como em nomes próprios. Para esta dissertação, assume-se que nenhuma das letras tem movimento, assumindo todos os gestos representados na Figura 2.1. Em particular, as letras D", K" e "Z" são tratadas como gestos estáticos, com base nas formas apresentadas na figura mencionada, para a sua tradução.


Figura 2.1: *Abecedário em Língua Gestual Portuguesa [7]*

A LGP, como já referido é uma língua visual rica e complexa, cuja aprendizagem e utilização envolvem não apenas o conhecimento de sinais manuais, mas também uma ampla gama de expressões faciais e movimentos corporais que são fundamentais para a comunicação eficaz. Cada gesto ou sinal pode conter múltiplas camadas de significado, dependendo do contexto em que é usado, da expressão facial que o acompanha, e da sequência de sinais que o precede ou segue.

## 2.2 Reconhecimento Automático de Letras da LGP

Com o avanço das tecnologias de visão computacional e inteligência artificial, surge a possibilidade de automatizar o reconhecimento de gestos da LGP, facilitando a comunicação entre surdos e não-surdos. No entanto, o reconhecimento automático de letras da LGP é um problema complexo de padrões e classificação, dada a variabilidade dos gestos, a ambiguidade entre alguns sinais e as diferentes condições de captura. Apesar disso, gestos com movimento, como os presentes em algumas letras, não representam um desafio adicional significativo neste trabalho, já que foram tratados como formas estáticas.

Este problema pode ser formulado como uma tarefa de classificação em que o sistema deve identificar, a partir de uma imagem ou vídeo de um gesto manual, a letra correspondente. Isto requer a construção de um classificador capaz de generalizar bem, ou seja, reconhecer com precisão gestos que variam em detalhes, como o ângulo das mãos ou o posicionamento dos dedos.

## 2.3 Aprendizagem Automática e Classificação de Padrões

A aprendizagem automática (*machine learning*) tem transformado multiplas áreas, permitindo que sistemas de inteligência artificial aprendam a partir de dados e façam previsões com base em padrões complexos. No caso da LGP, o reconhecimento automático de letras gestuais representa um problema típico de classificação de padrões, onde a tarefa do sistema é atribuir uma classe correta a um gesto específico, ou seja, identificar a letra que está a ser representada.

Para resolver este problema, a aprendizagem automática permite a utilização de algoritmos que ajustam os seus parâmetros com base em exemplos, aprendendo a reconhecer gestos visuais complexos. Esta abordagem é particularmente importante em contextos de visão computacional, onde é necessário processar imagens para identificar padrões subtis, como a configuração dos dedos numa determinada posição.

O uso de técnicas como *deep learning*, *transfer learning* e *data augmentation* tem facilitado avanços significativos neste campo. Estes conceitos não só ajudam a melhorar a precisão dos modelos como também permitem a generalização, que é a capacidade do modelo de realizar boas previsões em dados que não foram vistos durante o treino.

### 2.3.1 Modelos e Classificadores em Aprendizagem Automática

Na tarefa de reconhecimento de padrões visuais, como as letras da LGP, a escolha do classificador é um dos elementos mais importantes para garantir um bom desempenho. Existem vários classificadores que podem ser aplicados, cada um com vantagens e limitações próprias, dependendo da complexidade dos dados e do problema.

Um dos modelos mais utilizados para problemas de classificação de imagem é a Rede Neuronal Convolucional (CNN) (*Convolutional Neural Network*), amplamente adotada devido à sua capacidade de aprender automaticamente padrões visuais complexos a partir de dados. As CNNs são redes neuronais compostas por camadas convolucionais, que aplicam

filtros para extrair características específicas das imagens, como bordas, texturas e formas, e por camadas de *pooling*, que reduzem a dimensionalidade dessas características, mantendo as informações mais relevantes [6]. Como ilustra a Figura 2.2, esta estrutura hierárquica permite que a CNN analise progressivamente os dados em diferentes níveis de abstração.

No contexto da LGP, a capacidade das CNNs de capturar padrões visuais é crucial para identificar detalhes subtis nos gestos manuais, como a orientação dos dedos ou pequenas variações na posição da mão. Estes detalhes são essenciais para distinguir letras que podem parecer semelhantes à primeira vista, como 'V' e 'X' ou 'D' e 'K'. A eficácia da CNN em capturar esses detalhes pode ser avaliada através da análise das predições corretas e incorretas realizadas pelo modelo, verificando quais as características foram usadas para separar as classes de forma eficaz. Adicionalmente, métodos como visualizações de mapas de ativação (*activation maps*) podem demonstrar quais áreas da imagem influenciam mais as decisões do modelo, provando a sua capacidade de reconhecer características visuais específicas.

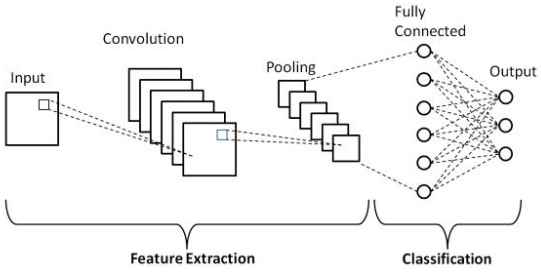


Figura 2.2: *Exemplo de Arquitetura de uma CNN [4].*

Outro algoritmo de classificação comum em problemas de visão computacional é o MLP, uma rede neuronal composta por camadas densas totalmente conectadas. A MLP é frequentemente utilizado em conjunto com as CNNs, onde as primeiras camadas convolucionais extraem características visuais e a MLP realiza a classificação final. Esta combinação cria uma abordagem híbrida robusta, especialmente eficaz em múltiplos problemas, em especial neste caso onde a variação nos gestos pode ser grande, como ilustrado na Figura 2.3.

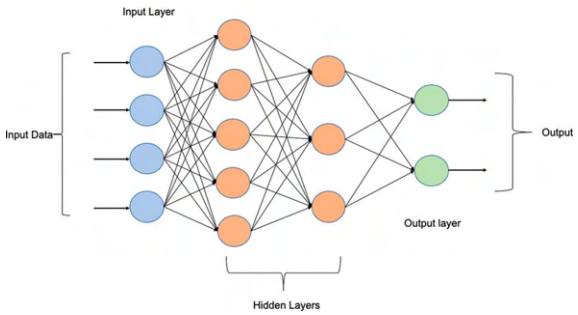


Figura 2.3: *Exemplo de Arquitetura de uma MLP [8].*

Algoritmos mais simples, como o *k-Nearest Neighbors* (kNN), também são utilizados em problemas de classificação, embora tenham limitações consideráveis em problemas complexos como o reconhecimento de gestos. O kNN classifica um ponto de dados com base

nos seus vizinhos mais próximos, como ilustra a Figura 2.4. No entanto, o seu desempenho tende a degradar-se à medida que a quantidade de dados aumenta, uma vez que é necessário calcular a distância para todos os exemplos de treino a cada predição.

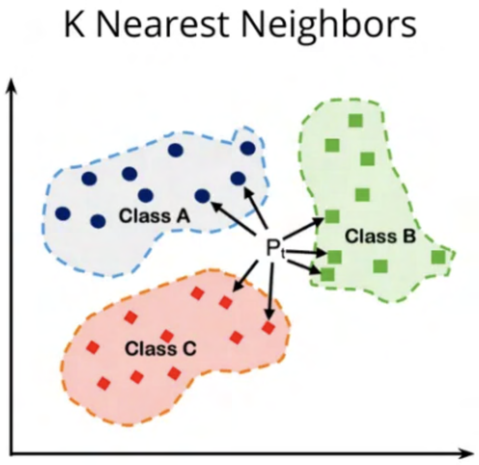


Figura 2.4: *Representação do k-Nearest Neighbors.*

Outro classificador relevante é o *Support Vector Machine* (SVM), que tenta encontrar um hiperplano que separe de forma clara as classes no espaço de características. O SVM é eficaz em problemas de alta dimensionalidade, mas enfrenta dificuldades quando as classes não são linearmente separáveis, como no caso dos gestos manuais com variações subtis. A Figura 2.5 mostra um exemplo de separação de classes pelo SVM [5].

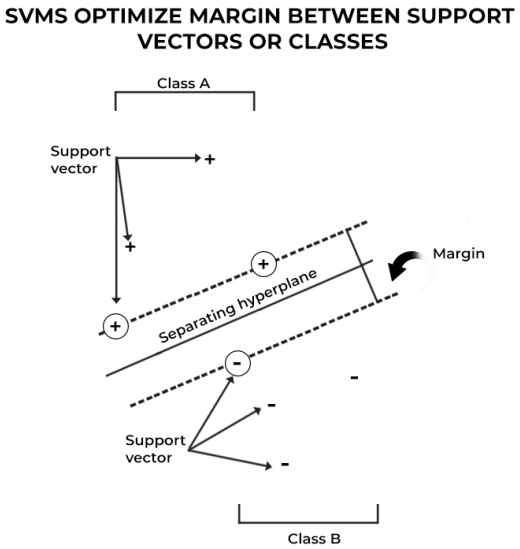


Figura 2.5: *Exemplo de separação de classes pelo hiperplano no SVM [1].*

Além disso, as Árvores de Decisão são um método frequentemente utilizado, onde as decisões são tomadas de forma hierárquica, dividindo os dados com base em condições simples. No entanto, no reconhecimento de gestos, este método pode ter dificuldades em capturar variações visuais complexas, como ilustrado na Figura 2.6 . O *Random Forest*, uma variante mais robusta das árvores de decisão, constrói múltiplas árvores em paralelo

e toma decisões com base no voto da maioria [19].

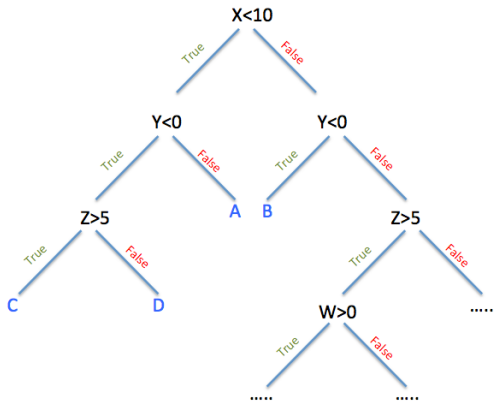


Figura 2.6: *Exemplo de uma Árvore de Decisão.*

### 2.3.2 Conceitos em Aprendizagem Automática

Agora que os modelos e classificadores foram apresentados, é importante explorar os conceitos que complementam e aprimoram o desempenho desses modelos em tarefas específicas, como o reconhecimento de gestos.
O primeiro conceito a destacar é o *Deep Learning*. Esta subárea da aprendizagem automática baseia-se no uso de redes neuronais profundas, que são compostas por múltiplas camadas de neurónios artificiais interligados. Cada camada numa rede profunda processa os dados de entrada de maneira hierárquica, extraindo informações progressivamente mais complexas. O *Deep Learning* é amplamente utilizado em várias tarefas, por exemplo de visão computacional devido à sua capacidade de capturar padrões visuais e abstrair características úteis sem a necessidade de intervenção manual.

#### 2.3.2.1 Transfer Learning

O *Transfer Learning* é uma técnica que aproveita o conhecimento adquirido por um modelo treinado numa tarefa para resolver outra tarefa relacionada. Normalmente, modelos de redes profundas são treinados em grandes bases de dados, como o *ImageNet* [24], que contém milhões de imagens. Este conhecimento aprendido pode ser transferido para tarefas específicas, como o reconhecimento de letras gestuais, ajustando apenas uma parte do modelo pré-treinado, enquanto outras partes mantêm os seus parâmetros inalterados face ao modelo previamente treinado [37].

Ao utilizar o *Transfer Learning*, é possível acelerar o processo de treino e alcançar melhores resultados, mesmo com poucos dados específicos da tarefa, como a LGP. A técnica reduz o custo computacional e melhora a precisão ao reutilizar pesos e camadas que já capturam características visuais úteis.

#### 2.3.2.2 Data Augmentation

O *Data Augmentation* é uma técnica utilizada para aumentar o volume de dados de treino através da criação de novas amostras a partir das existentes. No reconhecimento de gestos, este processo pode envolver transformações simples, como rotações, redimensionamentos, espelhamento e alterações de brilho das imagens capturadas.

O objetivo do *Data Augmentation* é aumentar a capacidade de generalização do modelo, garantindo que ele funcione bem em condições variadas de captura, como diferentes ângulos, iluminação ou gestos ligeiramente diferentes. Esta técnica é especialmente útil quando há uma quantidade limitada de dados disponíveis, como é frequentemente o caso na LGP.

#### 2.3.2.3 Otimizadores de Aprendizagem

Os otimizadores são algoritmos que ajustam os pesos de uma rede neuronal durante o processo de treino, minimizando a função de custo. Entre os mais utilizados estão o *Stochastic Gradient Descent* (SGD), o *Adam* e o *RMSprop* [3]. O *Adam*, por exemplo, é amplamente utilizado em redes profundas por ser eficiente e adaptar dinamicamente as taxas de aprendizagem, tornando-o ideal para a otimização de grandes redes.

#### 2.3.2.4 Pesos em Redes Neurais

Os pesos (*weights*) são um componente fundamental das redes neuronais. Eles representam a força das conexões entre os neurónios em diferentes camadas da rede e determinam como os dados de entrada são transformados ao longo do modelo. Durante o processo de treino, os pesos são ajustados iterativamente pelo algoritmo de otimização para minimizar a função de custo, permitindo que a rede aprenda padrões nos dados de treino.

O ajuste dos pesos é realizado através do *backpropagation*, que calcula os gradientes dos erros em relação aos pesos e utiliza esses gradientes para atualizar os valores. Este processo permite que o modelo melhore as suas predições ao longo do tempo. A importância dos pesos é evidente em tarefas complexas, como o reconhecimento de padrões visuais, onde pequenos ajustes podem ter um impacto significativo na precisão final da rede [10].

Além disso, em cenários de classes desbalanceadas, onde algumas classes possuem menos exemplos do que outras, é comum aplicar pesos adicionais às classes com menor representação para garantir que a função de custo penalize mais os erros nessas classes. Esta técnica, conhecida como ajuste de pesos de classes, ajuda a rede a aprender de forma equilibrada, reduzindo o viés para classes com mais exemplos.

### 2.3.3 Métricas de Avaliação

Para avaliar a eficácia de um modelo de aprendizagem automática, várias métricas de desempenho são usadas, permitindo medir o quão bem o modelo está a generalizar a partir dos dados de treino e a prever novos exemplos.

A métrica mais comum é a exatidão (*accuracy*), que mede a proporção de predições corretas em relação ao total de predições feitas pelo modelo. Embora útil, a precisão pode

ser enganadora em problemas com classes desequilibradas, onde algumas classes podem ter muito mais exemplos que outras.

A *classificação binária* é outra métrica importante, que mede a capacidade do modelo de identificar corretamente todas as instâncias de uma classe específica. Um *recall* ( elevado indica que o modelo consegue detetar a maioria dos exemplos positivos de uma classe, o que é essencial em cenários onde se pretende minimizar os falsos negativos.

O *F1-Score* combina precisão (precision) e *recall* numa única métrica, sendo a média harmónica entre ambos. Esta métrica é particularmente útil quando há um equilíbrio entre evitar falsos positivos e falsos negativos.

Além destas métricas, a matriz de confusão é uma ferramenta visual amplamente utilizada para avaliar a performance dos classificadores. Esta matriz apresenta uma tabela que compara as predições feitas pelo modelo com os rótulos reais, permitindo identificar quais as classes que são mais frequentemente confundidas.

## 2.4 Trabalhos Relacionados

O reconhecimento de gestos, especialmente no contexto de línguas gestuais, evoluiu consideravelmente ao longo das últimas décadas. No entanto, o foco específico no reconhecimento de letras da LGP tem sido mais limitado, com muitos trabalhos concentrando-se no reconhecimento de gestos completos ou frases. Inicialmente, muitos investigadores recorreram a métodos baseados em hardware especializado, como luvas sensoriais. Por exemplo, no trabalho de Neiva [22], foram utilizadas luvas equipadas com sensores de flexão para capturar os gestos da LGP, com o objetivo de reconhecer tanto gestos estáticos como as letras do alfabeto manual. As luvas mediam a curvatura de cada dedo, permitindo uma alta precisão na deteção dos gestos, facilitando o reconhecimento das letras da LGP. No entanto, apesar da sua precisão, estas soluções enfrentavam desafios em termos de usabilidade e custo, limitando a sua aplicabilidade para o uso diário em contextos mais informais.

Com a evolução tecnológica, os investigadores começaram a desenvolver sistemas que pudessem capturar gestos sem a necessidade de dispositivos físicos intrusivos. O trabalho de Patrícia Reis Ribeiro [28] utilizou o Microsoft Kinect, que capturava gestos tridimensionais através de sensores de profundidade. Este sistema era eficaz para captar movimentos amplos que envolvem não apenas as mãos, mas também outras partes do corpo, facilitando a tradução de frases completas. Contudo, a sua aplicação no reconhecimento de letras da LGP era limitada pela incapacidade de detetar com precisão a configuração dos dedos, essencial para o reconhecimento de letras, que dependem de variações subtis nos gestos das mãos. Além disso, o Kinect é um dispositivo fixo, o que limita a sua portabilidade e impede o seu uso em ambientes dinâmicos.

Posteriormente, surgiram abordagens baseadas em algoritmos de *machine learning*, que eliminaram a necessidade de hardware especializado e começaram a utilizar câmaras comuns. O trabalho de Oliveira [23] exemplifica esta evolução, combinando uma CNN com o Kinect. Embora o Kinect fosse eficaz na captura de movimentos, era a CNN que processava as imagens capturadas e realizava o reconhecimento dos gestos. Esta abordagem mostrou-se

promissora para gestos dinâmicos, mas continuou a enfrentar dificuldades na captura de detalhes subtis, como a posição exata dos dedos, o que é crucial para o reconhecimento de letras da LGP.

O uso de algoritmos mais simples, como o k-Nearest Neighbors (kNN), também foi explorado para o reconhecimento de gestos. Singh [30] utilizou o kNN para permitir que a assistente virtual Alexa reconhecesse comandos gestuais em ASL. Embora esta abordagem fosse eficaz para pequenos conjuntos de dados, a sua escalabilidade era limitada à medida que a complexidade dos gestos aumentava, tornando o kNN menos eficiente em tarefas mais complexas, como o reconhecimento do alfabeto da LGP.

Com a chegada das CNNs, o reconhecimento de gestos deu um grande salto em termos de precisão, permitindo identificar padrões visuais complexos, e em termos de rapidez no processamento das predições, tornando-o mais adequado para aplicações em tempo real. Tilottama Goswami [12] utilizou uma CNN para reconhecer gestos da Língua Gestual Americana (ASL), especificamente o alfabeto manual, com base no conjunto de dados Sign Language MNIST. Embora o sistema tenha alcançado uma precisão elevada para gestos estáticos e simples, o conjunto de dados utilizado era limitado em variação, o que o tornava inadequado para lidar com a complexidade e variações subtis dos gestos da LGP.

Uma outra inovação neste campo foi a introdução do MediaPipe, desenvolvido pela Google, que permite capturar pontos-chave (landmarks) das mãos de forma eficiente e acessível, sem a necessidade de hardware especializado [16]. Utilizado em conjunto com redes neuronais como MLPs, o MediaPipe tornou-se uma ferramenta popular para o reconhecimento de gestos em tempo real, utilizando apenas câmaras comuns. Kazuhito00 [33] utilizou o MediaPipe para capturar pontos das mãos e classificá-los com uma MLP, demonstrando a sua eficácia no reconhecimento de gestos manuais detalhados.

O *MediaPipe Hands* é um módulo dedicado à deteção e seguimento de mãos, utilizando um modelo de deteção de palmas para localizar rapidamente a mão e um modelo de regressão 3D para identificar com precisão 21 pontos-chave. Esta modularidade torna o MediaPipe adequado para ser integrado em sistemas de reconhecimento de gestos em dispositivos móveis, mostrando-se eficiente mesmo em condições desafiadoras [36]. A Figura 2.7 ilustra as *landmarks* detetadas pelo MediaPipe, bem como as distâncias e ângulos calculados entre pares de pontos num exemplo de um gesto da letra Q, que podem ser utilizados para melhorar a precisão de sistemas baseados em MLP. Outros exemplos podem ser observados na Figura III.1 nos Anexos.

