INTELIGÊNCIA ARTIFICIAL E REALIDADE AUMENTADA PARA RECONHECIMENTO DE PADRÕES

ARTIFICIAL INTELLIGENCE AND AUGMENTED REALITY FOR PATTERN RECOGNITION

Gustavo Monte Sião DE SANTANA
gu.santana1296@gmail.com
Ciência da Computação, Centro Universitário Padre Anchieta

Lívia Mastrangelo DE SOUZA
livia.mastrangelods@gmail.com
Ciência da Computação, Centro Universitário Padre Anchieta

Yasmin Alba TONET
Yasmintonet1@gmail.com
Ciência da Computação, Centro Universitário Padre Anchieta

Clayton Augusto VALDO
clayton.valdo@anchieta.br
Ciência da Computação, Centro Universitário Padre Anchieta

Resumo
Este artigo apresenta o desenvolvimento de um aplicativo que combina Inteligência Artificial (IA) e Realidade Aumentada (RA) para reconhecimento de padrões. Utilizando CNNs, Deep Learning e TensorFlow Lite, o sistema realiza detecção de objetos, OCR e conta com Text-To-Speech (TSS) para interação. Desenvolvido no Android Studio, possui interface intuitiva com três telas principais, processando imagens e textos em tempo real. Com o modelo MobileNet, foi treinado usando mais de 800 imagens por objeto, rotuladas em condições variadas, com o auxílio do Roboflow, garantindo precisão e leveza. O protótipo reconhece objetos como maçanetas, cadeiras, faixas de pedestres e textos, sendo ideal para acessibilidade, educação e automação, demonstrando o impacto positivo da IA e RA na inclusão social.
Palavras-Chave: Realidade Aumentada, Inteligência Artificial, Redes Neurais Convolucionais, Deep Learning, Text-To-Speech, Reconhecimento Ótico de Caracteres.

Abstract
This article presents the development of an application that integrates Artificial Intelligence (AI) and Augmented Reality (AR) for pattern recognition. Using CNNs, Deep Learning, and TensorFlow Lite, the system performs object detection, OCR, and includes a Text-To-Speech (TTS) feature for interaction. Developed in Android Studio, it features an intuitive interface with three main screens, processing images and texts in real-time. Powered by the MobileNet model, it was trained with over 800 images per object, labeled under varying conditions, using Roboflow to ensure accuracy and efficiency. The prototype recognizes objects such as doorknobs, chairs, pedestrian crossings, and texts, making it ideal for accessibility, education, and automation, demonstrating the positive impact of AI and AR on social inclusion.
Keywords: Augmented Reality, Artificial Intelligence, Convolutional Neural Networks, Deep Learning, Text-To-Speech, Optical Character Recognition

INTRODUÇÃO

Para entender a história da tecnologia no Brasil, é essencial destacar a relação próxima que ela tem

com os eventos acadêmicos. No início da década de 1960, o país teve expressivo avanço em relação ao hardware de computação. Em 1961, o Instituto Tecnológico de Aeronáutica (ITA) desenvolveu o que mais tarde foi considerado o primeiro computador brasileiro, nomeado 'Zezinho'. Até o final daquela década, as principais universidades do país já contavam com seus centros de computação, chamados de "Centro de Processamento de Dados" (WAZLAWICK; JUNIOR, 2021).

Nesse contexto acadêmico, as tecnologias de Inteligência Artificial (IA) e Realidade Aumentada (RA) surgem como os novos marcos da inovação. O surgimento da Inteligência Artificial pode ser visto como um marco tecnológico que traz implicações significativas para a sociedade. Na década de 1950, essa ideia começou a tomar forma e, em 1956, pesquisadores como John McCarthy e Marvin Minsky, da universidade estadunidense Massachusetts Institute of Technology (MIT), traçaram como objetivo o desenvolvimento de máquinas que fossem capazes de replicar o comportamento humano, resolver problemas e tomar decisões (LUG, 2023).

A Realidade Aumentada, por sua vez, é uma inovação mais recente que, se integrada com a IA, é capaz de criar experiências imersivas. Essa integração é visível principalmente no setor de varejo, na qual esse recurso permite com que os clientes interajam com produtos de interesse antes de realizar uma compra. Além disso, empresas têm utilizado a combinação de IA e RA em treinamentos, oferecendo dessa forma simulações de caráter mais realista e educacional (BOTPRESSS, 2023).

O Reconhecimento de Padrões é um campo de estudo mais amplo dentro da Inteligência Artificial e Machine Learning. Esse se baseia na extração de dados e na identificação de similaridades ou divergências em padrões, transformando-as em conhecimentos úteis, com aplicações como reconhecimento facial e detecção de objetos. Essa tecnologia tem ganhado popularidade por possibilitar com que sistemas tomem decisões complexas sem nenhuma intervenção humana. Além disso, ele desempenha um papel fundamental no Processamento de Linguagem Natural (PLN), onde as máquinas coletam informações e aprendem a entender a estrutura e o significado da linguagem humana, reconhecendo normas em textos (BICKHAM, 2023).

A proposta deste artigo é desenvolver um aplicativo capaz de realizar o reconhecimento de padrões físicos e apresentá-los através da realidade aumentada, sendo uma solução eficiente e que automatize a identificação de moldes em diversas áreas, principalmente em objetos simples do dia a dia como portas, escadas, sinalizações e textos.

FUNDAMENTAÇÃO TEÓRICA

Baseado nos conceitos apresentados, fundamentou-se uma proposta de desenvolvimento de um aplicativo de reconhecimento de padrões físicos com o uso da realidade aumentada, automatizando a identificação de moldes. A partir dessa ideia, realizou-se uma extensa pesquisa sobre tecnologias que, em conjunto, possibilitam e enriquecem a construção do aplicativo.