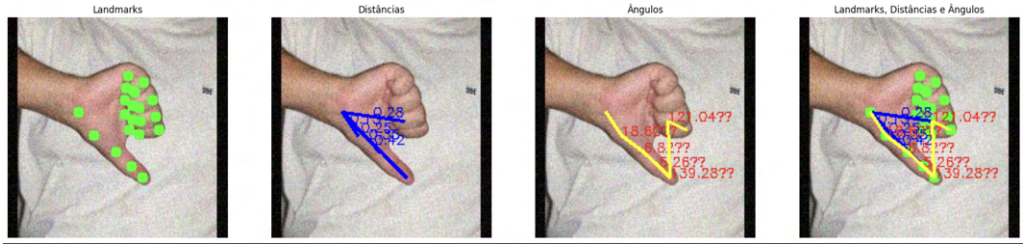


Figura 2.7: *Exemplo de landmarks, distâncias e ângulos calculados.*

As *landmarks* representam pontos de referência específicos na mão, como as pontas dos dedos, articulações e a base da palma, posicionados num espaço tridimensional (3D) para capturar a postura e o movimento da mão em diferentes gestos. As distâncias entre *landmarks* permitem medir a separação entre pontos, como entre o polegar e o indicador ou entre articulações dos dedos, sendo úteis para identificar o grau de abertura da mão ou a proximidade entre os dedos. Já os ângulos são calculados entre três *landmarks* consecutivas, indicando o grau de flexão ou extensão de um dedo em relação à posição da mão. Com isso, contribuem para distinguir com precisão gestos específicos, aprimorando a robustez do sistema de reconhecimento de gestos.

Em suma, a evolução do reconhecimento de gestos, desde o uso de hardware especializado até a utilização de câmaras comuns e algoritmos avançados de *machine learning*, resultou em soluções mais acessíveis e práticas para o reconhecimento de letras da LGP. As CNNs e o MediaPipe oferecem uma abordagem mais flexível e eficaz para a tradução da LGP em dispositivos móveis, permitindo uma implementação escalável e eficiente em tempo real.

# 3 Metodologia

A metodologia adotada neste trabalho foi desenhada com o objetivo principal de desenvolver um sistema capaz de reconhecer as letras da LGP em tempo real, utilizando a câmara de um dispositivo móvel. Este capítulo descreve de forma geral as abordagens tecnológicas e metodológicas que orientaram o desenvolvimento do sistema, sem entrar em detalhes específicos, que serão discutidos em maior profundidade na secção de implementação.

Desde o início do projeto, foi essencial garantir que o sistema tivesse a capacidade de realizar o reconhecimento de letras com rapidez no processamento e alta confiabilidade, considerando desafios como variações de iluminação, ângulos de captura e diferentes dispositivos móveis. Assim, a metodologia focou-se em encontrar soluções robustas, escaláveis e adaptáveis a diferentes condições de utilização, sempre com o objetivo de proporcionar uma experiência de utilizador fluida e acessível.

A arquitetura geral do sistema foi dividida em quatro componentes principais: a aquisição e pré-processamento de dados, o desenvolvimento dos modelos de aprendizagem automática e a criação de uma aplicação móvel intuitiva. Cada um destes componentes foi abordado de forma a garantir um equilíbrio adequado entre desempenho e praticidade.

## 3.1 Fundamentos Tecnológicos e Abordagens de Desenvolvimento

A criação de um sistema de reconhecimento de gestos envolve a integração de várias tecnologias e abordagens que lidam com a captura de dados, o processamento de informações em tempo real, e a interface com o utilizador. O desenvolvimento da aplicação móvel exigiu a escolha de ferramentas que permitissem o desenvolvimento multiplataforma e a integração eficiente de modelos de aprendizagem automática.

O sistema foi desenhado para funcionar tanto com processamento local como remoto, de forma a garantir flexibilidade em diferentes cenários. Para o processamento local, optou-se por utilizar o modelo CNN, pois era possível executá-lo diretamente nos dispositivos móveis graças à sua conversão para *TensorFlow Lite*. Este formato otimizado permite que o modelo funcione de forma eficiente em dispositivos com recursos limitados. Por outro lado, o MediaPipe foi configurado para processamento remoto, uma vez que, atualmente,

não possui suporte nativo para execução local no contexto de visão computacional, como a deteção de pontos-chave (*landmarks*) das mãos [11]. Esta limitação tecnológica determinou a necessidade de integrar o MediaPipe como uma solução dependente de conexão à internet, sendo utilizado em um servidor remoto. A decisão de maximizar a capacidade de processamento de inferências diretamente no dispositivo, utilizando o modelo CNN, foi motivada por diversas razões práticas e técnicas. O modelo local oferece a vantagem de operar em modo *offline*, eliminando a dependência de conectividade com a internet. Isso é especialmente importante em ambientes onde a conectividade pode ser limitada ou inexistente, garantindo que a aplicação continue funcional e proporcionando uma experiência de utilizador mais fluida, com tempos de resposta reduzidos. Além disso, o processamento local reduz a necessidade de transferir imagens para um servidor remoto, aumentando a privacidade dos dados dos utilizadores.

## 3.2 Aquisição e Pré-processamento de Dados

Para garantir a qualidade (imagens definidas e consistentes, e a representatividade dos dados utilizados no treino dos modelos de reconhecimento de letras da LGP, será seguida uma metodologia baseada na recolha de dados novos.Por qualidade entende-se a obtenção de dados claros e precisos, com imagens bem definidas e consistentes em termos de iluminação, resolução e ângulos de captura, evitando ruído ou distorções que possam comprometer o desempenho do modelo. Representatividade refere-se à inclusão de exemplos que cubram adequadamente as variações naturais dos gestos, como diferenças na execução por diferentes utilizadores, tamanhos de mãos, posições e condições de captura. Este equilíbrio entre qualidade e representatividade é essencial para garantir que o modelo seja robusto e capaz de generalizar para situações reais de utilização.

Dado que não existem *datasets* públicos amplamente disponíveis e adequados para esta tarefa, será necessário criar um *dataset* específico, recolhendo imagens e vídeos de pessoas a realizarem gestos correspondentes às letras da LGP.

Além da recolha de dados, será aplicado um processo de pré-processamento para garantir a qualidade dos dados utilizados no treino dos modelos. Esta fase incluirá técnicas de normalização e aumento de dados (*data augmentation*), que permitirão expandir o conjunto de dados e aumentar a sua diversidade. Estas metodologias são amplamente utilizadas em problemas de visão computacional para melhorar a capacidade de generalização dos modelos, garantindo que consigam lidar com variações nos gestos, condições de iluminação e ângulos de captura.

## 3.3 Modelos de Classificação

A escolha de modelos de aprendizagem automática equilibrou eficiência e precisão, considerando as capacidades dos dispositivos e os requisitos de desempenho. O CNN, com baixa complexidade computacional, foi escolhido para processamento local, permitindo reconhecimento rápido em dispositivos móveis. Já o MediaPipe, focado na precisão de

gestos complexos, foi destinado ao processamento remoto, onde maior capacidade computacional está disponível. O sistema de classificação de gestos foi desenhado para identificar corretamente as letras da LGP, utilizando abordagens que garantissem a adequação tanto para execução local em dispositivos móveis como para processamento remoto.

A arquitetura de modelos necessita de métodos que lidam bem com a complexidade de dados visuais e gestos manuais, assegurando um tempo de resposta em tempo real, ao mesmo tempo que mantêm a precisão necessária para reconhecimento eficaz.

## 3.4 Avaliação do Sistema

A avaliação do sistema será baseada em várias métricas que permitirão analisar o desempenho dos modelos de reconhecimento de letras. Métricas como a precisão (*accuracy*), *recall*, *F1-Score* e a matriz de confusão foram identificadas como ferramentas essenciais para medir a capacidade do sistema de classificar corretamente os gestos e identificar áreas de melhoria.

O uso dessas métricas garante uma análise completa do comportamento do sistema e a identificação de possíveis ajustes que poderão ser feitos para otimizar o desempenho e a precisão na identificação de gestos.

## 3.5 Avaliação da Usabilidade do Sistema

Para isso, foi planeada a recolha de *feedback* através de entrevistas com utilizadores reais, com o objetivo de obter informações relevantes sobre a clareza e a facilidade de utilização da aplicação, identificando pontos fortes e áreas a melhorar."

O foco estará em garantir que a aplicação seja intuitiva e simples de usar, tendo em conta que a acessibilidade e a facilidade de interação são aspetos fundamentais para o sucesso de um sistema destinado à tradução da LGP.

# 4 Aquisição e Pré-processamento de Dados

A criação de um sistema de reconhecimento de letras do alfabeto da LGP exige uma base sólida de dados que represente fielmente os gestos e as suas variações. Como não existia um *dataset* público adequado para a LGP, foi necessário construir um *dataset* personalizado, obtendo dados de várias fontes e submetendo-os a um processo de pré-processamento. Este capítulo detalha as etapas de aquisição e preparação dos dados, fundamentais para o desenvolvimento dos modelos de aprendizagem automática utilizados no projeto.

A Figura 4.1 ilustra o fluxo completo do processo de aquisição e pré-processamento de dados, desde a captura das imagens até à criação de variantes através de técnicas de *data augmentation*. O processo descrito na figura será explorado nas secções seguintes, onde cada etapa será detalhada para garantir a construção de um conjunto de dados robusto, essencial para o desempenho dos modelos de aprendizagem automática.

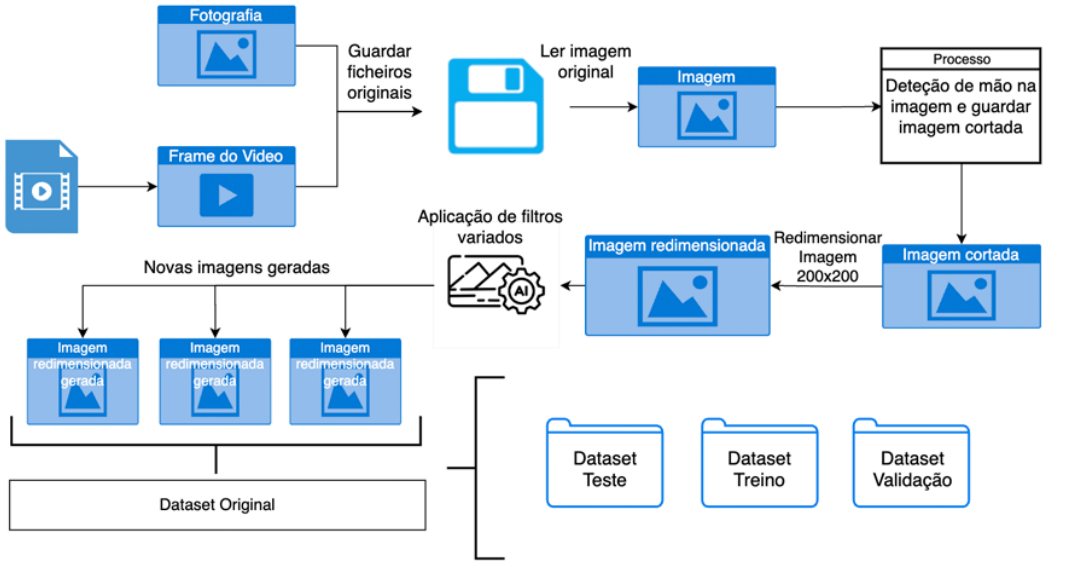


Figura 4.1: *Fluxo de aquisição e pré-processamento de dados para o reconhecimento de letras da LGP.*

## 4.1 Recolha e Preparação de Dados

A recolha de dados para o sistema de reconhecimento da LGP envolveu capturar imagens e vídeos de vários amigos e colegas a realizar os gestos das letras da LGP em diferentes condições. A diversidade dos dados foi uma preocupação constante, garantindo que os modelos de aprendizagem automática pudessem generalizar bem para diferentes contextos e condições de captura.

### 4.1.1 Fotografias Capturadas em Diferentes Contextos

Uma das principais fontes de dados foi a captura de imagens de várias pessoas a realizar os gestos das letras da LGP. Para garantir a representatividade, as imagens foram tiradas em ambientes variados, com diferentes fundos, condições de iluminação e ângulos. Essa diversidade reflete as condições reais em que o sistema poderá ser utilizado e ajuda a aumentar a robustez do modelo ao lidar com diferentes variações de gestos. A Figura 4.2 mostra três exemplos de imagens do *dataset*, representando a letra "A" em diferentes ambientes.


Figura 4.2: *Imagens da letra A de LGP capturadas em diversos ambientes*

Além de capturar imagens individuais, foi necessário explorar outras abordagens para garantir que o *dataset* fosse suficientemente diverso e robusto para o treino dos modelos, como a extração de frames a partir de vídeos a fazer as letras.

### 4.1.2 Extração de *Frames* a Partir de Vídeos

Outra importante fonte de dados foi a gravação de vídeos onde as letras da LGP foram realizadas continuamente. A vantagem de utilizar vídeos reside na possibilidade de capturar múltiplos *frames* que contêm variações subtis dos gestos, aumentando assim o número de exemplos disponíveis para treino. Após a gravação dos vídeos, foram extraídos os *frames* mais relevantes, correspondentes a cada letra. Para isolar a mão do resto do corpo e garantir que o modelo se concentrasse apenas no gesto, foi utilizado o *MediaPipe*, uma *framework* de visão computacional da Google, que detetou a região da mão em cada *frame*. A Figura 4.3 apresenta alguns *frames* obtidos de um vídeo gravado, representando a letra "A".

É importante notar que, ao extrair imagem a imagem de vídeos, o conjunto de dados gerado torna-se bastante grande, contudo este tinha um número significativo de imagens muito semelhantes entre si. Embora essas imagens similares possam parecer redundantes, elas ajudam a capturar pequenas variações nos gestos, o que pode melhorar a capacidade

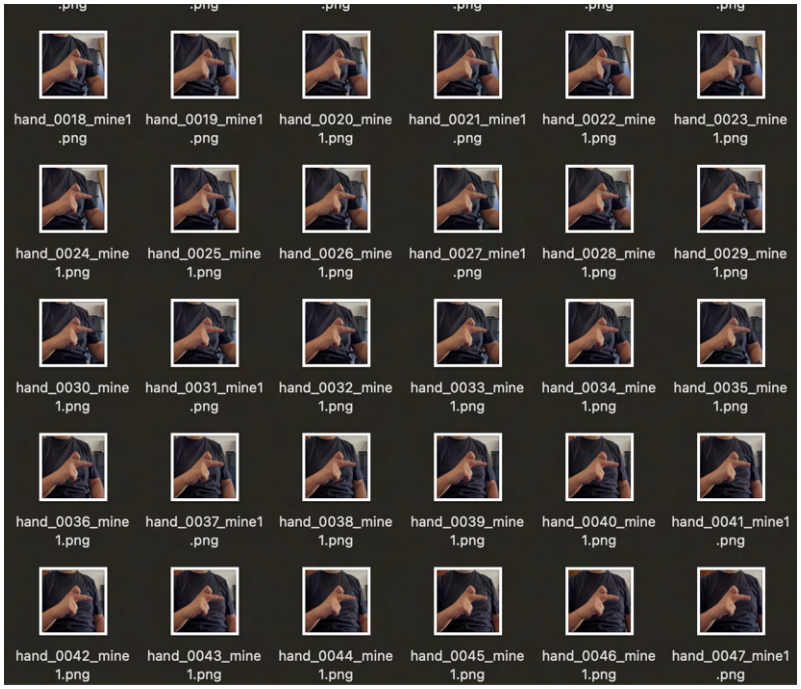


Figura 4.3: *Imagens da letra A de LGP obtidas através dos frames de um vídeo*

do modelo de reconhecer gestos em diferentes condições. No entanto, a repetição de *frames* também aumenta a necessidade de estratégias como *data augmentation* para diversificar ainda mais o conjunto de treino e evitar o sobre-ajuste do modelo a esses padrões muito específicos.

### 4.1.3 Inclusão da Classe "Nothing"

Além das imagens que representam as letras da LGP, foi necessário incluir uma classe adicional para indicar quando nenhuma letra está sendo gesticulada, conhecida como "classe nothing". Esta classe é crucial para que o sistema seja capaz de diferenciar entre quando um gesto válido está sendo realizado e quando não há nenhum gesto presente. As imagens para esta classe foram obtidas de um *dataset* disponível no Kaggle [32], onde não havia qualquer tipo de gesto presente. A inclusão dessa classe ajuda o modelo a evitar falsos positivos, melhorando a precisão das predições.

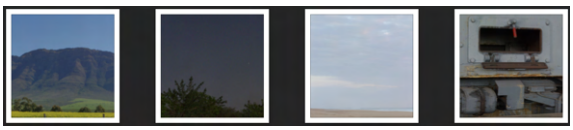

Figura 4.4: *Imagens representativa da classe* ***nothing***

## 4.2 Pré-processamento dos Dados

Após a recolha dos dados, estes foram submetidos a um processo de pré-processamento para garantir que o modelo fosse treinado com amostras bem capturadas, com iluminação adequada, foco claro e gestos visíveis, eliminando informações irrelevantes ou inconsistentes. Além disso, foi assegurada a uniformidade no formato, tamanho e orientação das imagens,

permitindo que o modelo recebesse dados consistentes. Este pré-processamento desempenha um papel crucial na melhoria da capacidade do modelo de aprender padrões relevantes e generalizar para novos dados.

### 4.2.1 Normalização das Imagens

Uma das primeiras etapas do pré-processamento foi a normalização das imagens. Todas as imagens capturadas e extraídas dos vídeos foram redimensionadas para o tamanho padrão de 128x128 pixeis. Este tamanho foi escolhido por representar um equilíbrio: não são imagens muito grandes, o que reduziria a eficiência do processamento e exigiria mais recursos computacionais, nem muito pequenas, o que poderia comprometer a preservação de detalhes importantes dos gestos necessários para o reconhecimento das letras. A normalização do tamanho das imagens reduz a variabilidade que não está relacionada com os gestos em si, permitindo que o modelo se concentre nas características relevantes das letras. Em alguns casos, durante o redimensionamento, foram adicionadas barras pretas na parte superior e inferior das imagens para evitar distorções, que ocorreriam se as proporções originais fossem alteradas. A Figura 4.5 mostra o processo de normalização das imagens, exemplificado com a letra "A".

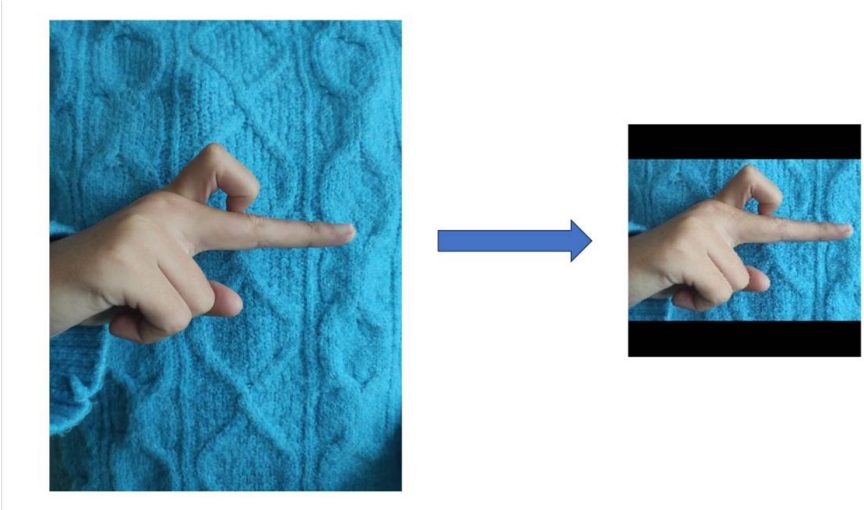

Figura 4.5: *Criação de novas imagens a partir de uma original - Letra A*

### 4.2.2 Aumento de Dados (Data Augmentation)

Para aumentar a robustez do modelo e melhorar a sua capacidade de generalização, foi aplicado um conjunto de técnicas de *data augmentation*. O *data augmentation* é uma técnica que permite aumentar o volume de dados de treino, gerando novas amostras a partir das existentes. No caso das imagens das letras da LGP, foram aplicadas transformações como rotação, *flip* horizontal que simula as letras feitas por pessoas esquerdinas, ajuste de brilho e contraste aplicando variações aleatórias dentro de intervalos predefinidos, simulando condições reais de captura sob diferentes iluminações, e pequenas translações. Estas técnicas permitem que o modelo aprenda a lidar com variações comuns nas condições de captura, incluindo diferentes ângulos de visão, iluminação excessiva, insuficiente ou

desigual, alterações na orientação das mãos, e pequenas deslocações ou inclinações dos gestos. Este processo assegura que o modelo seja mais robusto e capaz de generalizar melhor para cenários reais, onde essas variações são frequentemente inevitáveis. A Figura 4.6 apresenta exemplos de novas imagens geradas a partir de uma imagem original da letra "A", utilizando técnicas de *data augmentation*.

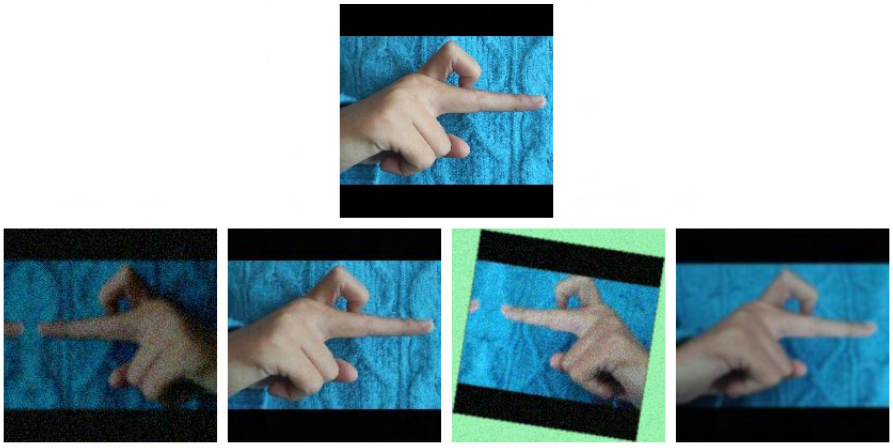

Figura 4.6: *Criação de novas imagens a partir de uma original - Letra A*

Essas técnicas são essenciais para aumentar a diversidade dos dados e permitir que o modelo seja robusto o suficiente para realizar predições precisas em diferentes condições.