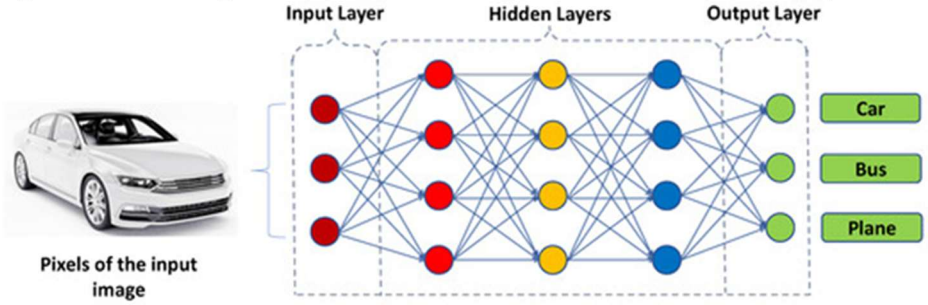
Redes Neurais Convolucionais (CNNs)

Segundo Krichen (2023), as Redes Neurais Convolucionais (CNNs) são amplamente empregadas em tarefas de reconhecimento de imagem. Elas consistem em camadas convolucionais que extraem atributos relevantes, seguidas de hierarquias totalmente conectadas para prever resultados. O treinamento da CNN envolve numerosos conjuntos de dados rotulados, na qual a rede aprende a vincular as características coletadas com etiquetas correspondentes por meio de retro propagação. Após o treinamento, ela pode classificar novos objetos, atribuindo o rótulo com a maior probabilidade de acerto.

Nos últimos anos, a aprendizagem automática evoluiu com o desenvolvimento das Redes Neurais Artificiais (RNA), inspiradas no funcionamento do cérebro humano e superando abordagens tradicionais em diversas tarefas. Como observam O'shea e Nash (2015), as CNNs, projetadas especificamente para o reconhecimento de padrões em imagens, mantêm a estrutura básica das RNAs, contudo, otimizando o processamento visual ao codificar características diretamente em sua arquitetura. Isso as confere mais acuracidade na computação de dados ópticos e reduz a quantidade

de parâmetros necessários, tornando-as mais simples e eficazes.
As CNNs foram inicialmente propostas por Kunihiko Fukushima em 1988, apesar da limitação de seu uso pelo poder computacional da época. Na década de 1990, entretanto, Yann LeCun e colaboradores aplicaram algoritmos de aprendizado baseados em gradientes às CNNs, obtendo sucesso na classificação de dígitos manuscritos. Estas tecnologias se destacam principalmente por otimizar o processamento de modelos 2D e 3D, operando com menos parâmetros que redes totalmente conectadas (ALOM et al., 2019).
A Figura 1 abaixo apresenta um exemplo da estrutura de CNN simplificada para a rotulação de objetos. A entrada, uma imagem de um automóvel, passa pela primeira camada oculta (input), na qual os filtros extraem características como arestas e texturas. Esse processo se repete por várias hierarquias, extraindo atributos cada vez mais complexos, até o nível final (output), no qual é gerado uma distribuição de probabilidade sobre as classes — carro, ônibus ou avião. A categoria com maior probabilidade é então selecionada e os pesos dos filtros são ajustados via retro propagação, minimizando o erro entre a previsão e o resultado real.

Figura 1. Representação do processamento em camadas ocultas das CNNs (KRICHEN, 2023).



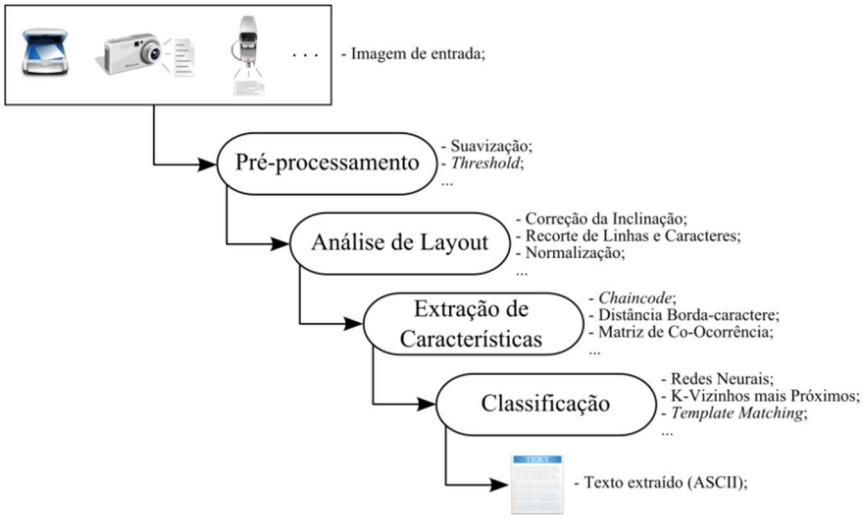
Uma CNN tipicamente é composta por diversos níveis, cada um desempenhando uma função específica. Dentre as principais camadas estão: Convolutional, Pooling, Activation, Batch Normalization, Dropout e Fully Connected. Além dessas camadas básicas, algumas CNNs incorporam gradações especializadas para capturar dependências mais complexas nos dados. CNNs recorrentes (RCNN), por exemplo, introduzem hierarquias recorrentes para lidar com pendências temporais nas sequências de entrada. Já em CNNs com Long Short-Term Memory (LSTM), são empregados níveis para aprender dependências de longo prazo, otimizando o desempenho em tarefas sequenciais (KRICHEN, 2023).
Krichen (2023) também observa que as CNNs são altamente adequadas para tarefas que envolvem a análise de informações visuais, como classificação de imagens, detecção de objetos e segmentação. Isto se dá devido a sua capacidade de identificar representações hierárquicas de dados ópticos, nas quais características de baixo nível são combinadas para formar propriedades mais complexas e abstratas, permitindo ao modelo reconhecer padrões multifacetados e realizar previsões precisas. Adicionalmente, as CNNs são competentes com elementos sequenciais como recognição de fala (Speech-to-text) e Processamento de Linguagem Natural (PNL), aproveitando estruturas espaciais ou temporais nos dados. No reconhecimento de voz, por exemplo, as camadas convolucionais podem extrair características fonéticas de uma sequência de fotogramas. Alternativamente, em PNL essas hierarquias podem captar o significado e as relações entre palavras no texto de entrada, otimizando o desempenho de ambas as áreas.

Deep Learning (DL)
Segundo Minar e Naher (2018), o Deep Learning é uma área de Machine Learning que se baseia em Redes Neurais Profundas, compostas por gradações para estudo de representações hierárquicas de dados. Cada camada da rede transforma os conteúdos de nível inferior em superior, abstraindo conceitos mais complexos. Isso permite que essas redes aprendam diretamente, e em quantidades abundantes de informações, sem a necessidade de intervenção manual para a extração de

características, o que é essencial em tarefas como visão computacional, reconhecimento de fala e processamento de linguagem natural.
Emmert-Streib et al. (2020) apontam que, no contexto da aprendizagem profunda, uma propriedade comum a todos os modelos é a capacidade de realizar o que se denomina aprendizagem de representações ou de características. Isso significa que o mecanismo não trabalha diretamente com os dados em sua forma bruta, mas adquire novas e melhores representações a partir deles. É importante destacar que esse processo ocorre em várias etapas, mediante múltiplas camadas ocultas, com cada uma aplicando transformações nas informações em diferentes níveis.
De acordo com Emmert-Streib et al. (2020), outro aspecto relevante dos mecanismos de aprendizagem profunda é a natureza não linear das transformações realizadas entre as camadas, ampliando a capacidade expressiva do modelo. Além disso, essas representações não são programadas manualmente, mas aprendidas diretamente a partir dos dados de treino, conferindo uma flexibilidade significativa aos modelos.
O aumento da capacidade computacional, especialmente com o uso de Unidades de Processamento Gráfico (GPUs), desempenhou um papel crucial no desenvolvimento prático do Deep Learning (DL). Graças ao poder de processamento paralelo das GPUs, foi possível acelerar significativamente a execução de operações em redes neurais profundas, permitindo a manipulação de volumes de dados massivos e a realização de cálculos complexos em um período mais curto. Ademais, avanços em algoritmos de otimização tornaram o treinamento de redes neurais mais competentes, viabilizando sua aplicação em larga escala (MINAR; NAHER, 2018).
As CNNs estão intimamente conectadas ao DL, se beneficiando deste aprendizado de padrões complexos para extrair automaticamente características em várias camadas, o que as torna uma das arquiteturas mais eficientes em visão computacional, como destaca Younesi (2024). Os avanços em aprendizado profundo desempenham um papel essencial no aprimoramento das Redes Neurais Convolucionais, permitindo sua aplicação em problemas de alta complexidade. Essas inovações tornaram as CNNs modelos mais robustos, capazes de lidar eficientemente com grandes volumes de dados (MINAR; NAHER, 2018).

Reconhecimento Óptico de Caracteres (OCR)
Uzai (2018) discorre sobre como a capacidade cognitiva humana para processamento de informações visuais é impressionante. Em segundos é possível não apenas entender dinâmicas de cor, perspectiva e profundidade dos elementos apresentados como também identificar os dados coletados e atribuir contexto ao conjunto observado. Diante deste mecanismo orgânico tão complexo à disposição, a ânsia para não somente replicar como também refinar e ampliar suas habilidades no meio computacional não é novidade.
Segundo Folter (2022), em meados de 1950, esforços foram realizados para automação do processamento de materiais documentados em papel para o meio digital, sem a necessidade de transcrição intermediária. Entretanto, foi o pioneiro Ray Kurzweil, em 1974, que formalizou a ferramenta capaz de reconhecer textos em fontes impressas e vocalizá-los via aprendizado de máquina, desenvolvido para acessibilidade para pessoas com deficiência visual, aponta IBM (2024). Desde então, a tecnologia denominada como Reconhecimento Óptico de Caracteres — ou, do inglês, como é popularmente referida, Optical Character Recognition (OCR) — foi extremamente aprimorada, sendo um recurso apto a discernir e converter textos, símbolos e padrões contidos em imagens para documentos editáveis, seja em escrita impressa ou manuscrita (FARIAS, 2023).
A figura 2 abaixo expõe de forma simples o fluxo de conversão do sistema OCR para processar imagens de forma legível à máquina. O primeiro estágio se baseia na aquisição de uma base de evidências fotográficas contendo os objetos-alvo para computação, variando em cores, iluminação, contraste, distância e tamanho da fonte, com o intuito de simular as pluralidades de diferentes ambientes (DE OLIVEIRA, PITERI E ARTERO, 2016).

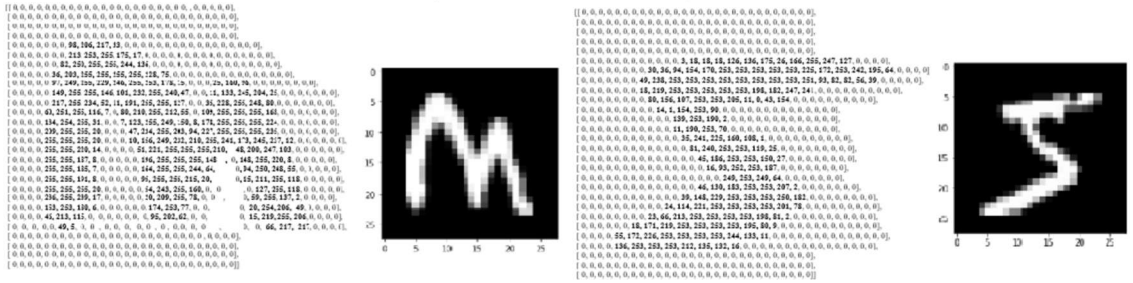
Figura 2. Esquema figurativo das etapas de processamento em um sistema OCR (DE OLIVEIRA, PITERI E ARTERO, 2016).