### 4.2.3 Organização do *Dataset*

Após o processo de pré-processamento, todas as imagens foram organizadas em pastas correspondentes a cada uma das 27 classes do *dataset*, sendo 26 classes referentes às letras da LGP e uma classe adicional, denominada "nothing", que indica a ausência de gestos. Essa organização foi fundamental para o treino eficiente dos modelos, permitindo que os dados fossem alimentados de forma estruturada durante o processo de aprendizagem. A organização clara e eficiente dos dados garante que o processo de treino ocorra de maneira fluida e sem erros.

Além disso, para evitar o *data leakage*, um fenómeno em que informação presente nos dados de treino também aparece nos dados de validação ou teste, comprometendo a avaliação da capacidade de generalização do modelo, foi realizada uma separação rigorosa dos dados em três conjuntos: treino, validação e teste. Esse processo foi feito manualmente, especialmente para as imagens extraídas dos *frames* dos vídeos, garantindo que as imagens de cada pasta (correspondentes a cada uma das 27 classes) não estivessem presentes em mais de um dos conjuntos. Essa separação foi essencial para assegurar que o modelo fosse avaliado de maneira justa e precisa, utilizando apenas dados que não foram vistos durante o treino.

Como pode ser observado na Figura 4.7, o número de imagens no conjunto de treino é de 94064, sendo significativamente maior do que nos conjuntos de validação, com 6157 imagens, e teste, com 2520 imagens. Esta disparidade deve-se ao uso de técnicas de *data augmentation* aplicadas sobre as imagens de treino, criando variações artificiais a partir das imagens originais. Ao gerar diferentes versões de uma mesma imagem (alterando, por

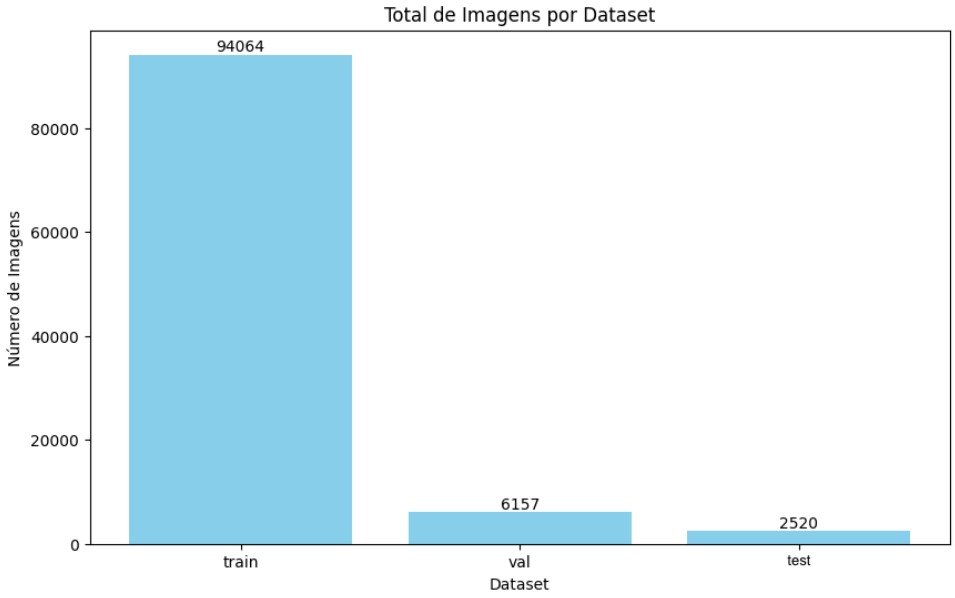


Figura 4.7: *Contagem de imagens do dataset, dividido por treino, validação e teste*

exemplo, o ângulo, o brilho e a posição), foi possível aumentar a diversidade de amostras apresentadas ao modelo, melhorando assim a sua capacidade de generalização.

Adicionalmente, a Figura 4.8 mostra a contagem de imagens de treino, validação e teste dividida por cada classe (letras da LGP e a classe "nothing"). Essa visualização fornece uma melhor compreensão do equilíbrio entre as diferentes classes, essencial para garantir que o modelo não seja enviesado em relação a letras com mais ou menos exemplos disponíveis.

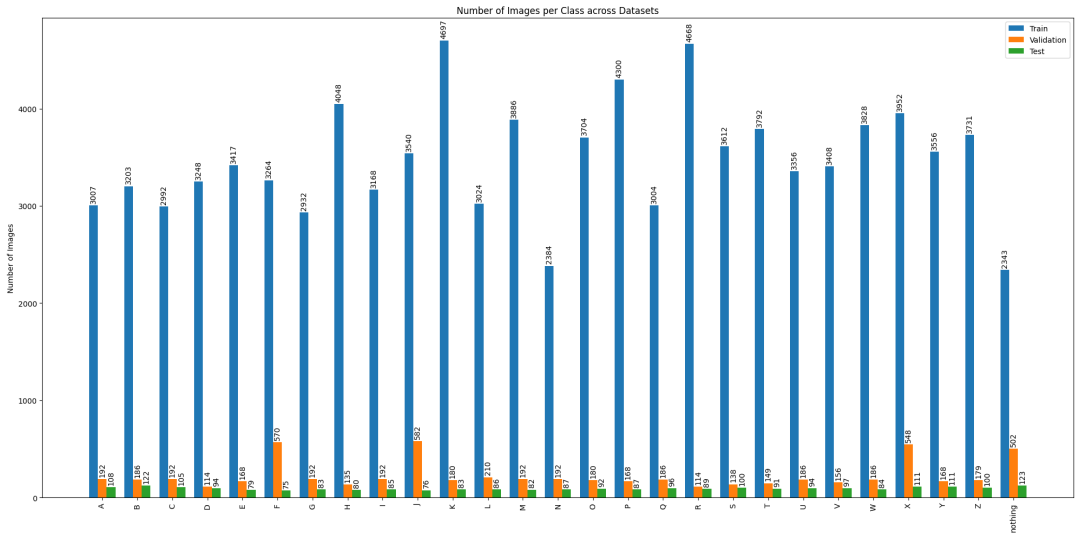


Figura 4.8: *Contagem de imagens do dataset por classe, dividida entre treino, validação e teste*

Além disso, a Figura 4.9 apresenta exemplos de imagens dos três conjuntos de dados (treino, validação e teste), ilustrando a diversidade visual incluída em cada conjunto. Nos Anexos, as Figuras I.1 à I.6 mostram mais exemplos detalhados de cada um dos *datasets* individuais, proporcionando uma visão mais ampla da variedade de gestos capturados e processados.

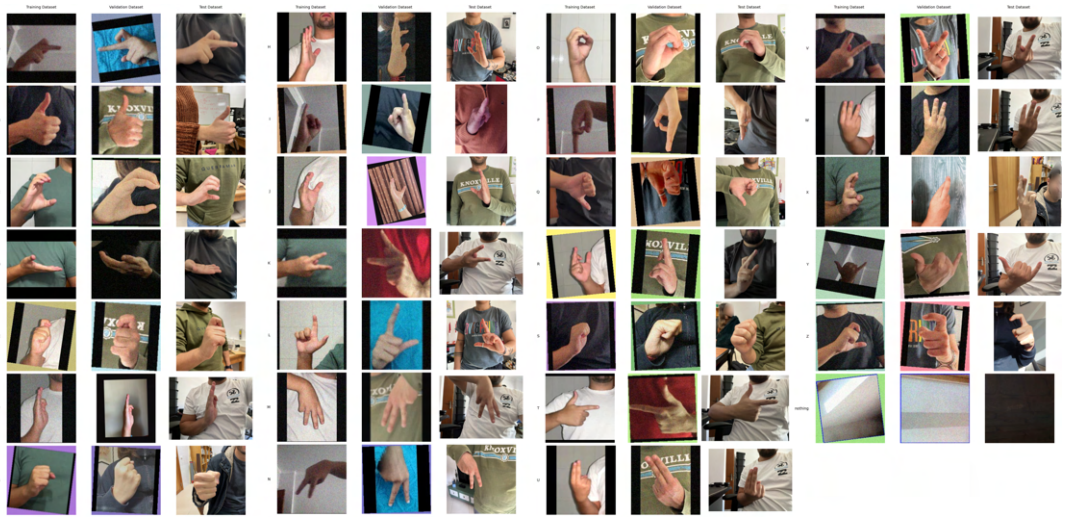

Figura 4.9: *Exemplos de imagens de cada classe dos conjuntos de treino, validação e teste*

## 4.3 Considerações Finais

A criação e preparação de um *dataset* personalizado para o reconhecimento de letras da LGP foi um dos grandes desafios deste trabalho, devido à ausência de *datasets* públicos adequados para esta tarefa. A recolha de dados diversificados, o pré-processamento cuidadoso – incluindo a normalização do tamanho das imagens e ajustes de qualidade – e a aplicação de técnicas de *data augmentation* foram etapas fundamentais para garantir que os modelos desenvolvidos fossem treinados com dados consistentes e representativos. Este processo foi essencial para melhorar a precisão dos modelos e assegurar a sua capacidade de generalizar para diferentes cenários reais.

# 5 Implementação

Este capítulo descreve a implementação detalhada do sistema de reconhecimento de letras do alfabeto da LGP, focando-se em três componentes principais: o desenvolvimento dos modelos de classificação, o desenvolvimento da aplicação móvel, e a implementação do servidor de apoio para processamento de predições. A integração dessas três áreas foi essencial para garantir um sistema confiável – capaz de lidar com variações nos gestos e nas condições de captura – e otimizado para fornecer resultados precisos e rápidos, tanto em modo local como remoto.

A Figura 5.1 apresenta o esquema geral do sistema, ilustrando a integração entre os componentes principais, desde a captura de gestos na aplicação móvel até ao processamento dos modelos e a exibição da tradução em tempo real.

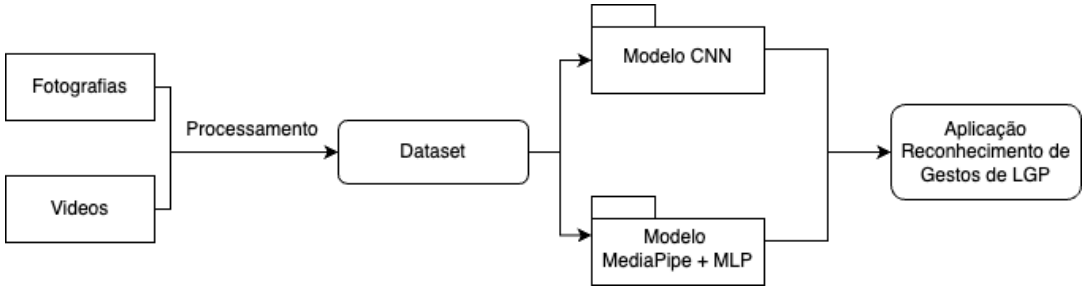


Figura 5.1: *Esquema completo do sistema*

## 5.1 Modelos de Classificação

Nesta secção, serão detalhadas as etapas e decisões tomadas durante o desenvolvimento dos dois modelos de aprendizagem automática utilizados no sistema. Cada modelo foi projetado para uma situação específica, garantindo que o sistema funcione de forma flexível, independentemente das condições de conectividade.

A Figura 5.2 apresenta o fluxo de construção dos dois modelos de classificação utilizados no projeto: o primeiro baseado numa CNN, otimizado para dispositivos móveis através do *TensorFlow Lite*, e o segundo baseado na deteção de *landmarks* das mãos com o MediaPipe, integrado com uma MLP. Este diagrama serve como uma visão geral do desenvolvimento

dos modelos, cujas etapas serão discutidas em detalhe nas secções seguintes.

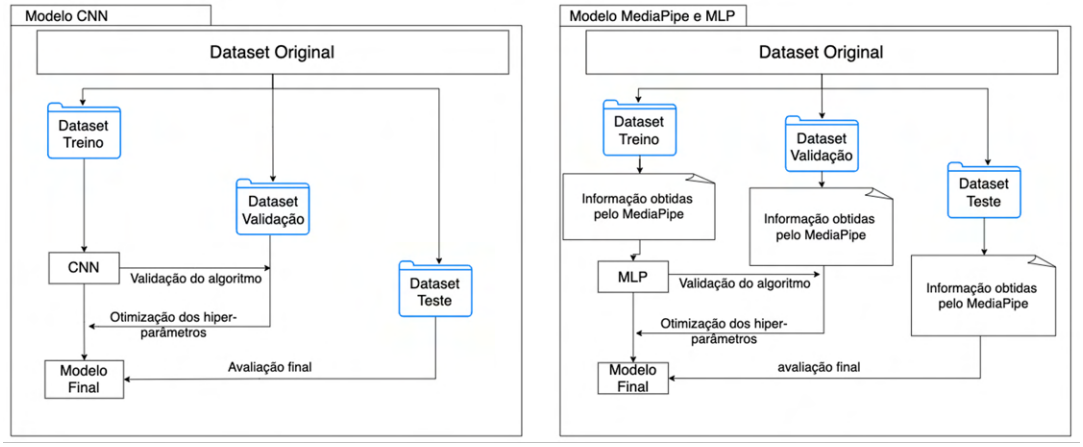


Figura 5.2: *(a) Modelo CNN e (b) Modelo MediaPipe + MLP*

### 5.1.1 Modelo 1: CNN para Reconhecimento de Letras

As Redes Neuronais Convolucionais (CNNs) foram selecionadas como a abordagem inicial para o reconhecimento de letras da LGP, devido à sua eficácia comprovada em tarefas de visão computacional. As CNNs são particularmente adequadas para a extração de características visuais complexas, como as configurações dos dedos nos gestos, sendo uma escolha robusta para este desafio.

Para garantir a eficiência do sistema em dispositivos móveis, optou-se pela arquitetura *MobileNetV2*, conhecida pelo seu equilíbrio entre precisão e eficiência de recursos [29]. Esta arquitetura, aliada ao *transfer learning*, permitiu a utilização de pesos pré-treinados no *ImageNet*, acelerando o processo de treino e melhorando a generalização do modelo no reconhecimento de gestos.

#### 5.1.1.1 Arquitetura do Modelo

A *MobileNetV2* foi projetada especificamente para dispositivos com limitações de processamento e memória. Uma das suas principais inovações é a utilização de blocos de *inverted residuals*, que permitem uma comunicação eficiente entre as camadas mais superficiais (responsáveis por capturar padrões básicos) e as camadas mais profundas (que identificam padrões mais complexos). Isso otimiza a extração de características visuais sem aumentar significativamente a complexidade ou o número de parâmetros do modelo.

Com esta abordagem, o modelo consegue manter um número reduzido de parâmetros, o que é essencial para a execução eficiente em dispositivos móveis, sem comprometer a capacidade de identificar padrões visuais detalhados nos gestos da LGP.

A *MobileNetV2* mostrou-se especialmente adequada para ser utilizada com *transfer learning*, devido à sua arquitetura leve e modular. Essa característica permite que camadas pré-treinadas em grandes bases de dados, como o *ImageNet*, sejam facilmente ajustadas para tarefas específicas, como o reconhecimento de letras da LGP. A reutilização de pesos

pré-treinados no *ImageNet* facilitou a adaptação do modelo à tarefa específica de reconhecimento de letras da LGP, sem a necessidade de treinar uma CNN desde o início. Ao congelar as camadas inferiores, responsáveis por aprender padrões visuais genéricos, o ajuste focou-se apenas nas camadas superiores, que foram treinadas para reconhecer as 27 classes específicas da LGP (26 letras e a classe "nothing").

A escolha de um modelo pré-treinado revelou-se particularmente vantajosa devido à natureza do *dataset*, que, embora extenso, apresenta baixa variabilidade entre as imagens. Essa característica poderia limitar a capacidade de generalização de um modelo treinado do zero. O *transfer learning*, ao aproveitar o conhecimento previamente adquirido sobre configurações visuais amplamente presentes em grandes bases de dados como o *ImageNet* – incluindo formas, texturas e estruturas gerais – permitiu compensar a limitação de variabilidade do *dataset* de treino. Este conhecimento foi ajustado para reconhecer configurações específicas dos dedos e gestos da LGP, melhorando significativamente a capacidade de generalização do modelo.

Após a extração de características visuais pelas camadas convolucionais da *MobileNetV2*, o modelo foi complementado com uma camada de *Global Average Pooling*. Esta camada substitui parcialmente o papel das camadas densas tradicionais na compressão das informações, ao reduzir o número de parâmetros, diminuindo assim o risco de *overfitting*. No entanto, as camadas densas continuam presentes na arquitetura, sendo utilizadas nas etapas finais para mapear as características extraídas para as classes correspondentes, assegurando a capacidade de classificação do modelo. Esta abordagem ajudou a mitigar o risco de *overfitting*, especialmente importante devido ao conjunto de dados limitado disponível. Para refinar a saída da rede, foram adicionadas duas camadas densas com 128 e 64 neurónios, respetivamente. Estas camadas complementam o papel da camada de *Global Average Pooling*, agregando maior capacidade de modelagem para mapear as características extraídas às classes finais. A inicialização *He Normal* foi utilizada para atribuir valores iniciais aos pesos de forma a manter a variância dos gradientes estável, especialmente em redes profundas, garantindo uma convergência mais eficiente durante o treino. Adicionalmente, a ativação Rectified Linear Unit, uma função de ativação utilizada em redes neuronais (ReLU) foi aplicada para introduzir não-linearidade e a normalização por *batch* foi utilizada para estabilizar o processo de treino, reduzindo o risco de *overfitting* e acelerando a convergência.

A arquitetura completa do modelo é ilustrada na Figura 5.3.

### 5.1.2 Treino e Ajustes de Hiperparâmetros

O modelo foi treinado ao longo de até 100 épocas, com a implementação de *Early Stopping* para interromper o treino antecipadamente caso não fossem observadas melhorias significativas na perda de validação por um número predefinido de épocas consecutivas. Foram ajustados vários hiperparâmetros importantes, como o tamanho do *batch*, definido como 32, e a taxa de aprendizagem inicial, configurada como 0.001. Esta taxa foi escolhida para equilibrar a rapidez do treino com a precisão final, garantindo que o modelo aprendesse de forma progressiva sem convergir prematuramente para um mínimo local inadequado.

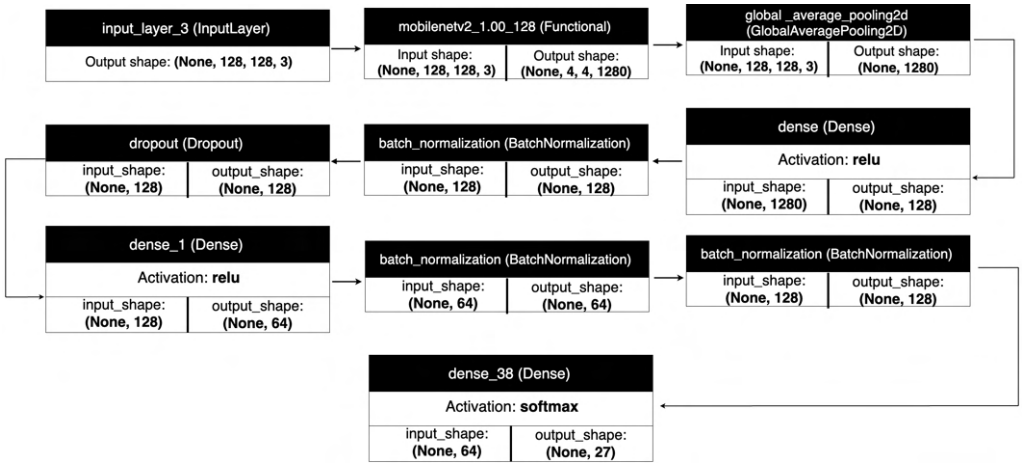


Figura 5.3: *Diagrama da arquitetura CNN com o modelo pré-treinado MobileNetV2.*

Um otimizador *Adam* foi utilizado para acelerar o processo de convergência, pois adapta dinamicamente a taxa de aprendizagem com base no gradiente, permitindo que o modelo atingisse uma performance consistente de forma eficiente.

À medida que o treino avançava, foi introduzido um *scheduler* de taxa de aprendizagem, que reduzia automaticamente o valor da taxa sempre que a função de perda no conjunto de validação estagnava. A perda de validação mede o erro do modelo ao prever em dados que não foram usados no treino, sendo um indicador da capacidade do modelo de generalizar para novos dados. A redução gradual da taxa de aprendizagem permitiu ajustes mais finos nas etapas finais do treino, evitando oscilações desnecessárias ao redor dos mínimos locais. O *Early Stopping* foi configurado para monitorizar a perda de validação, encerrando o treino se não houvesse melhorias significativas após 10 épocas consecutivas, prevenindo assim o *overfitting* e otimizando o tempo de treino.

O ajuste contínuo dos hiperparâmetros, combinado com a utilização de técnicas como o *scheduler* de taxa de aprendizagem e o *Early Stopping*, permitiu que o modelo atingisse uma boa capacidade de generalização com um máximo de 100 épocas de treino, embora a média de treino efetivo tenha sido de cerca de 60 épocas devido à interrupção antecipada. O *scheduler* ajustava dinamicamente a taxa de aprendizagem ao longo do treino, refinando as atualizações dos pesos em fases críticas, enquanto o *Early Stopping* preveniu o *overfitting* ao monitorizar a perda de validação e interromper o treino em momentos de estagnação, otimizando tanto a eficiência computacional quanto a qualidade do modelo final.

### 5.1.3 Ajustes de Pesos de Classes

Durante as primeiras etapas de treino e validação, foi identificado que algumas classes, em particular as letras “G”, “X”, “F”, e “L”, apresentavam uma precisão significativamente inferior em relação às outras. Esse comportamento pode ser atribuído à dificuldade inerente dessas classes, que possuem gestos com maior sobreposição visual ou menos exemplos no conjunto de treino.

ara lidar com este desequilíbrio, foi implementado um ajuste de pesos nas classes, atribuindo pesos maiores às classes com menor frequência ou precisão no conjunto de treino. Por

exemplo, para classes como "G"e "X", que apresentavam uma menor taxa de acerto inicial, foram aplicados pesos proporcionalmente maiores, aumentando a penalização dos erros dessas classes durante o cálculo da função de perda. Esta abordagem, frequentemente recomendada para cenários de dados desbalanceados [13], ajudou a equilibrar o impacto das classes no treino e a melhorar o desempenho em classes mais desafiadoras. Este método visa penalizar mais os erros em classes com menor precisão, incentivando o modelo a focar-se mais nas classes com baixo desempenho. Inicialmente, foi utilizada uma abordagem automática, em que os pesos de cada classe foram calculados de forma inversamente proporcional à precisão observada durante os primeiros ciclos de validação. A fórmula utilizada foi:

$$\text{Weight} = \frac{1}{\text{Accuracy}} \tag{5.1}$$

A fórmula descrita implementa uma estratégia em que classes com menor desempenho (menor precisão) recebem pesos maiores, diretamente aplicados na função de perda. Este ajuste tem como objetivo aumentar a penalização dos erros associados a essas classes, incentivando o modelo a focar mais nos gestos mais desafiadores durante o treino e melhorar sua capacidade de correção.

Inicialmente, os pesos foram calculados de forma inversamente proporcional às precisões observadas para cada classe, com base nos resultados das predições iniciais. Esse ajuste automático garantiu que classes com menor precisão, como ”X” e ”G”, recebessem maior atenção no processo de otimização. Após esta etapa, os pesos foram normalizados para manter o equilíbrio geral entre as classes no cálculo da perda.

Apesar da melhoria geral observada após o ajuste automático, algumas classes continuaram a apresentar dificuldades. Para lidar com essas limitações, foi realizada uma intervenção manual com base nos resultados de validação intermédios, atribuindo pesos ainda maiores a essas classes problemáticas, como “X” e “G”. Esta abordagem manual visou reforçar o impacto destas classes no cálculo da função de perda e promover melhorias adicionais no desempenho.

Após os ajustes, observou-se uma melhoria significativa em algumas classes, como “H” e “L”, cujas predições aumentaram consideravelmente. No entanto, classes como “X” e “G” continuaram a apresentar desafios, mesmo após a intervenção. Estes resultados sugerem que a complexidade visual dessas letras pode exigir estratégias complementares, como a utilização de um segundo modelo ou a ampliação do conjunto de dados para essas classes específicas.

### 5.1.4 Considerações Finais

O desenvolvimento deste modelo baseado na arquitetura *MobileNetV2* resultou numa possível solução para o reconhecimento de letras da LGP em dispositivos móveis. O uso de técnicas como *transfer learning* e ajustes de pesos de classes contribuiu para melhorar o desempenho global, especialmente em algumas das classes mais desafiantes.

No entanto, o modelo ainda apresentou limitações em gestos visualmente semelhantes, como as letras "G" e "X", sugerindo a necessidade de abordagens complementares. As soluções alternativas, como a deteção de *landmarks* com MediaPipe, serão exploradas no desenvolvimento do segundo modelo.

## 5.2 Modelo 2: MediaPipe para Deteção dos Pontos das Mãos

Após a implementação do primeiro modelo baseado em CNN, foram identificadas várias limitações, especialmente em classes com gestos mais complexos ou com sobreposições dos dedos. Para abordar esses desafios e melhorar o reconhecimento de gestos da LGP, decidiu-se implementar um segundo modelo, utilizando o MediaPipe para a deteção de *landmarks* das mãos e uma MLP para a classificação dos gestos.

O MediaPipe foi escolhido por ser uma solução eficiente para capturar os pontos-chave das mãos, extraindo informações geométricas diretamente a partir de vídeos ou imagens. Ao detetar 21 *landmarks* em tempo real, o MediaPipe simplifica o pré-processamento, focando-se nas coordenadas dos pontos essenciais, eliminando a necessidade de processar a imagem completa.

Para a classificação das coordenadas detetadas, foi utilizada uma MLP, oferecendo um equilíbrio entre simplicidade e precisão. A MLP consegue modelar relações não lineares entre as posições dos dedos e as classes de letras da LGP, sendo mais leve e computacionalmente menos exigente em comparação com as CNNs, o que a torna mais eficiente para este tipo de tarefa que usa os pontos das mãos para a classificação.

### 5.2.1 Arquitetura do Sistema

Como referido, a arquitetura deste segundo modelo combina a deteção dos *landmarks* das mãos, realizada pelo MediaPipe, com uma MLP responsável pela classificação dos gestos.O MediaPipe extrai 21 pontos de referência tridimensionais das mãos, representados pelas coordenadas X, Y e Z de cada ponto, resultando em um total de 63 valores numéricos. Essas coordenadas descrevem a posição relativa de cada ponto no espaço 3D, capturando a estrutura e a pose da mão. Esses dados são utilizados como entrada para a MLP, que processa estas informações para realizar a classificação dos gestos.

O *MediaPipe Hands* utiliza dois modelos principais para alcançar uma deteção eficiente e precisa das mãos:

1. O **modelo de deteção inicial das mãos** foi concebido para identificar as palmas em tempo real, servindo como ponto de partida para a localização das mãos completas. Este modelo utiliza um detetor de uma única etapa, simplificando o processo ao focar na área da palma, em vez de tentar identificar diretamente mãos com dedos articulados. A arquitetura incorpora um extrator de características *encoder-decoder*, que analisa o contexto da cena, e utiliza a técnica de perda focal (*focal loss*) para lidar com a variação significativa no tamanho e escala das palmas. Este design apresenta

uma precisão média de 95,7% na deteção das palmas, mesmo em cenários desafiadores, como o de oclusão parcial, onde as mãos se sobrepõem durante um aperto de mão.

2. O modelo de **detecção e mapeamento de pontos-chave das mãos** calcula as coordenadas tridimensionais (X, Y, Z) de 21 pontos específicos nas mãos dentro das áreas previamente identificadas. Utiliza um método de regressão para determinar as posições das articulações das mãos e foi treinado com cerca de 30.000 imagens, incluindo dados reais e sintéticos, garantindo uma maior robustez mesmo em situações de oclusão parcial ou mãos parcialmente visíveis. Este modelo fornece uma representação consistente da pose das mãos, mesmo em condições adversas [16].

A robustez do MediaPipe, especialmente em lidar com variações de pose, iluminação e oclusão, foi fundamental para garantir que o sistema pudesse ser aplicado de forma prática em interfaces de utilizador baseadas em gestos, como a classificação de letras da LGP. A capacidade de fornecer uma representação consistente das mãos, mesmo em cenários adversos, foi essencial para o sucesso da classificação subsequente.

A seguir, os pontos detetados pelo MediaPipe são processados pela MLP, que foi projetada para classificar os gestos com base nas coordenadas dos 21 *landmarks*. Inicialmente, essas coordenadas foram usadas diretamente como entradas para uma MLP simples, composta por três camadas densas e camadas de *dropout* para mitigar o risco de *overfitting*. . Embora semelhante à estrutura ilustrada na Figura 5.4, esta versão é mais simplificada, tendo menos camadas e menor densidade de neurónios, adaptando-se à menor complexidade do conjunto de *features* usado.

#### 5.2.1.1 Problemas Iniciais com Generalização e Deteção

Durante as primeiras fases de validação, verificou-se que o modelo apresentava problemas de generalização, particularmente em classes mais complexas, como "G", "J" e "X". Em testes preliminares, estas classes registaram precisões de 15%, 36% e 32%, respetivamente. Esses resultados indicaram que o modelo não conseguia capturar adequadamente as variações subtis nos gestos ou nas condições de captura.

As causas principais deste baixo desempenho incluíam:

- **Deteção inconsistente dos *landmarks***: Em gestos mais complicados, onde os dedos estavam sobrepostos ou em posições não usuais, o MediaPipe por vezes falhava na deteção correta de alguns pontos;
- **Estrutura simplificada da rede MLP**: A rede neuronal inicial, com poucas camadas, não era suficientemente robusta para distinguir as nuances nos gestos, resultando em previsões incorretas.

### 5.2.2 Ajustes Implementados para Mitigação de Problemas

Com base nos desafios observados, foram implementados diversos ajustes para melhorar o desempenho do modelo:

1. **Cálculo de Novas *Features***: Para capturar mais informações sobre a geometria dos gestos, além das coordenadas dos *landmarks*, foram calculadas novas *features*, como as distâncias entre pares de *landmarks* e os ângulos formados por grupos de três pontos. Estas *features* adicionais permitiram uma representação mais rica dos gestos, expandindo os dados de 3D para um espaço de maior dimensão, que inclui tanto as coordenadas originais dos *landmarks* quanto as distâncias e ângulos entre pares de pontos. Essa nova representação ajudou o modelo a distinguir gestos que envolvem variações subtis.

2. **Aumento da Complexidade da Rede**: A arquitetura da MLP foi expandida para lidar com o novo conjunto de *features*. A nova rede passou a incluir:

   - Uma camada densa com 256 neurónios e ativação ReLU.
   - Uma camada de normalização por *batch*, para estabilizar o treino.
   - Duas camadas densas subsequentes, com 128 e 64 neurónios, respetivamente, com *dropout* para prevenir o *overfitting*.
   - Uma camada de saída *softmax* para classificar as letras da LGP.

Essas modificações aumentaram a capacidade do modelo de capturar as variações subtis entre os gestos, resultando numa melhoria geral na precisão e numa maior capacidade de generalização. Tal arquitetura pode ser observada na Figura 5.4.

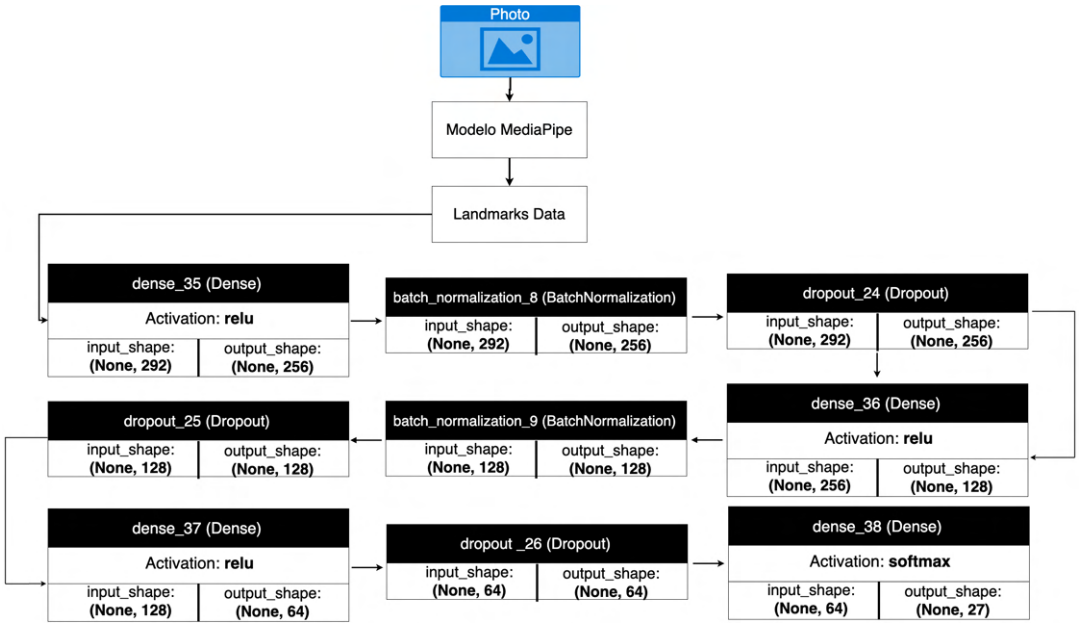


Figura 5.4: *Diagrama da nova arquitetura da rede neuronal.*

### 5.2.3 Ajustes Finais: Pesos de Classes e Resultados Gerais

Mesmo após as melhorias na arquitetura e no cálculo de *features*, algumas classes continuaram a ter baixo desempenho. Para mitigar esse problema, foram implementados ajustes nos pesos de classes, baseados nas precisões observadas nas primeiras iterações de treino.

Os pesos das classes foram ajustados inversamente proporcional à sua precisão inicial, permitindo que o modelo prestasse mais atenção às classes mais difíceis, como "J", "G" e "X".

Após o ajuste dos pesos, a precisão destas classes melhorou significativamente. A precisão da letra "J" subiu para 90%, enquanto "F" e "X" atingiram 70% e 71%, respetivamente. No entanto, algumas classes continuaram a apresentar dificuldades, indicando que podem ser necessários ajustes adicionais em futuras iterações.

### 5.2.4 Considerações Finais

O desenvolvimento do segundo modelo, combinando a deteção de *landmarks* com o MediaPipe e a classificação com uma MLP, resultou numa abordagem eficiente para o reconhecimento de gestos. As modificações introduzidas, incluindo o cálculo de novas *features* e o ajuste dos pesos de classes, contribuíram para melhorias substanciais na precisão, especialmente em classes que apresentavam desafios no modelo anterior.

Embora algumas classes continuem a exigir melhorias, este modelo mostrou-se uma solução promissora e menos exigente em termos de carga computacional, sendo uma boa alternativa a modelos de classificação de imagem.

## 5.3 Aplicação Móvel

O desenvolvimento da aplicação em Flutter foi um componente fundamental desta tese, com o objetivo de criar uma ferramenta intuitiva e prática para a tradução em tempo real da LGP para texto. O Flutter foi escolhido pela sua capacidade de desenvolver aplicações multiplataforma (Android e iOS) com uma única base de código, o que simplifica a manutenção ao permitir atualizações consistentes e simultâneas para ambas as plataformas. Além disso, o Flutter oferece ferramentas que facilitam a criação de interfaces de utilizador rápidas e adaptáveis, garantindo que a aplicação responda bem em dispositivos com diferentes tamanhos de ecrã e capacidades. Para além da tradução de gestos para texto, a aplicação inclui funcionalidades que incentivam a aprendizagem da LGP, melhorando a experiência global do utilizador.

A Figura 5.5 apresenta o fluxo geral do desenvolvimento da aplicação, desde a interface do utilizador até à integração dos dois modelos de reconhecimento de gestos. O diagrama ilustra como a aplicação interage com os modelos *TensorFlow Lite* e MediaPipe. Esta visão geral, e toda a interação da aplicação com o utilizador, será explicada em mais detalhe nas secções seguintes.

### 5.3.1 Funcionalidades

A principal funcionalidade da aplicação é a tradução em tempo real de gestos da LGP para letras, que são agregadas numa palavra exibida ao utilizador. Esta funcionalidade é essencial para o objetivo central da aplicação: facilitar a comunicação entre utilizadores surdos e ouvintes.

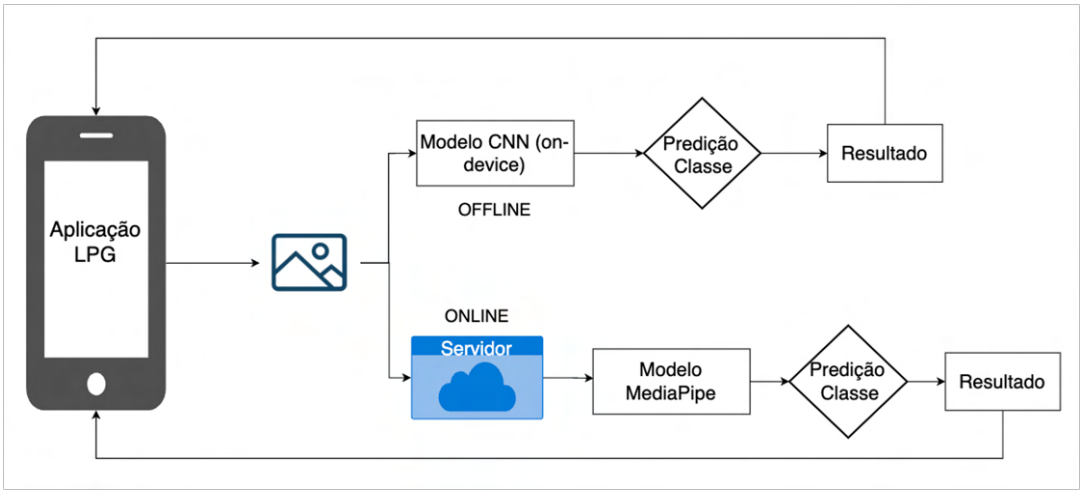


Figura 5.5: *Fluxo da classificação na aplicação*

A navegação na aplicação é feita através de uma barra de navegação inferior (*bottom navigation bar*), onde o utilizador pode clicar em ícones personalizados para aceder às diferentes páginas. A interação foi concebida com base em princípios de design centrado no utilizador, utilizando elementos visuais claros e uma navegação direta, que facilitam a utilização das funcionalidades da aplicação. O layout simples e as opções acessíveis na interface contribuem para uma experiência intuitiva, permitindo ao utilizador tirar proveito das várias funcionalidades disponíveis:

1. **Página Principal (Home Page)**
   Na página inicial, o utilizador é recebido com uma "letra do dia", escolhida aleatoriamente. O utilizador pode clicar num botão central para atualizar a letra e aprender novos gestos de forma aleatória e contínua. Esta funcionalidade foi pensada para incentivar os utilizadores a aprender e praticar uma letra da LGP de forma aleatória a cada acesso.

2. **Página de Câmara (Camera Page)**
   Nesta página, a tradução dos gestos ocorre de forma contínua, com a câmara do dispositivo capturando os gestos do utilizador a cada segundo. O reconhecimento é validado quando a mesma letra é identificada consecutivamente em pelo menos duas iterações, garantindo maior precisão na apresentação do resultado na interface. As letras são depois agrupadas numa frase que é continuamente atualizada à medida que o utilizador faz novos gestos. Há ainda um botão que permite alternar entre a câmara frontal e traseira, conforme a preferência do utilizador. Para além disso, um *switch* manual possibilita a alternância entre dois modelos de reconhecimento de gestos: o MediaPipe ou o modelo CNN de forma local. garantindo a continuidade do reconhecimento de gestos.

3. **Página de Letras (Letters Page)**
   Nesta página, o utilizador pode visualizar todas as letras da LGP e a sua correspondência no alfabeto português. Ao clicar numa letra, uma imagem detalhada é exibida,

mostrando como realizar o gesto corretamente, tanto da perspetiva do emissor quanto do recetor.

4. **Página de Tradução de Texto (Translate Page)**
   Esta página permite ao utilizador introduzir texto em português e visualizar como as letras correspondentes seriam gesticuladas em LGP. Esta funcionalidade serve como uma ferramenta educativa, ajudando os utilizadores a aprender a traduzir palavras e frases do português para a LGP.

A Figura 5.6 abaixo apresenta capturas de ecrã das principais páginas da aplicação, ilustrando o design da interface e as funcionalidades descritas acima.