Farias (2023) explica que, a fim de potencializar a acuracidade e eficiência na detecção de padrões, aumentar a legibilidade e riqueza de detalhes é uma etapa substancial. Nesse sentido, o pré-processamento objetiva auxiliar na diferenciação entre as áreas de interesse e os demais elementos em tela, mediante estratégias como o aumento de resolução (upscaling) por densidade de pontos por polegadas (DPI), conversão para escala monocromática cinza, limiarização (threshold), redução de ruídos e/ou granulação por suavização.

Após a minimização de possíveis defeitos visuais, a fase de análise de layout e segmentação utiliza técnicas de detecção de bordas para determinar o ângulo de inclinação da foto e efetuar a rotação para melhor reconhecimento. A partir disso, um mapeamento em linhas e colunas indica a distribuição de objetos na composição. Mediante esta subdivisão, é possível isolar os caracteres e normalizar a quantidade de píxeis por matriz na escala da imagem, conforme ilustrado na figura 3. A recognição é orientada da esquerda para a direita, de cima a baixo (DE OLIVEIRA, PITERI E ARTERO, 2016; ENUCOMPI, 2023).

Figura 3. Exemplos de matrizes de píxeis normalizadas da letra m e número 5, respectivamente (ENGETECH, 2021).



De acordo com De Oliveira, Piteri e Artero (2016) e Enucompi (2023), no processo de extração de características, diversas metodologias relacionadas à quantificação de atributos individuais são aplicadas com o propósito de agrupar propriedades relevantes para o reconhecimento e associação de padrões. O produto desta operação é um vetor numérico que descreve as particularidades gravadas durante o aprendizado de máquina. Alicerçado nisso se inicia a classificação, na qual a cada ciclo de identificação destes conjuntos, estes são então confrontados com os anteriores por intermédio de estruturas de CNN para determinar o grau de similaridade entre os objetos já conhecidos e as novas amostras.

Text-To-Speech (TTS)

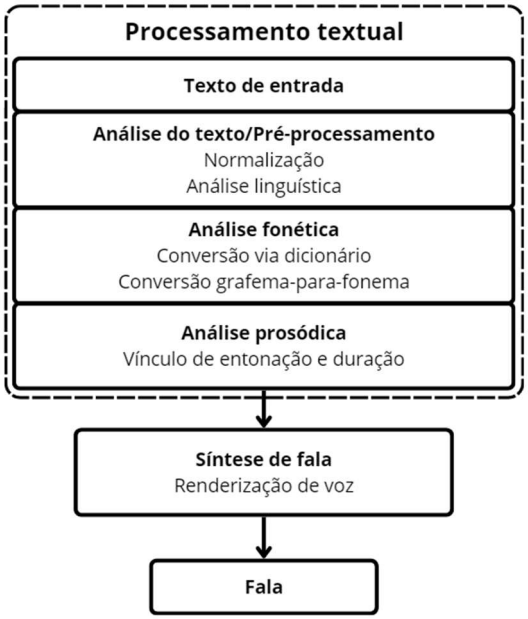
Pant et al. (2018) destacam que a comunicação verbal é uma das formas mais assertivas para troca de informação, evitando possíveis interpretações equivocadas, situações que são mais recorrentes em outros meios de interação digital, como a visual — via imagens ou ilustrações — ou escrita, como por mensagens e e-mails. Com a conveniência proporcionada pela era tecnológica, é natural que a ideia de incorporar esta eficiência ao ambiente computacional se tornasse uma necessidade. Embora existam precursores relacionados que datam desde o século XVIII, foi somente em 1968, no laboratório Eletrotécnico no Japão, que Noriko Umeda desenvolveu o primeiro sistema formalizado de Text-To-Speech (TTS) para a língua inglesa (WEITZMAN, 2022).

Esta tecnologia pode ser definida como a produção computacional de sentenças enunciadas baseadas em entradas de texto digital, possuindo uma gama de utilidades em áreas como aprendizado linguístico, serviços de telecomunicação, indústria de jogos e brinquedos, pesquisas acadêmicas e no auxílio de pessoas com distúrbios motores. A diferença fundamental entre TTS e outros sistemas de saída de áudio — os chamados Voice Response Systems — reside em sua habilidade de automatizar a síntese de novas palavras, sem a limitação de um grupo de vocabulário pré-estabelecido (DUTOIT, 1997).

Diferente da curva de aprendizado humano, composta por indicativos de pronúncia, mesmo para palavras novas, por conta da bagagem acumulada de variáveis de voz como fonemas, fluência, entonação e afinação, máquinas TTS dependem de um sistema de avaliações críticas para gerar a estruturação da fala (PANT et al., 2018).

Seguindo o esquema apresentado na figura 4, o fluxo se inicia com a entrada do texto digital, seja ele originado de dispositivos de entrada (inputs), scanners ou, semelhantemente ao empregado neste estudo, por meio de um sistema de Reconhecimento Óptico de caracteres. A análise dos trechos informados se inicia pela identificação de pontuações e formatações de parágrafo, organizando posteriormente as sentenças em listas lógicas e desmembrando-as em grupos de números, abreviações, acrônimos e sinais. Após a normalização, é realizada uma série de verificações linguísticas para sintaxe, semântica, morfologia e pragmática, a fim de facilitar a articulação e possíveis inconsistências textuais nos estágios subsequentes (DUTOIT, 1997; KATADE ET AL., 2023).

Figura 4. Representação gráfica simplificada do processamento realizado por sistemas TTS (KATADE ET AL., 2023).