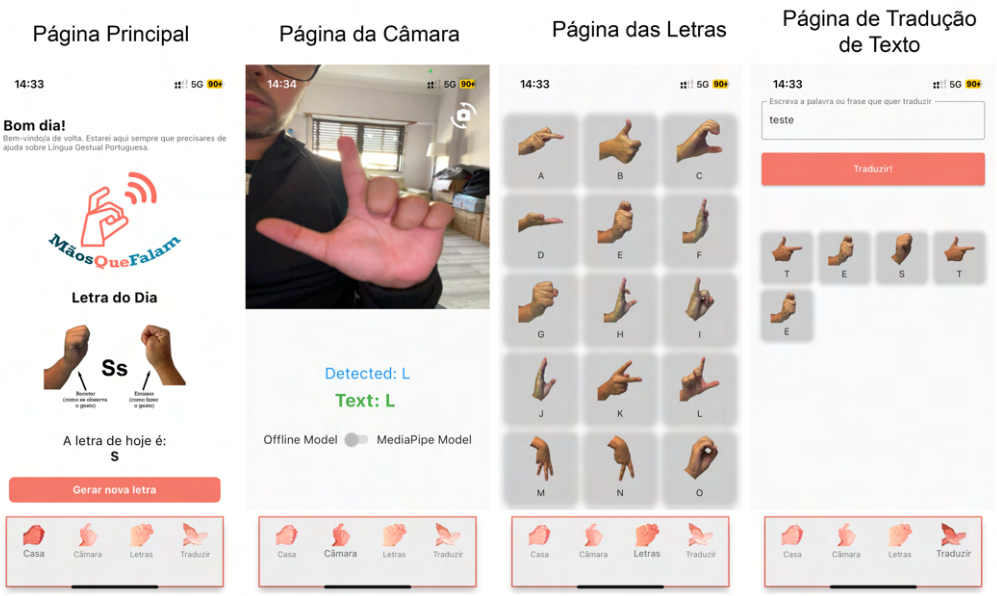


Figura 5.6: *Capturas de ecrã das principais páginas da aplicação.*

### 5.3.2 Estrutura e Organização do Código

A aplicação foi estruturada de forma modular para facilitar a manutenção e permitir a escalabilidade futura. A organização dos ficheiros segue uma estrutura clara, separando as funcionalidades principais e as responsabilidades de cada componente, facilitando o desenvolvimento de novas funcionalidades e a gestão da aplicação.

```
lib
|- main.dart
|- utils
|   |- customNavbar.dart
    \- tf_classifier
        |- classes.dart
        |- classifier_category.dart
        |- classifier_model.dart
        |- classifier.dart
        |- image_utils.dart
        |- predict.dart
\- views
    |- camera_page.dart
    |- home_page.dart
    |- init_page.dart
    |- letters_page.dart
    |- translate_page.dart
```

Listagem 5.1: Estrutura de Diretórios da Aplicação

Abaixo encontra-se um resumo das responsabilidades de cada ficheiro:

- **main.dart**: Inicializa a aplicação e carrega a página inicial (init_page), configurando o menu de navegação.
- **utils/customNavbar.dart**: Define a lógica para o menu de navegação personalizado.
- **views/camera_page.dart**: Controla a captura de imagens em tempo real e permite alternar entre a câmara frontal e traseira.
- **views/home_page.dart**: Mostra a "letra do dia" e permite ao utilizador atualizá-la.
- **views/letters_page.dart**: Exibe todas as letras da LGP e as suas correspondências.
- **views/translate_page.dart**: Implementa a funcionalidade de tradução de texto para gestos.

A pasta **tf_classifier/** contém os componentes responsáveis pela classificação de gestos.

- **classes.dart**: Define as classes de deteção (A a Z e "nothing") e mapeia-as para strings correspondentes.
- **classifier_category.dart**: Representa a categoria de predição com o rótulo e a confiança da predição.
- **classifier_model.dart**: Carrega o modelo *TensorFlow Lite (tflite)* e define a estrutura do modelo, incluindo dimensões de entrada e saída.

- **classifier.dart**: Executa o fluxo de predição, desde o pré-processamento da imagem até à inferência no modelo.
- **image_utils.dart**: Converte as imagens capturadas para o formato adequado ao modelo.
- **predict.dart**: Coordena o processo de predição, garantindo o correto funcionamento da classificação.

## 5.4 Integração dos Modelos com a Aplicação

A integração dos modelos de reconhecimento de gestos com a aplicação móvel foi um dos principais desafios técnicos deste projeto, exigindo uma abordagem híbrida para garantir a flexibilidade e o desempenho desejados em diferentes cenários.

O modelo CNN foi inicialmente convertido para *TensorFlow Lite* para ser executado diretamente nos dispositivos móveis. O *TensorFlow Lite* foi escolhido devido à sua capacidade de executar modelos de *machine learning* em dispositivos com recursos limitados, como telemóveis, sem a necessidade de uma conexão constante à internet.

O processo de conversão para *tflite* envolveu a utilização da ferramenta *TensorFlow Converter*, que transforma o modelo original em um formato otimizado para dispositivos móveis. Durante essa conversão, ocorre frequentemente uma quantização dos pesos do modelo, onde os dados originalmente representados em 32 bits (ponto flutuante) são convertidos para 16 bits ou 8 bits. Esta redução no tamanho dos pesos é essencial para melhorar a eficiência do modelo em termos de uso de memória e velocidade de inferência, mas pode introduzir uma ligeira perda de precisão [34].

Além disso, algumas operações complexas do modelo original, como cálculos avançados em camadas convolucionais, normalizações ou funções de ativação específicas, são simplificadas para se adequarem às limitações de processamento dos dispositivos móveis. Essas simplificações, embora otimizem a execução, podem impactar negativamente a capacidade do modelo de captar variações subtis nos gestos, resultando em uma menor qualidade de predição, especialmente em gestos com configurações de dedos mais detalhadas [35].

### 5.4.1 Servidor Flask com MediaPipe

Para resolver os problemas de precisão e desempenho observados com o modelo *tflite*, foi adotada uma abordagem alternativa, integrando o MediaPipe para a deteção dos *landmarks* das mãos. Dado que o MediaPipe não tem suporte nativo no Flutter, foi necessário criar um servidor *backend* em Flask para processar as imagens e realizar as predições de gestos.

O servidor Flask processa as imagens capturadas pela câmara da aplicação, utilizando o MediaPipe para detetar os 21 pontos de referência tridimensionais das mãos (coordenadas X, Y e Z). Com base nesses pontos, o servidor identifica a letra correspondente da LGP e devolve o resultado à aplicação com uma latência média de 3 a 5 segundos, dependendo da qualidade da conexão à internet. Embora este processamento seja realizado com rapidez suficiente para muitos cenários, a dependência de conectividade à internet pode introduzir

atrasos adicionais em ambientes com rede limitada, impactando a experiência de utilizador e reduzindo a sensação de resposta em tempo real.

A comunicação entre a aplicação e o servidor é feita através de uma Application Programming Interface (Interface de Programação de Aplicações) (API) HTTP. Para minimizar a latência e otimizar o desempenho, as imagens transmitidas são compactadas, assegurando um processamento mais rápido. O reconhecimento dos gestos ocorre continuamente em todas as imagens processadas de segundo a segundo, permitindo uma tradução fluida e precisa das letras da LGP.

### 5.4.2 Alternância entre os Modelos

A aplicação foi configurada para permitir a alternância entre os dois modelos de reconhecimento de gestos – o modelo *tflite*, que opera localmente, e o modelo MediaPipe, que funciona via servidor Flask. Essa funcionalidade é implementada através de um *switch* manual na interface, que oferece ao utilizador a flexibilidade de escolher o modelo mais adequado às condições do ambiente de utilização.

Por padrão, o modelo *tflite* é selecionado como base, garantindo que a aplicação funcione de forma fluida e independente da conectividade à internet. O utilizador pode, no entanto, optar por utilizar o modelo MediaPipe, que proporciona maior precisão na deteção dos gestos, clicando no *switch* para ativar esta funcionalidade. Esta configuração é particularmente útil em cenários onde há uma conexão de internet estável e o utilizador prefere priorizar a precisão em vez da latência.

Contudo, para garantir a continuidade do funcionamento da aplicação, foi implementado um mecanismo de *fallback* automático. Caso o modelo MediaPipe seja selecionado, mas ocorra algum erro na comunicação com o servidor Flask – seja devido à falta de conexão à internet ou a outros problemas de conectividade – a aplicação deteta automaticamente a falha e alterna de volta para o modelo *tflite*. Desta forma, o sistema assegura que o reconhecimento de gestos continua a ser realizado, mesmo que com uma ligeira redução na precisão, garantindo uma experiência fluida e confiável para o utilizador.

Essa alternância automática minimiza interrupções no uso da aplicação, permitindo que o utilizador continue a interagir sem precisar fazer alterações manuais na configuração em caso de falha no servidor. Assim, o *switch* proporciona tanto flexibilidade quanto robustez, adaptando-se dinamicamente às condições de conectividade do ambiente.

### 5.4.3 Considerações Finais

A integração híbrida dos modelos *tflite* e MediaPipe foi essencial para garantir que a aplicação oferecesse uma solução robusta e adaptável, funcionando tanto em modo *offline* como *online*. Essa abordagem foi validada através de testes em dispositivos de diferentes gamas e condições de rede. O modelo *tflite*, ao ser executado localmente, demonstrou tempos de resposta rápidos (entre 1 e 2 segundos em dispositivos de maior capacidade) e permitiu o funcionamento da aplicação em ambientes sem conexão à internet, garantindo a continuidade da funcionalidade. Por outro lado, o modelo MediaPipe mostrou-se mais

preciso em gestos complexos, ainda que com latências mais elevadas (3 a 5 segundos), especialmente em redes instáveis.

A introdução de um servidor remoto contribuiu para melhorar a precisão necessária para uma tradução eficaz dos gestos, enquanto o *fallback* automático para o modelo *tflite* assegurou a funcionalidade da aplicação em condições adversas, comprovando a adaptabilidade do sistema. Esses resultados mostram que a solução é capaz de oferecer um desempenho balanceado entre precisão e acessibilidade, adaptando-se às condições específicas de utilização.

## 5.5 Conclusão da Implementação

A implementação do sistema de reconhecimento de letras da LGP envolveu o desenvolvimento e integração de dois modelos principais: um modelo CNN otimizado para execução local em dispositivos móveis e um segundo modelo baseado na deteção de *landmarks* das mãos com o MediaPipe, processado através de um servidor Flask. A criação de uma aplicação móvel em Flutter, capaz de operar em múltiplas plataformas, foi outro ponto essencial para garantir acessibilidade e flexibilidade ao sistema.

Os desafios técnicos enfrentados, como a perda de precisão na conversão para *TensorFlow Lite* e a dependência de conexão à internet no uso do modelo MediaPipe, foram mitigados através da implementação de uma solução híbrida. Esta abordagem garante que o reconhecimento de gestos funcione de forma contínua, alternando automaticamente entre os modelos *tflite* e MediaPipe, adaptando-se às condições de rede e à capacidade dos dispositivos.

No próximo capítulo, serão apresentados e analisados os resultados obtidos com este sistema, avaliando o seu desempenho em diferentes cenários e destacando as áreas onde houve melhorias significativas, bem como as limitações observadas.

# 6 Avaliação

O sistema proposto foi avaliado sob duas perspetivas principais: o desempenho dos modelos de reconhecimento de gestos e a experiência de utilizadores com a aplicação móvel. Primeiro, será analisada a avaliação dos modelos de aprendizagem automática utilizados para classificar as letras da LGP, seguido de uma comparação dos seus desempenhos, análise das classes problemáticas e uma comparação com trabalhos relacionados. Posteriormente, será apresentada a avaliação da aplicação em si, onde foi recolhido *feedback* dos utilizadores através de entrevistas presenciais. Por fim, será discutida a integração dos modelos na aplicação, incluindo os desafios enfrentados e os resultados obtidos em termos de predição, rapidez e experiência do utilizador.

## 6.1 Avaliação dos Modelos

Os dois modelos de reconhecimento de gestos desenvolvidos foram avaliados utilizando métricas de desempenho como *accuracy*, *precision*, *recall*, *f1-score*, e a matriz de confusão. Estas métricas foram calculadas para analisar a capacidade dos modelos em identificar corretamente os gestos das diferentes letras da LGP. A avaliação incluiu tanto o modelo baseado em CNN, como o modelo baseado no MediaPipe, com foco nos seus pontos fortes, fracos e capacidade de generalização.

É importante destacar que ambos os modelos foram testados utilizando o mesmo *dataset* de teste, garantindo que as comparações de desempenho se baseiem nos mesmos exemplos e condições. Dessa forma, evita-se qualquer viés introduzido por diferenças nos dados de entrada, assegurando uma análise objetiva e equitativa entre os dois modelos.

### 6.1.1 Modelo 1: CNN para Reconhecimento de Letras

O primeiro modelo implementado baseou-se numa arquitetura de rede neural convolucional (CNN) utilizando a *MobileNetV2*, e o seu desempenho foi avaliado ao longo de 32 épocas de treino. A análise inicial das curvas de *accuracy* e *loss* revelou alguns padrões importantes que nos ajudaram a entender a capacidade do modelo em generalizar os gestos da LGP para novos dados. Desde o início do treino, observou-se um rápido aumento na exatidão (*accuracy*) no conjunto de treino, atingindo quase 100%. No entanto, a *accuracy* no conjunto

de validação estabilizou relativamente cedo, por volta dos 61.57%. Este padrão, visível na Figura 6.1, indicou sinais claros de *overfitting*. Embora o modelo estivesse a aprender os padrões do conjunto de treino, teve dificuldades em generalizar para os dados de validação, sugerindo que memorizava os exemplos de treino, em vez de aprender os padrões subjacentes que pudessem ser aplicados a novos dados. A introdução de técnicas de regularização, como *dropout*, e a redução gradual da taxa de aprendizagem ajudaram a mitigar parte deste *overfitting*, mas ainda assim, o modelo não conseguiu melhorar significativamente a *accuracy* no conjunto de validação nas últimas épocas de treino.

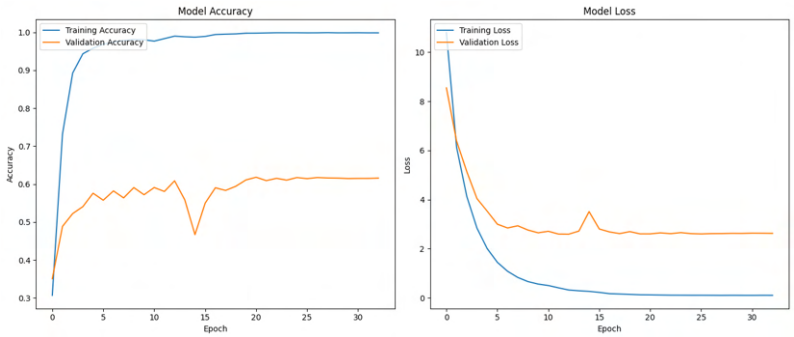


Figura 6.1: *Curva de Accuracy e Loss do modelo CNN ao longo do treino.*

Para obter uma visão mais detalhada do comportamento do modelo em cada classe de gestos, foi gerada a matriz de confusão, apresentada na Figura 6.2. Esta matriz permitiu observar tanto as forças quanto as fraquezas do modelo no reconhecimento das diferentes letras do alfabeto gestual. Por exemplo, as classes "A" e "B" apresentaram resultados sólidos, com um elevado número de verdadeiros positivos e poucos falsos positivos, indicando que o modelo foi eficaz em identificar corretamente estas letras na maioria das instâncias. No entanto, outras classes, como "X" e "G", evidenciaram limitações significativas, sendo frequentemente confundidas com letras cujos gestos possuem semelhanças visuais, como "V" e "P". Estes resultados destacam que, enquanto o modelo demonstrou bom desempenho em algumas classes, enfrentou dificuldades em capturar variações subtis nos gestos de outras.

Os resultados detalhados para cada classe foram ainda mais evidentes no relatório de desempenho do modelo apresentado na Tabela 6.1. Este relatório revelou que o modelo atingiu uma precisão média de 79% e um *f1-score* global de 76%, refletindo a sua capacidade moderada de equilibrar a precisão e o *recall* em diferentes classes. No entanto, também expôs discrepâncias significativas entre as classes mais fáceis e as mais difíceis. Por exemplo, enquanto as letras como "A" e "B" mantiveram altos valores de *precision* e *recall*, classes como "G" e "X" apresentaram *f1-scores* baixos devido à elevada taxa de falsos positivos e falsos negativos. A classe "G", por exemplo, teve um *recall* de apenas 36%, indicando que o modelo falhou em identificar corretamente muitas das suas instâncias. Da mesma forma, a dificuldade em distinguir o gesto da letra "X" de outros gestos semelhantes resultou num desempenho fraco nesta classe.

As confusões mais frequentes envolveram gestos que, embora diferentes, apresentavam características visuais semelhantes, como a posição dos dedos ou a orientação da mão. Este tipo de erro pode ser mitigado com a introdução de mais exemplos dessas classes

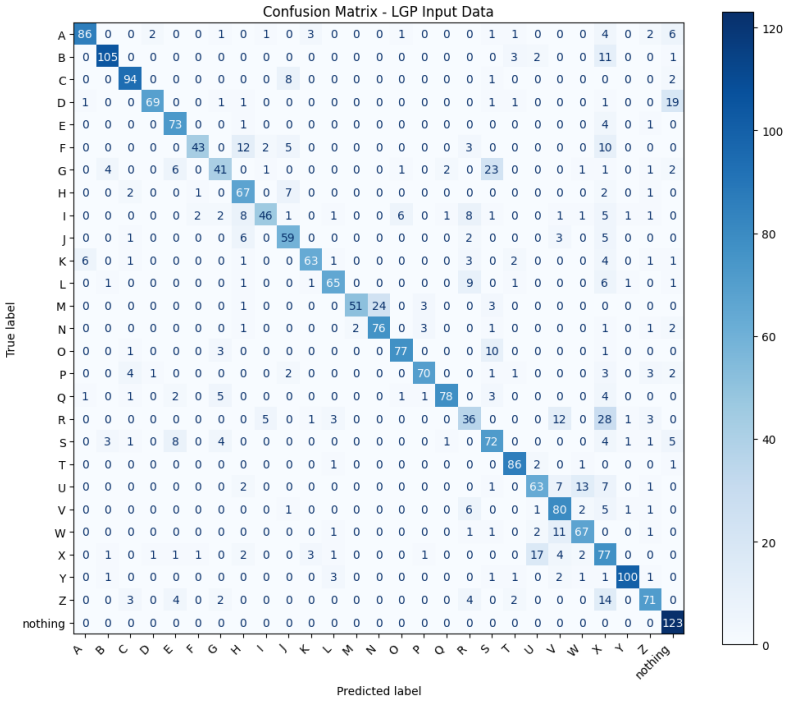


Figura 6.2: *Matriz de confusão do modelo CNN.*

problemáticas no conjunto de treino e com o ajuste dos pesos das classes durante o treino. O ajuste manual de pesos tinha sido testado nas iterações iniciais do desenvolvimento, mas com resultados limitados, o que sugere que talvez seja necessário um conjunto de treino mais equilibrado e diversificado para melhorar o desempenho em classes como "X" e "G".

Para ilustrar visualmente os erros do modelo, foi gerado um gráfico de predições incorretas. A Figura 6.3 mostra alguns dos erros entre classes. Para uma visualização completa de um número maior de erros obtidos no teste feito com o *dataset* de teste, consulte a Figura II.1 nos Anexos, que contêm a totalidade das predições incorretas.

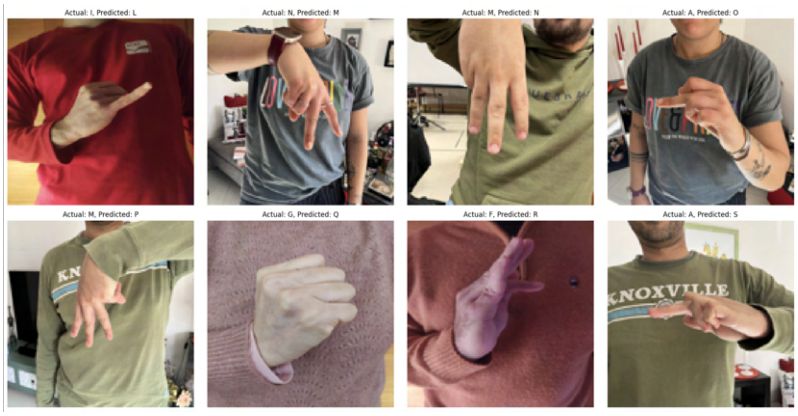


Figura 6.3: *Imagem com predições incorretas.*

No geral, o modelo CNN mostrou-se uma solução adequada para o reconhecimento de letras da LGP em dispositivos móveis, mas apresentou limitações evidentes em gestos mais difíceis de diferenciar. O *overfitting* foi um dos principais desafios, e a dificuldade do modelo em generalizar para novas instâncias sugere que melhorias podem ser alcançadas

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| A | 0.86 | 0.86 | 0.86 | 108 |
| B | 0.84 | 0.84 | 0.84 | 122 |
| C | 0.82 | 0.86 | 0.84 | 105 |
| D | 0.87 | 0.80 | 0.83 | 94 |
| E | 1.00 | 0.77 | 0.87 | 79 |
| F | 0.91 | 0.53 | 0.67 | 75 |
| G | 0.86 | 0.36 | 0.51 | 83 |
| H | 0.59 | 0.80 | 0.68 | 80 |
| I | 0.86 | 0.65 | 0.74 | 85 |
| J | 0.72 | 0.75 | 0.74 | 76 |
| K | 0.96 | 0.57 | 0.71 | 83 |
| L | 0.62 | 0.79 | 0.70 | 86 |
| M | 0.92 | 0.67 | 0.77 | 82 |
| N | 0.73 | 0.82 | 0.77 | 87 |
| O | 0.92 | 0.83 | 0.87 | 92 |
| P | 0.90 | 0.70 | 0.79 | 87 |
| Q | 0.93 | 0.85 | 0.89 | 96 |
| R | 0.33 | 0.78 | 0.46 | 89 |
| S | 0.64 | 0.72 | 0.68 | 100 |
| T | 0.80 | 0.76 | 0.78 | 91 |
| U | 0.76 | 0.73 | 0.75 | 94 |
| V | 0.76 | 0.73 | 0.75 | 97 |
| W | 0.79 | 0.87 | 0.83 | 84 |
| X | 0.66 | 0.53 | 0.59 | 111 |
| Y | 0.88 | 0.95 | 0.91 | 111 |
| Z | 0.71 | 0.75 | 0.73 | 100 |
| nothing | 0.80 | 0.96 | 0.87 | 123 |
| **Accuracy** | | | 0.76 | 2520 |
| **Macro avg** | 0.79 | 0.75 | 0.76 | 2520 |
| **Weighted avg** | 0.79 | 0.76 | 0.76 | 2520 |

Tabela 6.1: Relatório de classificação do modelo CNN, com precisão, recall e f1-score para cada classe.

com um conjunto de dados mais diversificado, um maior número de exemplos para as classes problemáticas e ajustes adicionais nos pesos das classes. Além disso, é possível que fatores como a dimensão das imagens, a escolha da arquitetura da rede e as características do modelo pré-treinado tenham influenciado o desempenho e a capacidade de generalização, especialmente em classes mais complexas.

### 6.1.2 Modelo 2: MediaPipe para Deteção de Mãos

O segundo modelo foi implementado utilizando a framework MediaPipe para a deteção dos pontos de referência das mãos, seguido de uma rede MLP para a classificação das letras. Este modelo trouxe uma abordagem diferente, focada na extração geométrica das mãos, o que permitiu captar informações essenciais sobre a posição dos dedos. O treino do modelo foi realizado ao longo de 22 épocas, e as curvas de *accuracy* e *loss* apresentaram um comportamento bastante promissor. Desde as primeiras épocas, o modelo conseguiu

melhorar a exatidão (*accuracy*) no conjunto de validação, atingindo um valor final de 93.17%. Ao contrário do que aconteceu com o primeiro modelo baseado em CNN, o MediaPipe conseguiu manter uma *accuracy* de validação mais consistente e elevada, demonstrando uma capacidade superior de generalização para novos dados, como se pode observar na Figura 6.4. Embora o *overfitting* tenha sido uma preocupação durante o treino, o uso de técnicas de regularização e a redução automática da taxa de aprendizagem permitiram mitigar esse problema, resultando numa menor divergência entre o conjunto de treino e validação.

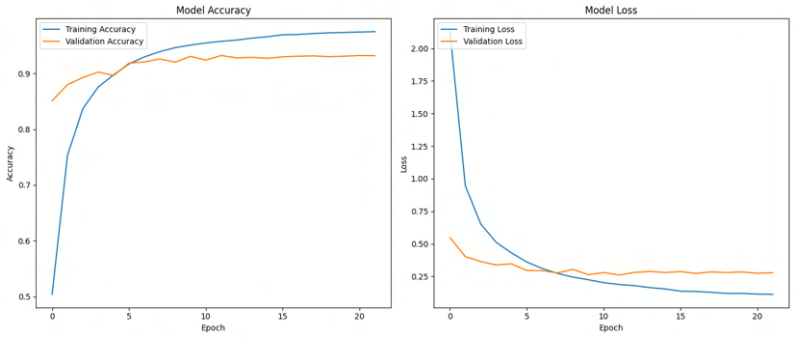


Figura 6.4: *Curva de Accuracy e Loss*

Tal como no modelo anterior, uma análise mais detalhada foi realizada utilizando a matriz de confusão, mostrada na Figura 6.5. Esta matriz permitiu observar de forma clara como o modelo se comportou em cada classe de letras. O MediaPipe demonstrou um bom desempenho em letras como "A", "B" e "O", onde a maioria das predições foi correta. No entanto, também houve dificuldades com classes visualmente mais semelhantes, tal como no modelo CNN. As letras "X" e "K", por exemplo, mostraram-se particularmente difíceis para o modelo, sendo confundidas com gestos como "V" e "P". Este comportamento pode ser atribuído à complexidade visual dos gestos e à proximidade dos pontos de referência das mãos nestes gestos, o que pode ter levado o modelo a cometer erros de predição. Ainda assim, o MediaPipe teve uma maior precisão em classes complexas do que o modelo CNN, beneficiando da extração geométrica dos pontos de referência, que forneceu uma representação mais detalhada das poses das mãos.

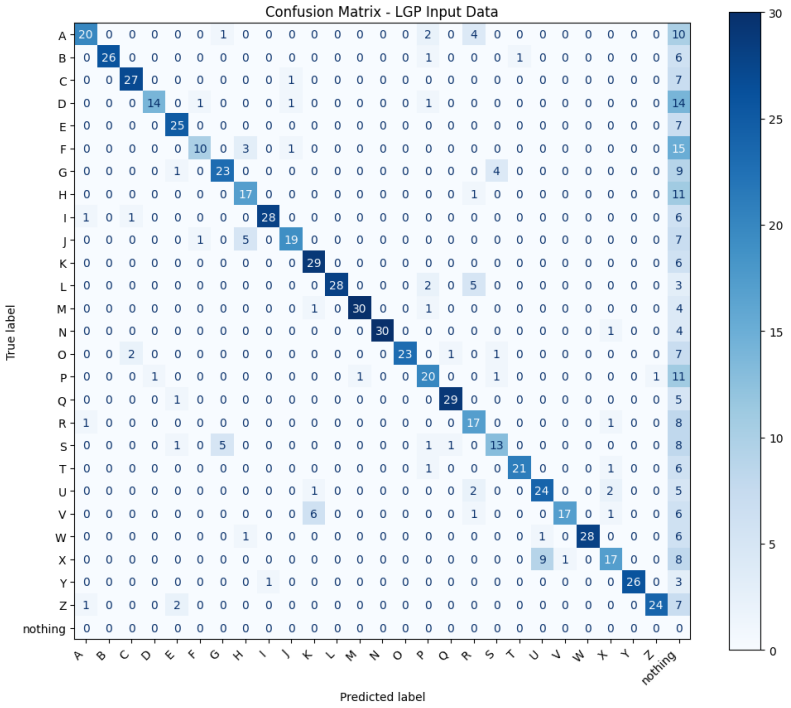


Figura 6.5: *Matriz de Confusão*

É importante mencionar que a classe "nothing" foi frequentemente *predicted* nas situações em que o MediaPipe não conseguiu detetar corretamente os pontos de referência (landmarks) das mãos nas imagens. A predição desta classe ocorre sempre que o MediaPipe falha ao identificar a mão nas imagens, seja devido à má qualidade da imagem, ângulos desfavoráveis ou condições de iluminação inadequadas. Este fator influenciou a matriz de confusão, pois, em várias instâncias, o modelo não conseguiu identificar corretamente os landmarks da mão, resultando numa previsão "nothing". Assim, essa classe não deve ser interpretada como uma classificação equivocada do gesto, mas sim como uma indicação de falha na deteção inicial das *landmarks*.

Os resultados detalhados para cada classe foram analisados no relatório de classificação (Tabela 6.2).

O modelo MediaPipe apresentou uma precisão média de 77% e um *f1-score* global de 77%. Estes números indicam que o modelo conseguiu manter uma relação relativamente equilibrada entre a precisão, que mede a proporção de predições corretas em relação ao total de predições positivas, e o *recall*, que avalia a capacidade do modelo de identificar corretamente todas as instâncias positivas. Esse equilíbrio reflete a capacidade do modelo de minimizar falsos positivos e falsos negativos de forma consistente em muitas classes, mostrando-se mais robusto do que o modelo CNN em várias categorias. Por exemplo, a classe "B" teve uma precisão de 96% e um *f1-score* de 88%, demonstrando o bom desempenho do modelo em identificar corretamente esta letra na maioria das instâncias. Contudo, classes como "X" e "K" continuaram a apresentar dificuldades, com *f1-scores* de 0.52 e 56%, respetivamente, o que revela desafios na diferenciação de gestos com configurações visuais semelhantes.

| Class | Precision | Recall | F1-Score | Support |
|---|---|---|---|---|
| A | 0.87 | 0.73 | 0.79 | 108 |
| B | 0.96 | 0.81 | 0.88 | 122 |
| C | 0.98 | 0.81 | 0.89 | 105 |
| D | 0.81 | 0.40 | 0.54 | 94 |
| E | 0.91 | 0.80 | 0.85 | 79 |
| F | 0.92 | 0.44 | 0.59 | 75 |
| G | 0.80 | 0.67 | 0.73 | 83 |
| H | 0.72 | 0.55 | 0.62 | 80 |
| I | 0.96 | 0.87 | 0.91 | 85 |
| J | 0.69 | 0.79 | 0.74 | 76 |
| K | 0.73 | 0.46 | 0.56 | 83 |
| L | 0.99 | 0.77 | 0.86 | 86 |
| M | 0.96 | 0.85 | 0.90 | 82 |
| N | 0.84 | 0.74 | 0.79 | 87 |
| O | 0.97 | 0.91 | 0.94 | 92 |
| P | 0.78 | 0.48 | 0.60 | 87 |
| Q | 0.95 | 0.82 | 0.88 | 96 |
| R | 0.71 | 0.61 | 0.65 | 89 |
| S | 0.74 | 0.66 | 0.70 | 100 |
| T | 0.96 | 0.86 | 0.91 | 91 |
| U | 0.81 | 0.63 | 0.71 | 94 |
| V | 0.74 | 0.53 | 0.61 | 97 |
| W | 0.84 | 0.61 | 0.70 | 84 |
| X | 0.54 | 0.50 | 0.52 | 111 |
| Y | 1.00 | 0.90 | 0.95 | 111 |
| Z | 0.99 | 0.83 | 0.90 | 100 |

Tabela 6.2: Relatório de Classificação - Precisão, Recall e F1-Score para cada classe.

Tal como já foi mencionado, as classes “X” e “K” foram as mais problemáticas, sendo frequentemente confundidas com outras letras. A confusão entre elas, por exemplo, foi um erro recorrente, refletindo a semelhança entre os gestos dessas letras. Este tipo de erro sugere que, embora o modelo baseado em MediaPipe tenha uma maior capacidade de extrair detalhes geométricos das mãos, este ainda enfrenta dificuldades em diferenciar gestos com pequenas variações nos posicionamentos dos dedos. Este problema pode ser abordado no futuro com o cálculo de novas *features* ou o aumento do conjunto de treino para as classes mais problemáticas.

Para ilustrar visualmente os erros do modelo, foram gerados dois gráficos de predições incorretas. O primeiro gráfico mostra erros que incluem a predição de “nothing”, enquanto o segundo exclui essa classe e foca nos erros de predição entre gestos reais.

A Figura 6.6 mostra alguns dos erros com a predição “nothing”, e a Figura 6.7 mostra alguns dos erros entre classes reais. Para uma visualização completa de todos os erros obtidos no teste feito com o dataset de teste, consulte as imagens II.2 à II.7 nos anexos, que contêm a totalidade das predições incorretas.

Em termos gerais, o modelo MediaPipe mostrou-se uma solução robusta para o reconhecimento de gestos na LGP, superando o modelo CNN em várias métricas, especialmente

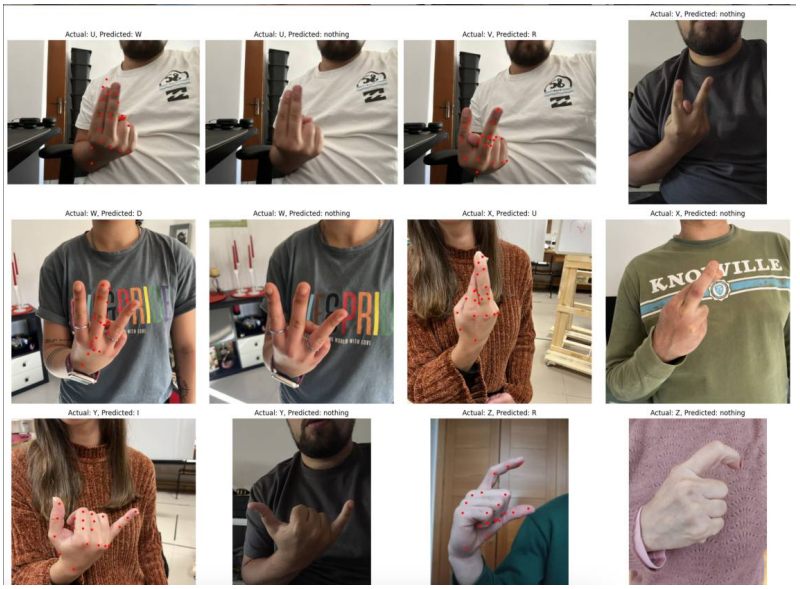


Figura 6.6: *Imagem com predições incorretas, incluindo a predição "nothing".*

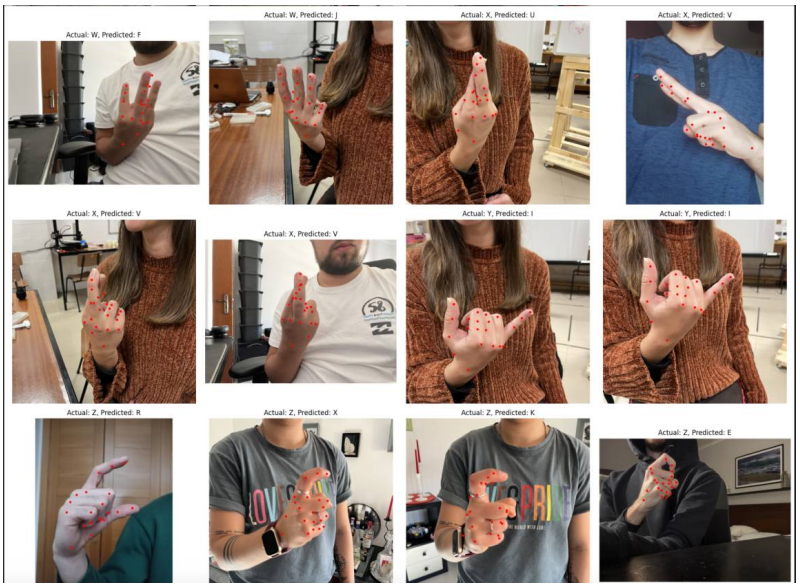


Figura 6.7: *Imagem com predições incorretas, excluindo a predição "nothing".*

no que diz respeito à capacidade de generalização para novos dados. A precisão global foi ligeiramente superior, e o modelo teve um desempenho mais consistente entre diferentes classes. No entanto, tal como no modelo CNN, houve dificuldades significativas em letras com gestos visualmente semelhantes. Para mitigar esses problemas, poderiam ser exploradas técnicas adicionais de *data augmentation*, como alterações de perspetiva, adição de ruído gaussiano, simulação de desfocagem, ou variações de cores, que permitiriam ao modelo lidar melhor com situações mais diversificadas e condições de captura adversas.

### 6.1.3 Análise Comparativa

A Tabela 6.3 oferece uma comparação entre os dois modelos. Ambos apresentaram desempenhos semelhantes em termos de precisão e *f1-score*. No entanto, o modelo MediaPipe mostrou-se mais robusto em gestos mais complexos, enquanto o modelo CNN ofereceu

tempos de resposta mais rápidos devido à execução local. Em problemas de classificação multiclasse, falsos positivos correspondem a casos onde o modelo atribui incorretamente um gesto a uma classe errada, enquanto falsos negativos ocorrem quando o modelo falha ao identificar corretamente um gesto da classe alvo. Por exemplo, um falso positivo para a classe 'X' ocorre quando um gesto de "V" é identificado como "X", enquanto um falso negativo acontece quando um gesto de 'X' é classificado como outra letra, como "P" ou "K". Estes erros ajudam a compreender os desafios enfrentados por cada modelo ao classificar gestos visualmente semelhantes.

| Modelo | Accuracy | Precision Média | Recall Médio | F1-Score Médio |
|---|---|---|---|---|
| CNN | 76% | 79% | 75% | 76% |
| MediaPipe (via API) | 77% | 78% | 76% | 77% |

Tabela 6.3: Comparação dos resultados de classificação entre o Modelo CNN e o Modelo MediaPipe.

Como se observou anteriormente, ambos os modelos enfrentaram desafios com as classes "X", "G", e "R", por exemplo. A semelhança visual entre certos gestos resultou numa alta taxa de falsos positivos e negativos. Para mitigar estes problemas, serão necessários ajustes nos pesos de classe ou o aumento de exemplos de treino para essas classes.

### 6.1.4 Comparação de Resultados com Trabalhos Relacionados

Ao comparar os resultados com trabalhos anteriores, como o de Patrícia Reis Ribeiro [28] e Oliveira [23], verifica-se que o uso de CNNs e do MediaPipe oferece uma abordagem mais acessível e prática do que soluções baseadas em hardware especializado, como o Kinect. A combinação de precisão, portabilidade e flexibilidade dos nossos modelos destaca-se como uma evolução na área do reconhecimento de gestos para a LGP. O uso de técnicas de *machine learning* aplicadas diretamente em dispositivos móveis ou via servidores remotos traz uma vantagem significativa sobre sistemas que dependem de hardware volumoso e caro, facilitando a utilização em ambientes do quotidiano e permitindo maior escalabilidade.

Adicionalmente, o trabalho de Kazuhito [33], que implementa reconhecimento de gestos utilizando o *MediaPipe* combinado com um MLP, também serve como uma base de comparação importante. Kazuhito concentrou-se na utilização de uma solução que dependia exclusivamente das coordenadas 2D fornecidas pelo *MediaPipe*, o que demonstrou ser eficaz para gestos simples. No entanto, a nossa abordagem vai além ao utilizar dois modelos complementares que funcionam de forma independente: o primeiro modelo é baseado no MediaPipe, em que os pontos-chave (*landmarks*) das mãos são processados por um MLP, complementado pelo cálculo de ângulos e distâncias entre os *landmarks*, para capturar variações subtis na configuração dos dedos. O segundo modelo é uma CNN, que trabalha diretamente com as imagens capturadas, extraindo características visuais a partir dos gestos realizados. Embora estes modelos atuem de forma independente, a sua implementação lado a lado oferece flexibilidade: o MediaPipe com o MLP é utilizado em situações que requerem maior eficiência e precisão na deteção das posições detalhadas das mãos, enquanto o modelo de CNN é otimizado para trabalhar diretamente com imagens, tornando-o mais

adequado para cenários onde as coordenadas dos *landmarks* não são suficientes.

## 6.2 Avaliação da Aplicação

A avaliação da aplicação desenvolvida para o reconhecimento de gestos da LGP foi realizada com base em dois aspetos principais: o desempenho técnico dos modelos integrados na aplicação e a experiência do utilizador. A aplicação, construída em Flutter para assegurar compatibilidade com dispositivos Android e iOS, teve como objetivo proporcionar uma solução prática e funcional, permitindo o reconhecimento de gestos capturados pela câmara e exibindo os resultados com uma latência entre 1 e 5 segundos, dependendo do dispositivo e do modelo utilizado. A aplicação foi testada em diferentes cenários, incluindo variações de iluminação, ângulos de captura e dispositivos de diferentes gamas, para garantir a sua robustez e adaptabilidade em situações do mundo real.

### 6.2.1 Desempenho Técnico e Integração dos Modelos

A integração dos dois modelos de reconhecimento de gestos, o *MobileNetV2* convertido para TensorFlow Lite (tflite) e o MediaPipe via servidor Flask, apresentou resultados distintos, dependendo do ambiente e das condições de utilização. O modelo *tflite*, executado localmente no dispositivo móvel, destacou-se pela sua baixa latência, com tempos de resposta entre 1 e 2 segundos em dispositivos de maior capacidade, proporcionando uma interação que se aproxima do tempo real. A sua capacidade de funcionar *offline* foi um dos principais pontos fortes, eliminando a dependência de uma conexão constante à internet, o que assegurou uma experiência fluida e uma aplicação responsiva – ou seja, capaz de responder rapidamente às ações do utilizador, exibindo os resultados das predições sem atrasos significativos na interface.

No entanto, embora a baixa latência tenha sido uma vantagem significativa, os resultados do modelo *tflite* em termos de precisão e desempenho geral não foram os melhores. Mesmo em dispositivos de maior capacidade, como o iPhone, observou-se uma precisão inferior em gestos mais complexos, onde o modelo frequentemente cometia erros de predição. Esta limitação foi ainda mais pronunciada quando o modelo foi testado em dispositivos de gama inferior. Nesses casos, a menor capacidade de processamento resultou não só em predições mais lentas, como também numa taxa de erro maior, demonstrando que o modelo *tflite* enfrenta dificuldades em lidar com dispositivos menos poderosos. Além dessas limitações de hardware, o modelo CNN em si poderia ser melhorado para otimizar o reconhecimento de gestos mais complexos. Outra fonte de dificuldade foi a conversão do modelo para *tflite*, que introduziu algumas perdas de capacidade no reconhecimento. O processo de conversão para *TensorFlow Lite* simplifica certas operações e reduz a complexidade do modelo para torná-lo mais eficiente em dispositivos móveis, mas essa simplificação comprometeu a precisão em gestos subtis ou com maior sobreposição de dedos, impactando negativamente a qualidade das predições. Assim, a combinação da conversão e das limitações do próprio modelo resultou numa perda de desempenho em comparação com a versão original não convertida.