Dutoit (1997) e Katade et al. (2023) explicam que após finalizada a etapa de pré-processamento, estabelece-se a transcrição fonética dos objetos de entrada. Esta composição sonora pode ser

estruturada mediante duas abordagens: a orientada a dicionário ou a regra. A primeira, como o nome indica, baseia-se apenas nas definições de pronúncia documentadas em dicionários, apoiando-se em flexões morfofonêmicas para prever como serão vocalizadas ressalvas não registradas. Apesar de ocupar consideravelmente menor espaço no processamento, esta estratégia é mais propensa a apresentar inconsistências audíveis, tomando um carácter mais robótico. Em contraposição a isso, a regra grafema-para-fonema, por sua vez, foca em atribuir à fala a maior quantidade de individualidade sonora, aderindo regras de exceção para cada uma das palavras.

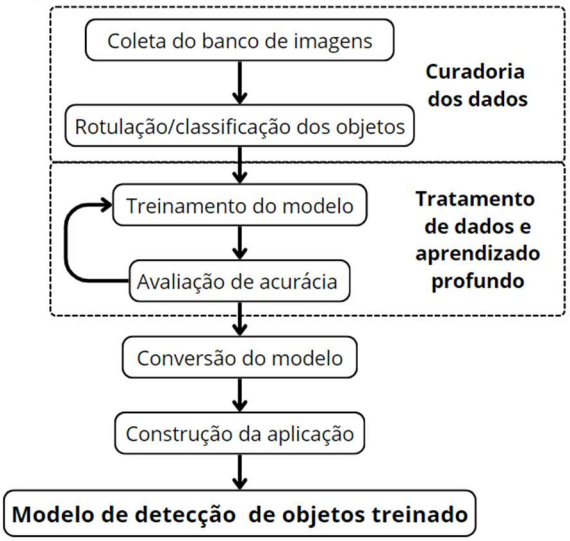
Independentemente do método utilizado, as amostras textuais são então processadas sílaba a sílaba por funções prosódicas como afinação, volume, duração de fala por sílaba, análise tônica e átona. A entonação possui um grande papel no contexto e intenção de uma frase, logo, a sofisticação computacional para mímica destas características verbais atribui à comunicação um tom muito mais natural e familiar aos usuários (DUTOIT, 1997; KATADE et al., 2023).

A síntese vocal pode ser entendida como o processo de produção artificial de fala humana. Essa é uma das etapas mais importantes do desenvolvimento, pois determina como o processamento será apresentado e, consequentemente, seu desempenho. Com isso em mente, os dados tratados previamente passam então por três técnicas fundamentais. A primeira, síntese articulada, foca na simulação biológica da fala. Já a integração dos formantes é responsável por agrupar os componentes acústicos em bases perimétricas. Por fim, a renderização pode ser baseada em: exemplares de fala pré-gravados; formantes, que simulam cordas vocais humanas; e a concatenação, que consolida as unidades de som mediante modelos sílaba a sílaba. O resultado deste encadeamento de etapas resulta na saída de dados em áudio (KATADE et al., 2023).

RESULTADOS E DISCUSSÕES

Como evidência prática deste artigo foi desenvolvida uma aplicação que, mediante um modelo treinado de aprendizado profundo, é capaz de identificar em tempo real textos e objetos pré-selecionados como maçanetas, cadeiras e faixas de pedestre. Explanando a estrutura de processos apresentada na figura 5, a primeira etapa baseia-se na coleta de dados de interesse em diversos ambientes e condições de visualização, por meio de conteúdos online, fotos ou vídeos de autoria própria, ou manipulação de bancos pré-existentes conforme comentado por Asif (2024). A fim de garantir uma eficiência satisfatória nos passos posteriores, englobou-se ao menos 500 imagens de natureza diferente para cada um dos itens.

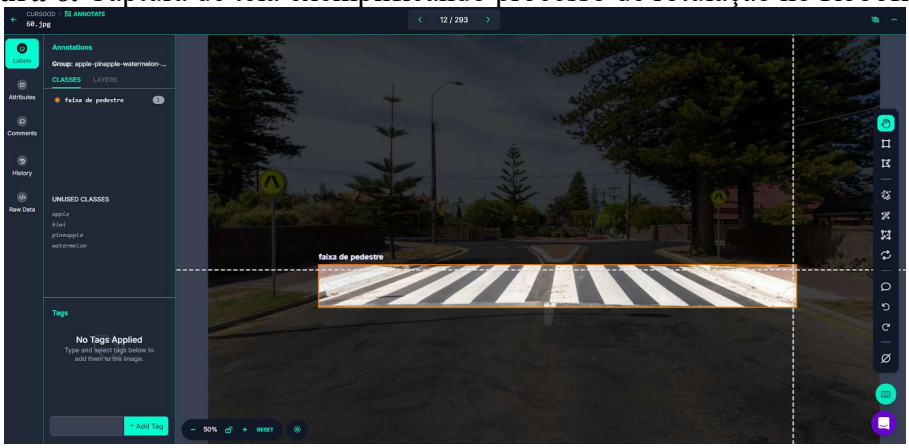
Figura 5. Representação do fluxo geral desenvolvido no projeto.



Diferente do reconhecimento de categorias, que classifica fotografias em sua totalidade e indica apenas a presença ou não do objeto alvo, a detecção de objetos é capaz de, além de identificar,

localizar também sua posição relativa na cena. Todavia, para alcançar tal resultado o modelo deve passar pela rotulação, passagem em que cada um dos conteúdos do banco recebe uma anotação assinalando a classe que o mesmo representa, em adição a visualização da delimitação do elemento em cena, como exemplificado na figura 6. Existem diversas ferramentas disponíveis para esta finalidade, contudo, aos fins desta composição foi utilizado o framework de visão computacional Roboflow.

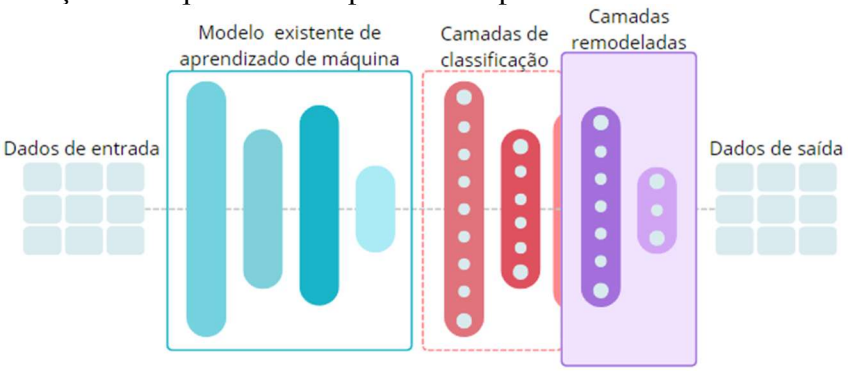
Figura 6. Captura de tela exemplificando processo de rotulação no Roboflow.



Asif (2024) discorre que uma vez completa a rotulação, o conjunto de dados deve então ser segregado em três pastas distintas para treinamento, validação e testes, respeitando balanceamento de 80% para o primeiro e 10% às demais. Adicionalmente, a plataforma Roboflow disponibiliza uma série de ferramentas para ampliar evidências, gerando novas imagens a partir das já catalogadas pela alteração de propriedades como tamanho, contraste e rotação. Posto isso, um arquivo de extensão zip é gerado com todas as fotos e anotações em XML.

A capacitação do modelo foi constituída por um algoritmo desenvolvido na linguagem de programação Python, fornecido publicamente pelo Google em servidor online (Google Colab). Este ambiente, aliado principalmente à biblioteca MediaPipe Model Maker, está respaldado no conceito de aprendizagem por transferência para simplificar o processamento dos data sets. Consoante o informativo no Criador de modelos do MediaPipe (2024), ao invés de processá-los a partir do zero, é reutilizada grande parte dos métodos aplicados em modelos de detecção já fundamentados em milhares de imagens, recriando as camadas de saída para se adequar ao novo conjunto personalizado. Esta abordagem, demonstrada na figura 7, é consideravelmente mais rápida e eficiente, sendo a versão mais recente e avançada para treinar protótipos para dispositivos de ponta.

Figura 7. Representação da arquitetura de aprendizado por transferência utilizada no Model Maker.



A remodelação e treinamento ocorre mediante um poderoso mecanismo denominado TensorFlow lite. Este framework de código aberto otimiza o aprendizado de máquina para dispositivos de baixa latência, proporcionando compatibilidade e compactabilidade de recursos entre os arquivos gerados

e aplicações mobile, sem a necessidade de conexão com a internet ou perda de dados, pois não há fluxo direto com um servidor online (TENSORFLOW […], 2021).
Priorizando a exatidão e padronização do protótipo, o aprendizado de máquina trabalha alicerçado em três parâmetros, conforme figura 8. Visando otimizar recursos, alguns processos são realizados simultaneamente, e o batch_size, o primeiro destes valores, simboliza justamente o número de imagens que serão agrupadas a cada lote de apreciação. O learning_rate, por sua vez, visa controlar a métrica de acuracidade dos ciclos de estudo. Distintivamente, o epoch diz respeito a quantidade de vezes que o banco de dados será processado pelo algoritmo. Estabelecidas estas definições, são configuradas as opções específicas do modelo de detecção reaproveitado, sendo nesta circunstância aplicada uma versão do MobileNet, um modelo de rede neural leve e eficiente projetado para dispositivos com capacidade computacional limitada. Por último, a função create desencadeia o fluxo de aprendizado conforme as propriedades informadas (ASIF, 2024).

Figura 8. Código fonte utilizado para definir os parâmetros de treino do modelo.

```
hparams = object_detector.HParams(batch_size=8, learning_rate=0.3, epochs=50, export_dir='exported_model')
options = object_detector.ObjectDetectorOptions(
    supported_model=object_detector.SupportedModels.MOBILENET_V2,
    hparams=hparams
)
model = object_detector.ObjectDetector.create(
    train_data=train_data,
    validation_data=val_data,
    options=options)
```

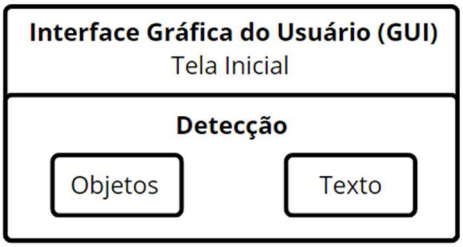
Asif (2024) especifica dois conceitos para validação do desempenho em DL, a precisão média e a intersecção sobre o valor da união. A primeira busca avaliar o desempenho do modelo estimando a conformidade dos resultados com os padrões conhecidos. A intersecção sobre o valor da união, em contrapartida, analisa o índice de sobreposição entre a área identificada pelo protótipo na imagem e a classe de teste validada. Ambas as metodologias quantificam as previsões em um intervalo de 0 a 1, sendo o último correspondente ao valor máximo. Através da biblioteca COCO, o exemplar foi ciclicamente examinado e realimentado ao treinamento, atualizando os valores dos parâmetros. Atingindo resultados satisfatórios, o arquivo pôde então ser exportado com a extensão tflite para próxima etapa de desenvolvimento.
Focando na eficiência para edição de código e versionamento mobile, a construção da aplicação fundamentou-se no ambiente de desenvolvimento integrado (IDE) conhecido como Android Studio. Estruturam-se três telas para abranger todas as funcionalidades propostas, seguindo o layout na figura 9 abaixo.

Figura 9. Layout das telas implementadas no aplicativo.