Por outro lado, o modelo MediaPipe, apesar de ser mais exigente em termos de conectividade, revelou-se uma solução mais robusta e precisa para o reconhecimento de gestos, particularmente para gestos mais complexos. A comunicação com o servidor Flask, que processa as imagens e retorna as predições, permitiu ao modelo beneficiar de uma maior capacidade de processamento e, como resultado, fornecer predições mais fiáveis e consistentes, mesmo em gestos visualmente semelhantes. Contudo, a dependência de uma boa conexão à internet introduziu variações significativas no tempo de resposta, com latências notáveis em redes mais lentas ou instáveis. Ainda assim, em cenários onde a precisão é mais importante que a latência, o modelo MediaPipe mostrou-se uma solução viável e mais robusta do que o modelo *tflite*, oferecendo uma experiência mais satisfatória em termos de precisão e reconhecimento de gestos complexos.

Em suma, embora o modelo *tflite* tenha demonstrado vantagens em termos de latência e capacidade de operação *offline*, os seus resultados de precisão deixaram a desejar, especialmente em dispositivos de menor gama. Já o modelo MediaPipe mostrou-se superior em termos de precisão, mas com a limitação de depender de uma conexão à internet para fornecer predições consistentes.

### 6.2.2 Experiência do Utilizador

Para avaliar a usabilidade da aplicação, foram realizadas entrevistas estruturadas com 10 utilizadores, utilizando dispositivos móveis com diferentes capacidades de processamento. O objetivo principal foi analisar a facilidade de navegação, a intuição dos menus, e a fluidez na interação com a aplicação. Os utilizadores avaliaram a simplicidade da interface e a eficácia das funcionalidades oferecidas.

A grande maioria dos participantes destacou a facilidade de utilização da aplicação e a simplicidade na navegação entre as funcionalidades. Todos os 10 utilizadores relataram que a alternância entre os modelos de reconhecimento de gestos (*TensorFlow Lite* e MediaPipe) era uma funcionalidade útil, permitindo adaptar o uso da aplicação às condições disponíveis, como a presença ou ausência de conexão à internet.

Contudo, alguns pontos de melhoria foram mencionados. Aproximadamente 70% dos utilizadores (6 em 10) notaram que a aplicação não fornecia *feedback* adequado quando um gesto não era reconhecido, o que gerava alguma confusão. Sugeriram que, em casos onde o reconhecimento falhasse, a aplicação deveria exibir uma mensagem clara, sugerindo possíveis correções, como ajustes na posição da mão ou mudanças na iluminação.

Além disso, cerca de 20% dos utilizadores relataram dificuldades na identificação de alguns ícones no menu de navegação. Estes utilizadores sugeriram que o uso de descrições textuais ou ícones mais intuitivos poderia melhorar ainda mais a experiência de navegação. Adicionalmente, alguns participantes mencionaram que a visualização dos gestos na página de letras poderia ser melhorada, com um design mais claro e sem sombras, especialmente para ecrãs com menor resolução.

### 6.2.3 Testes em Condições Reais

Para avaliar a robustez da aplicação em diferentes cenários, foram realizados vários testes com variações de iluminação, ângulos de captura e ruído visual, utilizando dispositivos de gama alta e de gama média/baixa. Estes testes permitiram observar o comportamento dos dois modelos implementados – o *CNN* em *tflite* e o MediaPipe com MLP – em condições mais próximas do uso real.

No caso do modelo *CNN*, executado localmente, o tempo de predição variou significativamente entre dispositivos de maior e menor capacidade. Em dispositivos de gama alta, o tempo médio de predição foi de aproximadamente 1 a 2 segundos, proporcionando uma interação quase imediata e uma experiência fluida para o utilizador. Em dispositivos de gama média ou baixa, no entanto, o tempo de resposta aumentou para 3 a 5 segundos, refletindo as limitações de processamento desses dispositivos. Além disso, a qualidade inferior das câmaras em dispositivos de baixa gama também influenciou os resultados, contribuindo para uma maior taxa de erro e uma diminuição na precisão, particularmente em gestos mais complexos como as letras "G", "X"e "P". Em gestos mais simples, como "B", "C"e "L", a taxa de acerto foi consideravelmente melhor, situando-se entre 50% e 60%.

Por outro lado, o modelo MediaPipe, que depende de uma conexão à internet, apresentou tempos de resposta mais consistentes entre diferentes dispositivos quando testado com a mesma velocidade de rede. O tempo médio de predição variou entre 3 a 5 segundos em ambos os casos, independentemente da capacidade de processamento do dispositivo. No entanto, os resultados ainda variaram em dispositivos de baixa gama, possivelmente devido à qualidade inferior das câmaras, que impacta a capacidade do MediaPipe em detetar os pontos-chave (*landmarks*) com precisão. Apesar da latência adicional causada pela dependência de uma conexão à internet, o modelo MediaPipe demonstrou maior robustez em condições de iluminação variada e na presença de gestos subtis, alcançando uma taxa de acerto geral entre 70% e 80%, mesmo para gestos mais complexos.

Assim, enquanto o modelo *CNN* destacou-se pela rapidez nas predições em dispositivos de maior capacidade, o modelo MediaPipe provou ser mais confiável em termos de precisão, embora com um tempo de resposta mais elevado. Ambas as abordagens apresentaram vantagens e limitações, dependendo das condições de utilização e do tipo de dispositivo.

### 6.2.4 Conclusão da Avaliação

No geral, a avaliação da aplicação foi positiva, tanto em termos de desempenho técnico quanto de experiência do utilizador. A alternância entre os dois modelos ofereceu flexibilidade e adaptabilidade, permitindo que a aplicação fosse utilizada em diferentes cenários e condições de rede. Enquanto o *tflite* se destacou pela rapidez e capacidade de operação *offline*, o MediaPipe mostrou-se mais robusto em termos de precisão, especialmente em gestos mais complexos. As sugestões de melhoria dos utilizadores, como a introdução de feedback mais claro em caso de erro e ajustes na navegação, serão cruciais para futuras iterações da aplicação, tornando-a ainda mais intuitiva e eficaz na tradução de gestos da LGP para texto.

# 7 Conclusão e Trabalhos Futuros

Este capítulo resume as principais conclusões do trabalho realizado, destacando os resultados obtidos no desenvolvimento dos modelos de reconhecimento de gestos da LGP e a aplicação móvel associada. Além disso, são discutidas as implicações e limitações do projeto, seguidas de sugestões para trabalhos futuros e melhorias no sistema.

## 7.1 Resumo das Conclusões

O desenvolvimento de uma aplicação móvel para a tradução das letras da LGP para texto envolveu a utilização de duas abordagens distintas: um modelo baseado em CNN (*MobileNetV2*), convertido para *TensorFlow Lite*, e uma solução baseada no MediaPipe. Ambos os modelos foram integrados numa aplicação desenvolvida em *Flutter*, permitindo uma interface acessível para o reconhecimento de gestos. A flexibilidade da aplicação, que opera em modo *offline* com o modelo *tflite* e *online* com o MediaPipe, é um dos seus pontos mais fortes.

Os testes realizados mostraram que o modelo *tflite* CNN se destacou pela sua baixa latência e capacidade de operar sem dependência de internet, sendo ideal para dispositivos com maior capacidade de processamento. No entanto, a precisão diminuiu significativamente em gestos mais complexos, especialmente em dispositivos de gama inferior. Em contrapartida, o MediaPipe, processado remotamente via servidor Flask, apresentou uma precisão superior, sendo mais robusto em gestos difíceis de identifica, mas a sua dependência de uma conexão estável à internet introduziu uma latência variável.

Ambos os modelos enfrentaram dificuldades na distinção de gestos visualmente semelhantes, sendo necessárias melhorias no reconhecimento de classes como "X" e "G". O uso de técnicas de ajuste, como a ponderação das classes e a adição de *features* geométricas, ajudou a melhorar o desempenho, mas ainda existe espaço para otimização.

## 7.2 Implicações

A implementação de modelos híbridos de reconhecimento de gestos, combinando as vantagens da execução *offline* com o *tflite* e o processamento remoto do MediaPipe, tem

implicações importantes no desenvolvimento de soluções tecnológicas para a LGP. Esta abordagem permite que a aplicação seja flexível, adaptando-se às condições do dispositivo e da conectividade à internet, o que é crucial em cenários de uso real.

Além disso, a aplicação móvel desenvolvida serve como uma ferramenta potencial para a educação e inclusão de utilizadores surdos e ouvintes, facilitando a aprendizagem e a comunicação através da LGP. A capacidade de reconhecer gestos em tempo real, mesmo em dispositivos com recursos limitados, abre novas oportunidades para o uso da tecnologia em contextos educativos e de acessibilidade.

## 7.3 Limitações

Apesar dos avanços alcançados, o projeto ainda enfrenta algumas limitações. A principal limitação do modelo *tflite* reside na dependência da capacidade de hardware do dispositivo. Em dispositivos de gama inferior, o desempenho foi afetado, resultando em tempos de processamento mais longos e numa maior taxa de erro nas predições. Essa variação de desempenho entre diferentes dispositivos pode comprometer a consistência da experiência do utilizador.

Além disso, o modelo MediaPipe mostrou-se dependente de uma conexão estável à internet, o que pode ser um entrave em situações com conectividade limitada. Outra limitação comum a ambos os modelos foi a dificuldade em reconhecer letras com gestos semelhantes, o que sugere que ainda são necessários mais dados de treino e estratégias de reconhecimento mais avançadas para aumentar a precisão nestas classes.

Por fim, a qualidade das predições dependeu da qualidade das imagens capturadas. Variáveis como a iluminação, ângulos da mão e ruído visual no ambiente influenciaram a eficácia dos modelos, indicando que melhorias no pré-processamento das imagens ou o uso de algoritmos de correção de imagem podem ser benéficos.

## 7.4 Trabalhos Futuros

Tendo em conta os desafios identificados, há várias direções promissoras para trabalhos futuros que podem melhorar e expandir a solução desenvolvida. Entre as possíveis melhorias estão:

- **Tradução de Gestos Complexos**: A próxima evolução natural seria expandir a aplicação para reconhecer não apenas letras isoladas, mas também palavras e frases completas em LGP. Esta abordagem exigiria modelos mais complexos e que pudessem capturar gestos contínuos, incluindo a segmentação automática entre gestos.

- **Suporte a Outras Línguas Gestuais**: A expansão da aplicação para suportar outras línguas gestuais, como a ASL ou a Língua Gestual Brasileira (Libras), permitiria alargar o alcance da solução, tornando-a útil para um público mais vasto e em contextos multiculturais.

- **Melhoria no Reconhecimento de Classes Semelhantes**: Uma melhoria importante seria desenvolver novas estratégias para melhorar a discriminação entre gestos semelhantes. Isso poderia ser feito através do aumento do conjunto de treino para as classes mais desafiantes ou da aplicação de novos algoritmos de aprendizagem, como técnicas de *context-aware recognition*, que podem ajudar o modelo a entender o contexto em que um gesto é realizado.
- **Incorporação de *Feedback* dos Utilizadores**: Os utilizadores que participaram nos testes sugeriram melhorias na interface e no design, tais como tornar o menu de navegação mais intuitivo e fornecer um *feedback* mais explícito quando um gesto não é reconhecido. A integração deste *feedback* nos trabalhos futuros contribuiria para uma aplicação mais amigável e eficaz.

## 7.5 Considerações Finais

Este projeto representa um avanço importante na utilização de tecnologias de reconhecimento de gestos para facilitar a comunicação entre pessoas surdas e ouvintes. A aplicação desenvolvida combina diferentes abordagens tecnológicas e mostra-se promissora em termos de flexibilidade e acessibilidade. A integração de modelos *offline* e *online* demonstrou ser eficaz, garantindo que a aplicação se adapte a diferentes condições de uso e capacidade de dispositivos.

Embora existam áreas de melhoria, como o reconhecimento de gestos complexos (aqueles que envolvem posições subtis dos dedos, sobreposição de mãos ou movimentos específicos) e a otimização para dispositivos de menor gama, a solução apresentada oferece uma base sólida para futuras expansões. Com o desenvolvimento contínuo, incluindo a tradução de gestos mais complexos e o suporte a outras línguas gestuais, esta aplicação tem o potencial de se tornar uma ferramenta essencial na inclusão da comunidade surda e na educação da LGP.

A continuidade deste trabalho, com foco na melhoria da precisão e na adaptação às necessidades dos utilizadores, poderá consolidar a aplicação como uma solução robusta e acessível para a comunicação através de gestos.

# Bibliografia


[1] *All You Need to Know About Support Vector Machines*. Spiceworks Inc. URL: https://www.spiceworks.com/tech/big-data/articles/what-is-support-vector-machine/ (acedido em 10/09/2024) (ver p. 13).

[2] M. A. Amaral, A. Coutinho e M. R. Delgado Martins. *Para uma gramatica da lingua gestual portuguesa*. Coleccao universitaria. Lisboa: Cahinho, 1994. 166 pp. ISBN: 9789722109819 (ver p. 10).

[3] Ayush. *A Comprehensive Guide on Optimizers in Deep Learning*. Analytics Vidhya. 7 de out. de 2021. URL: https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/ (acedido em 03/10/2024) (ver p. 15).

[4] *Basic CNN Architecture: Explaining 5 Layers of Convolutional Neural Network | upGrad blog*. URL: https://www.upgrad.com/blog/basic-cnn-architecture/ (acedido em 10/09/2024) (ver p. 12).

[5] C. Cortes e V. Vapnik. "Support-vector networks". Em: *Machine Learning* 20.3 (1 de set. de 1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018. URL: https://doi.org/10.1007/BF00994018 (acedido em 28/09/2024) (ver p. 13).

[6] Dharmaraj. *Convolutional Neural Networks (CNN) — Architectures Explained*. Medium. 1 de jun. de 2022. URL: https://medium.com/@draj0718/convolutional-neural-networks-cnn-architectures-explained-716fb197b243 (acedido em 02/01/2025) (ver p. 12).

[7] *Experiment@ Língua Gestual Portuguesa - Portal da Juventude de Vila Nova de Famalicão*. URL: http://www.juventudefamalicao.org/_experiment_lingua_gestual_portuguesa_2 (acedido em 05/05/2024) (ver p. 10).

[8] *Figure 2. Multi-layer perceptron (MLP-NN) basic Architecture.* ResearchGate. URL: https://www.researchgate.net/figure/Multi-layer-perceptron-MLP-NN-basic-Architecture_fig2_354817375 (acedido em 28/09/2024) (ver p. 12).

[9] GAF. *Sabia que a Língua Gestual Portuguesa (LGP) é uma das 3 Línguas Oficiais em Portugal*. URL: https://www.gaf.pt/pt/noticias/sabia-que-a-lingua-gestual-portuguesa-lgp-e-uma-das-3-linguas-oficiais-em-portugal (acedido em 28/09/2024) (ver p. 9).

[10] K. S. Ganesh. *What's The Role Of Weights And Bias In a Neural Network?* Medium. 7 de set. de 2022. URL: https://towardsdatascience.com/whats-the-role-of-weights-and-bias-in-a-neural-network-4cf7e9888a0f (acedido em 27/12/2024) (ver p. 15).

[11] *google/flutter-mediapipe*. original-date: 2023-04-26T00:39:16Z. 29 de dez. de 2024. URL: https://github.com/google/flutter-mediapipe (acedido em 02/01/2025) (ver p. 20).

[12] T. Goswami e S. R. Javaji. "CNN Model for American Sign Language Recognition". Em: 1 de out. de 2020, pp. 55–61. ISBN: 9789811579608. DOI: 10.1007/978-981-15-7961-5_6 (ver p. 17).

[13] H. He e E. A. Garcia. "Learning from Imbalanced Data". Em: *IEEE Transactions on Knowledge and Data Engineering* 21.9 (set. de 2009), pp. 1263–1284. ISSN: 1558-2191. DOI: 10.1109/TKDE.2008.239. URL: https://ieeexplore.ieee.org/document/5128907?denied= (acedido em 28/09/2024) (ver p. 35).

[14] E. S. Klima e U. Bellugi. *The Signs of Language*. Google-Books-ID: WeBOn6N8PJ8C. Harvard University Press, 1979. 436 pp. ISBN: 9780674807969 (ver p. 7).

[15] H. Lane, R. Hoffmeister e B. J. Bahan. *A journey into the deaf-world*. San Diego, Calif: DawnSignPress, 1996. 512 pp. ISBN: 9780915035625 9780915035632 (ver p. 7).

[16] *layout: forward target: https://developers.google.com/mediapipe/solutions/vision/hand_landmarker title: Hands parent: MediaPipe Legacy Solutions nav_order: 4 — MediaPipe v0.7.5 documentation*. URL: https://mediapipe.readthedocs.io/en/latest/solutions/hands.html (acedido em 17/09/2024) (ver pp. 17, 37).

[17] C. Lucas, C. Valli e R. Bayley. "Sociolinguistic Variation in American Sign Language". Em: *Bibliovault OAI Repository, the University of Chicago Press* (1 de jan. de 2001). DOI: 10.1017/CBO9780511612824 (ver p. 7).

[18] *Língua Portuguesa :: Gestualizando*. URL: https://gestualizando.webnode.pt/varias-linguas-gestuais/lingua-portuguesa/ (acedido em 06/05/2024) (ver p. 10).

[19] *Machine Learning: Decision Tree Classifier | by Michele Cavaioni | Machine Learning bites | Medium*. URL: https://medium.com/machine-learning-bites/machine-learning-decision-tree-classifier-9eb67cad263e (acedido em 10/09/2024) (ver p. 14).

[20] C. Morgado, A. M. Brito e F. d. Letras. *Língua gestual portuguesa e outras línguas de sinais : estudos linguísticos*. 2022. ISBN: 9789899082021. DOI: 10.21747/978-989-9082-02-1/ling. URL: https://repositorio-aberto.up.pt/handle/10216/140766 (acedido em 17/09/2024) (ver p. 9).

[21] *Mãos que falam: hoje é Dia Nacional da Língua Gestual Portuguesa (com vídeo)*. URL: https://www.portugal.gov.pt/pt/gc23/comunicacao/noticia?i=maos-que-falam-hoje-e-dia-nacional-da-lingua-gestual-portuguesa (acedido em 05/05/2024) (ver p. 9).

[22] I. G. S. Neiva. "Desenvolvimento de um tradutor de Língua Gestual Portuguesa". Tese de doutoramento. Faculdade de Ciências e Tecnologia da Universidade Nova de Lisboa (ver p. 16).

[23] O. R. S. A. Oliveira. "Tradutor da língua gestual portuguesa modelo de tradução bidireccional". masterThesis. 2013. URL: https://recipp.ipp.pt/handle/10400.22/6246 (acedido em 17/09/2024) (ver pp. 16, 55).

[24] *Papers with Code - ImageNet Dataset*. URL: https://paperswithcode.com/dataset/imagenet (acedido em 02/01/2025) (ver p. 14).

[25] R. Pfau, M. Steinbach e B. Woll, eds. *Sign Language: An International Handbook*. DE GRUYTER, 16 de ago. de 2012. ISBN: 9783110204216. DOI: 10.1515/9783110261325. URL: https://www.degruyter.com/document/doi/10.1515/9783110261325/html (acedido em 17/09/2024) (ver p. 9).

[26] M. C. Pinto. *O que todos devíamos saber sobre língua gestual (em dez pontos)*. PÚBLICO. 14 de nov. de 2017. URL: https://www.publico.pt/2017/11/14/p3/noticia/o-que-todos-deviamos-saber-sobre-lingua-gestual-em-dez-pontos-1828846 (acedido em 17/09/2024) (ver p. 8).

[27] D. d. República. *Diário da República n.º 218/1997, Série I-A de 1997-09-20*. 6 de mai. de 2024. URL: https://diariodarepublica.pt/dr/detalhe/diario-republica/218-1997-107291 (acedido em 06/05/2024) (ver p. 9).

[28] P. R. Ribeiro. "Sistema de reconhecimento de Língua Gestual Portuguesa recorrendo à Kinect". masterThesis. 2019. URL: https://repositorium.sdum.uminho.pt/handle/1822/72165 (acedido em 17/09/2024) (ver pp. 16, 55).

[29] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov e L.-C. Chen. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. version: 4. 21 de mar. de 2019. DOI: 10.48550/arXiv.1801.04381. arXiv: 1801.04381[cs]. URL: http://arxiv.org/abs/1801.04381 (acedido em 18/09/2024) (ver p. 32).

[30] A. Singh. *shekit/alexa-sign-language-translator*. original-date: 2018-07-28T20:04:44Z. 16 de set. de 2024. URL: https://github.com/shekit/alexa-sign-language-translator (acedido em 17/09/2024) (ver p. 17).