No concerne ao fluxo de arquitetura lógica do aplicativo, a figura 10 apresentada abaixo elucida a

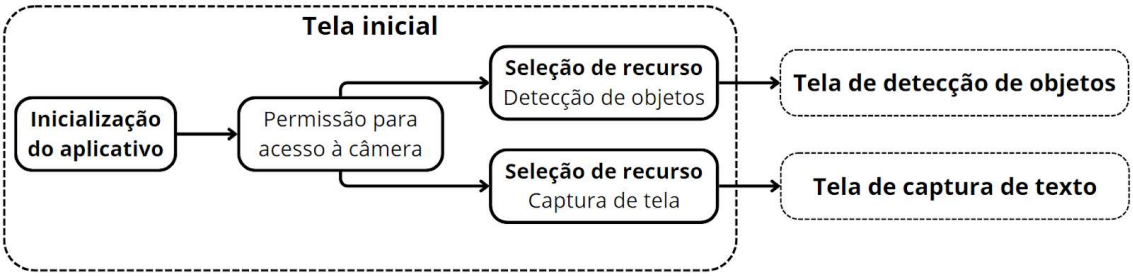
elaboração técnica do código, desenvolvido primordialmente em Java. Priorizando o esclarecimento quanto aos processos executados no desenho da aplicação, anexou-se a este capítulo somente os trechos de maior relevância ao assunto discorrido.

Figura 10. Representação figurativa da arquitetura desenvolvida no projeto.



A tela principal, apresentada na figura 11, serve como ponto inicial para seleção das subsequentes, apresentando destaque em relação às decisões estéticas do projeto, como o uso de cores contrastantes, facilitando a visualização dos campos escritos, em adição ao logo, artifício essencial para identidade visual. Tanto a seleção do recurso de detecção de objetos quanto a de captura de texto desencadeiam a ativação da câmera, distinguindo-se ao nível de design apenas em campos específicos como o botão para captar imagens.

Figura 11. Esquema apresentando os recursos da tela inicial.



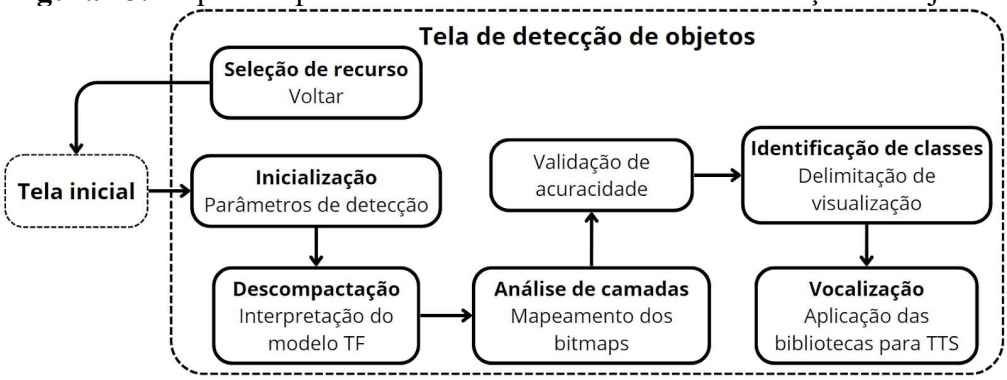
A primeira configuração realizada refere-se ao mecanismo mais substancial da aplicação, o acesso à câmera. Isto não apenas por ser uma autorização que deve ser explicitamente acordada por parte do usuário, como também pelo fato de que os artifícios programados dependem dessa visualização. A figura 12 aborda algumas destas especificações para compatibilidade, definindo a necessidade da permissão, a necessidade de uma câmera no dispositivo e funcionalidade mínima de autofoco.

Figura 12. Exemplo de pré-requisitos relacionados ao uso da câmera.

```
<uses-permission android:name="android.permission.CAMERA" />
<uses-feature android:name="android.hardware.camera" />
<uses-feature android:name="android.hardware.camera.autofocus" />
```

A fim de compreender o desenho do mecanismo de detecção de objetos exposto na figura 13, deve-se ter em mente que seu uso em tempo de execução é extremamente dinâmico. Por conta disso, são definidas algumas propriedades antes da inicialização para evitar sobrecargas, como limite de itens reconhecidos simultaneamente, cores para diferenciação de planos e porcentagem de acurácia mínima para vocalização.

Figura 13. Esquema apresentando os recursos da tela de detecção de objetos.



Além disso, o trecho contido na figura 14 abaixo é responsável por inicializar o rastreador a partir do momento em que o usuário é direcionado à esta tela. Este realiza ainda algumas declarações como o percentual de precisão para identificação — o que se refere a no mínimo 80% neste caso — uso da GPU e a nomeação do modelo que será descompactado e interpretado pelo TensorFlow.

Figura 14. Inicializador do processo de detecção de objetos.

```
    tracker = new MultiBoxTracker( context: this);
    objectDetectorHelper = new ObjectDetectorHelper( threshold: 0.8f, ObjectDetectorHelper.MAX_RESULTS_DEFAULT, ObjectDetectorHelper.DELEGATE_CPU,
            modelName: "objetos.tflite", RunningMode.IMAGE, getApplicationContext(), objectDetectorListener: this);
}
```

O próximo passo então é o mapeamento das fotografias bitmaps coletadas e alimentadas ao protótipo para treinamento. Segundo a imprensa da Universidade Federal do Ceará (2024), bitmaps são formatos de imagem que armazenam suas informações em matrizes de pixels. Este mapeamento percorre as camadas de aprendizado com intuito de comparar as classes treinadas com os objetos visíveis em tela. Ao obter uma correspondência de precisão satisfatória, os recursos Text-To-Speech são ativados e a lista resultante de itens detectados é utilizada como base para criação de novos objetos, que serão delimitados pelas caixas de marcação em tela, como demonstra a figura 15. Vale ressaltar que, ao ser chamado, o TTS pode definir as preferências de idioma consoante a linguagem nativa da máquina ou manualmente.