[31] W. C. Stokoe Jr. "Sign Language Structure: An Outline of the Visual Communication Systems of the American Deaf". Em: *The Journal of Deaf Studies and Deaf Education* 10.1 (1 de jan. de 2005), pp. 3–37. ISSN: 1081-4159. DOI: 10.1093/deafed/eni001. URL: https://doi.org/10.1093/deafed/eni001 (acedido em 28/09/2024) (ver p. 7).

[32] *Synthetic ASL Alphabet*. URL: https://www.kaggle.com/datasets/lexset/synthetic-asl-alphabet (acedido em 05/01/2025) (ver p. 25).

[33] S. Takahashi. *hand-gesture-recognition-using-mediapipe*. original-date: 2020-12-15T13:26:23Z. Dez. de 2020. URL: https://github.com/Kazuhito00/hand-gesture-recognition-using-mediapipe (acedido em 17/09/2024) (ver pp. 17, 55).

[34] TensorFlow Team. *Post-training Quantization*. Accessed: 2024-10-10. 2024. URL: https://www.tensorflow.org/lite/performance/post_training_quantization (acedido em 10/10/2024) (ver p. 43).

[35] TensorFlow Team. *TensorFlow Lite Model Conversion*. Accessed: 2024-10-10. 2024. URL: https://www.tensorflow.org/lite/convert (acedido em 10/10/2024) (ver p. 43).

[36] *What is MediaPipe? A Guide for Beginners*. Roboflow Blog. 10 de abr. de 2024. URL: https://blog.roboflow.com/what-is-mediapipe/ (acedido em 17/09/2024) (ver p. 17).

[37] *What is transfer learning? | IBM*. 12 de fev. de 2024. URL: https://www.ibm.com/think/topics/transfer-learning (acedido em 27/12/2024) (ver p. 14).

# I Anexo 1 - Exemplos dos conjuntos do dataset

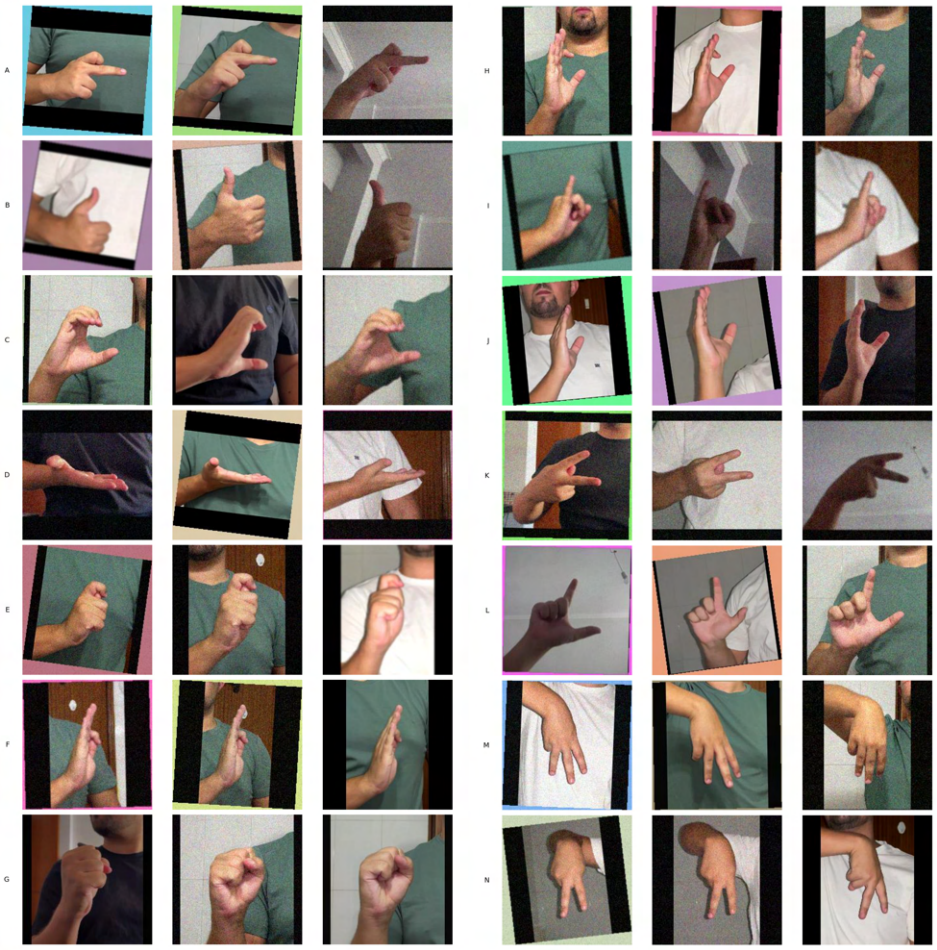

Figura I.1: *Exemplos de imagens de cada classe do conjunto de treino*


Figura I.2: *Exemplos de imagens de cada classe do conjunto de treino*

Figura I.3: *Exemplos de imagens de cada classe do conjunto de validação*


Figura I.4: *Exemplos de imagens de cada classe do conjunto de treino*

Figura I.5: *Exemplos de imagens de cada classe do conjunto de teste*

Figura I.6: *Exemplos de imagens de cada classe do conjunto de teste*

# II Anexo 2 - Erros dos Modelos

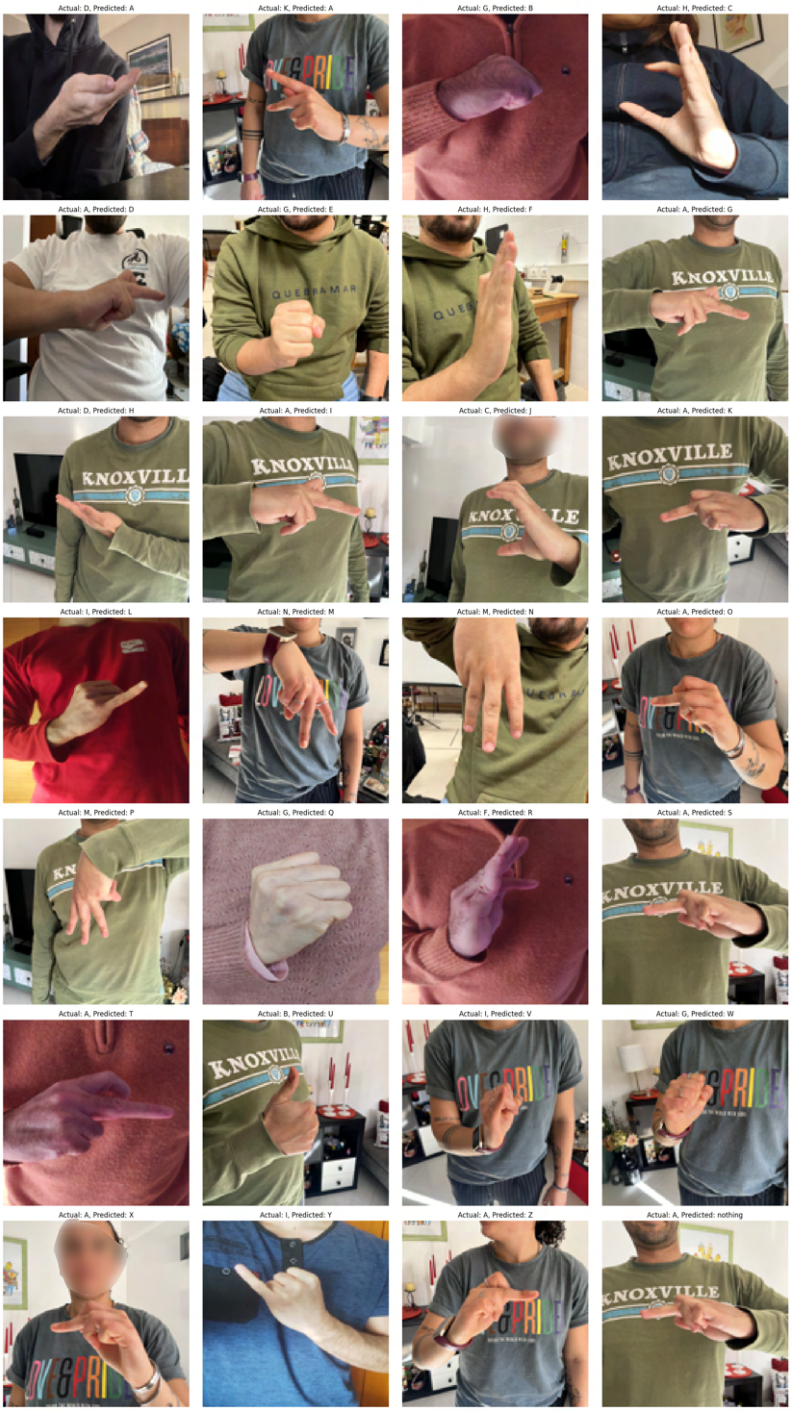


Figura II.1: *Imagem com predições incorretas*

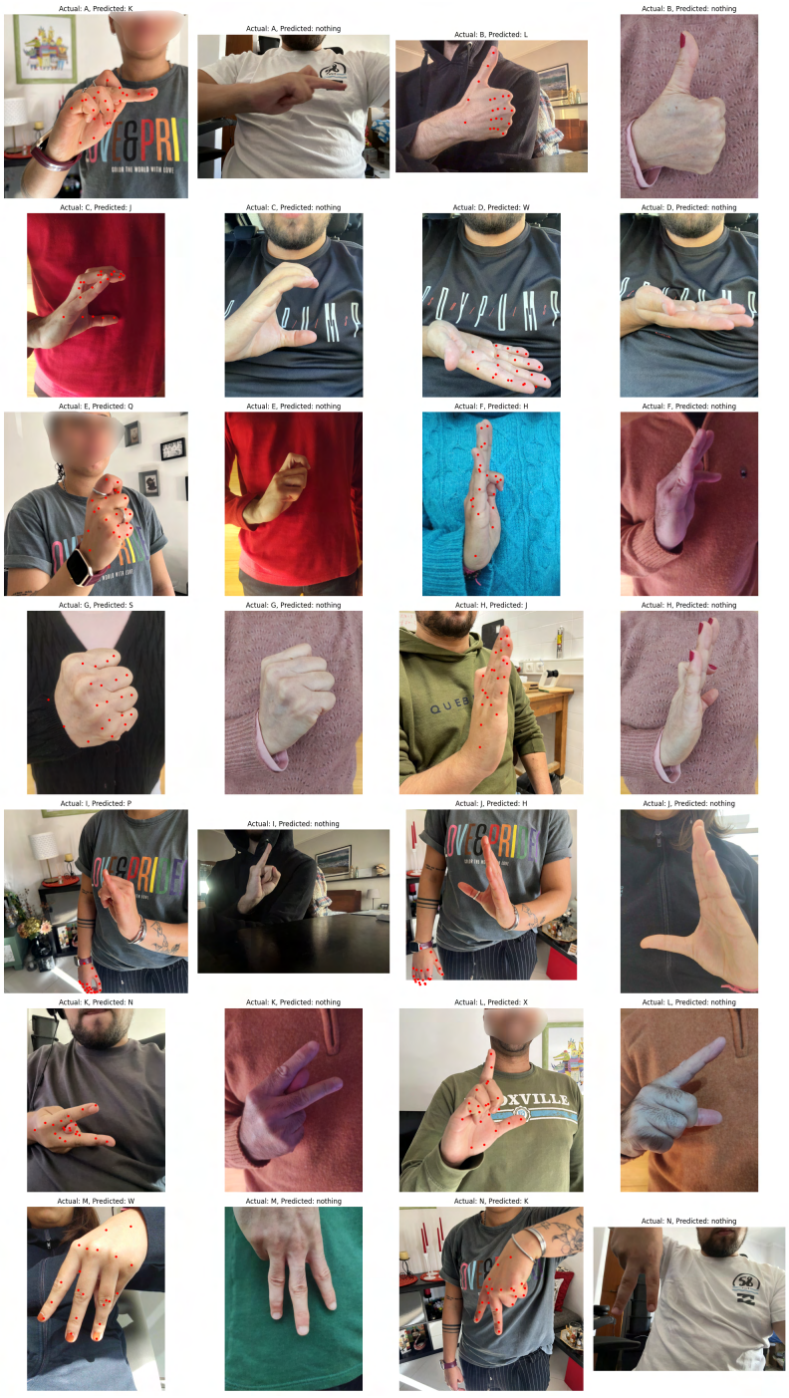


Figura II.2: *Imagem com predições incorretas, incluindo a predição "nothing".*

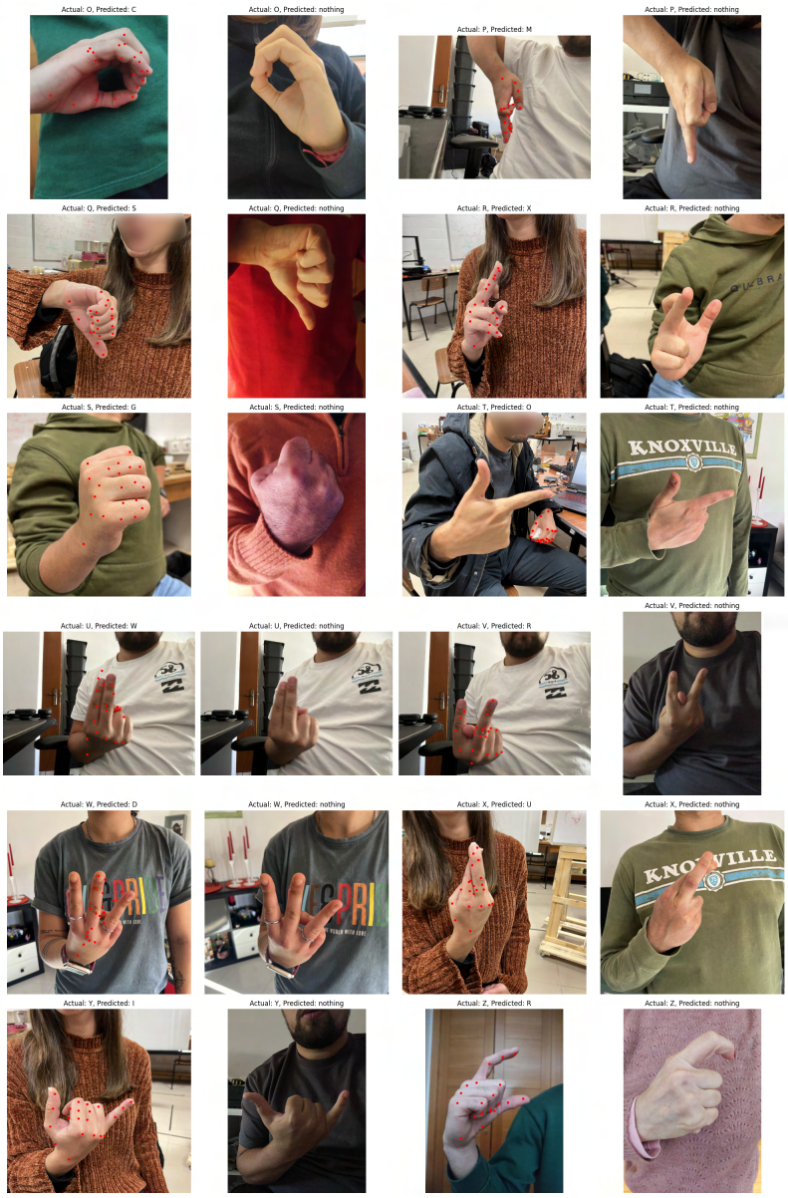


Figura II.3: *Imagem com predições incorretas, incluindo a predição "nothing".*


Figura II.4: *Imagem com predições incorretas, excluindo a predição "nothing".*

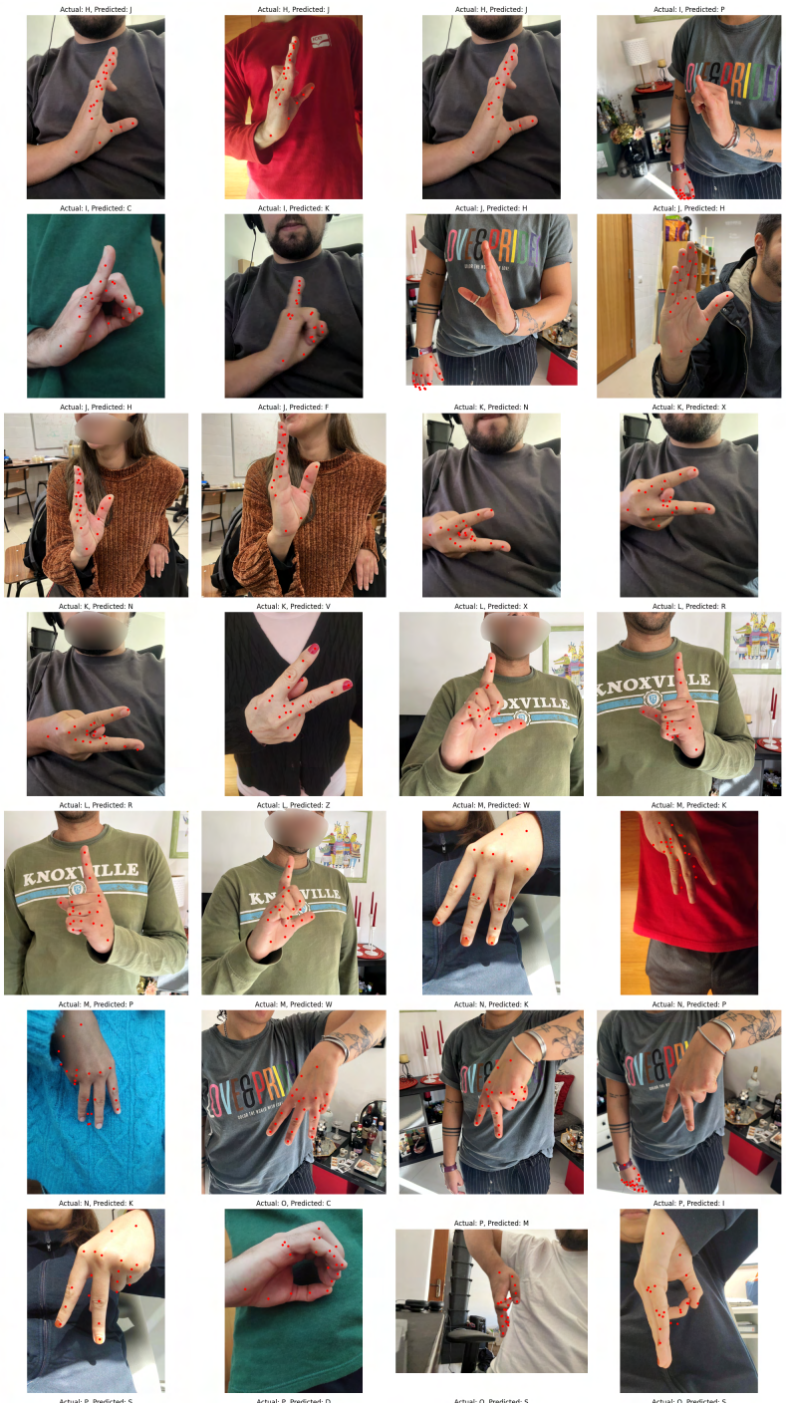


Figura II.5: *Imagem com predições incorretas, excluindo a predição "nothing".*


Figura II.6: *Imagem com predições incorretas, excluindo a predição "nothing".*


Figura II.7: *Imagem com predições incorretas, excluindo a predição "nothing".*

# III Anexo 3

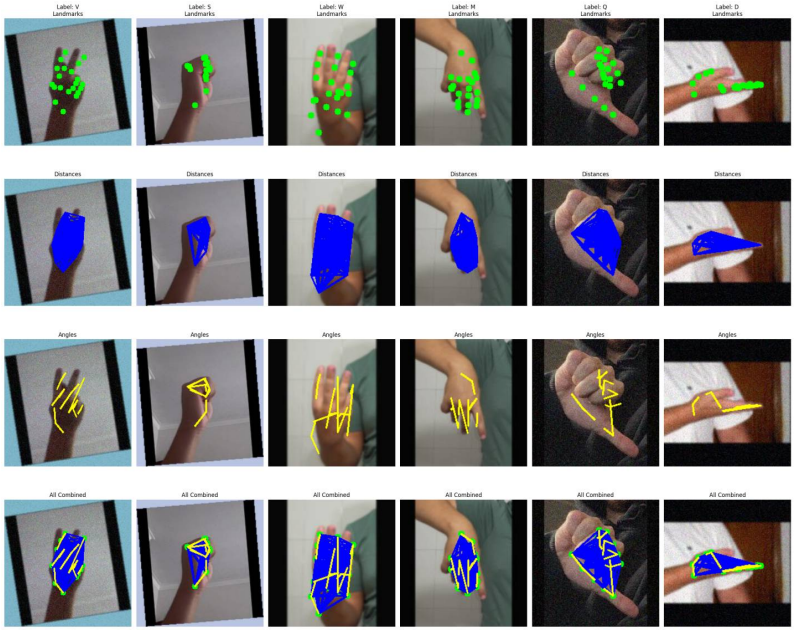


Figura III.1: *Exemplo de landmarks, distâncias e ângulos calculados.*

@is@a@figure