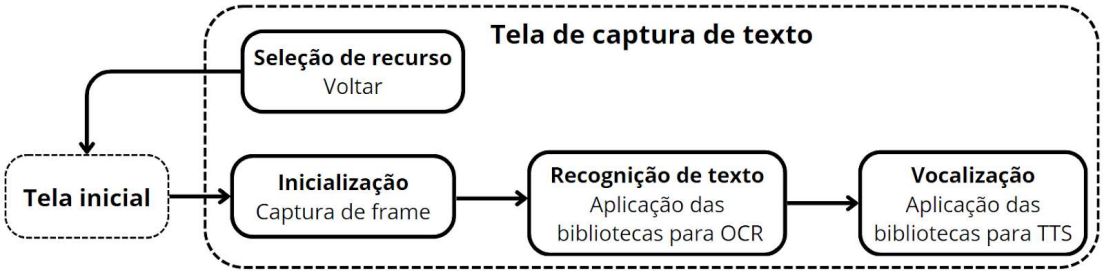
Figura 15. Mapeamento Bitmap para detecção de objetos.

```
public ResultBundle detectImage(Bitmap image) {  1 usage
    if (runningMode != RunningMode.IMAGE || objectDetector == null) {
        return null;
    }

    long startTime = SystemClock.uptimeMillis();
    MPImage mpImage = new BitmapImageBuilder(image).build();
    ObjectDetectionResult objectDetectionResult = objectDetector.detect(mpImage);

    if (objectDetectionResult != null) {
        long inferenceTimeMs = SystemClock.uptimeMillis() - startTime;
        List<ObjectDetectionResult> results = new ArrayList<>();
        results.add(objectDetectionResult);

        if (!objectDetectionResult.detections().isEmpty()) {
            String detectedCategory = objectDetectionResult.detections().get(0).categories().get(0).categoryName();
            String speechText = detectedCategory;
            textToSpeech.speak(speechText, TextToSpeech.QUEUE_FLUSH, params: null, utteranceId: null);
        }

        return new ResultBundle(results, inferenceTimeMs, image.getHeight(), image.getWidth());
    }

    return null;
}
```

A tela para captura de textos na figura 16, em contrapartida, apresenta uma leitura vocalizada em

tempo real baseada em frames e, por conta disso, pode apresentar dificuldade caso o percentual de detecção não seja relativamente estável. Com o intuito de assegurar a eficácia desta funcionalidade, um botão para captura de frames foi adicionado, evitando assim o acúmulo de informações processadas simultaneamente.

Figura 16. Inicializador do processo de captura de textos.



A figura 17 descreve o evento engatilhado após o clique, que chama a função captureImage () a fim de realizar a coleta de uma cena estática a partir da visualização em câmera. Seguindo esta dinâmica, a biblioteca para computação do OCR é ativada através do comando recognizeText(inputImage), que já inclui tratamento de erros em casos de falhas na captura ou reconhecimento de caracteres. Caso o processo seja bem-sucedido, uma função para enfileirar os recursos de TTS é inicializada e reproduz o texto alimentado.

Figura 17. Código fonte para captura e processamento de texto baseado em frames.

```
private void captureImage() {  1 usage
    imageCapture.takePicture(ContextCompat.getMainExecutor( context: this), new ImageCapture.OnImageCapturedCallback() {
        @Override  3 usages
        public void onCaptureSuccess(@NonNull ImageProxy image) {
            super.onCaptureSuccess(image);
            InputImage inputImage = InputImage.fromMediaImage(image.getImage(), image.getImageInfo().getRotationDegrees());
            recognizeText(inputImage);
            image.close();
        }

        @Override
        public void onError(@NonNull ImageCaptureException exception) {
            super.onError(exception);
            Toast.makeText( context: LeituraActivity.this,  text: "Erro ao capturar imagem", Toast.LENGTH_SHORT).show();
        }
    });
}
```

CONSIDERAÇÕES FINAIS

As facilidades e possibilidades que a Inteligência Artificial e Realidade Aumentada representam em conjunto são extremamente transformadoras. Algumas tecnologias selecionadas, como Redes Neurais Convolucionais, Aprendizado Profundo, Reconhecimento Ótico de Caracteres e Text-To-Speech, quando aliadas ao desenvolvimento de melhorias têm o potencial de enriquecer aplicações diversas, variando desde recursos de acessibilidade para pessoas no espectro da deficiência visual, vigilância para segurança em lojas e fábricas, desempenho de trabalho, análise de tráfego, contagem de multidões até a detecção de anomalias e padrões de foco de doenças em áreas como saúde e agropecuária.

O estudo se utilizou de um modelo de aprendizado de máquina adaptado, capaz de treinar um novo protótipo com base no banco com alimentação de imagens verificadas e classificadas. Após estabilizar sua acuracidade, o modelo foi então incorporado pelo código da aplicação mobile, no qual realizou tanto a adição de elementos gráficos, como interface das telas, e o processamento deste modelo em si, englobando, por exemplo, permissões de câmera, inicialização de bibliotecas e APIs para

reconhecimento de itens, vocalização de classes e textos identificados.
Entretanto, a implementação enfrentou alguns empecilhos a serem superados em próximas análises do tema, com destaque a dificuldades relacionadas à padronização de qualidade nas imagens para aprendizado e incompatibilidade do aplicativo com sistemas operacionais iOS.
Para futuros desenvolvimentos, indicam-se melhorias como a integração do aplicativo com dispositivos externos, permitindo, por exemplo, a leitura auxiliada para pessoas com deficiência visual mediante câmeras estáveis estabilizadas no usuário, eliminando a necessidade do direcionamento do celular para identificação.

REFERÊNCIAS BIBLIOGRÁFICAS

ALOM, Md Zahangir; TAHA, Tarek M.; YAKOPCIC, Chris; WESTBERG, Stefan; SIDIKE, Paheding; NASRIN, Mst Shamima; HASAN, Mahmudul; VAN ESSEN, Brian C.; AWWAL, Abdul A. S.; ASARI, Vijayan K. Computer Science & Engineering. *A State-of-the-Art Survey on Deep Learning Theory and Architectures*, [s. l.], 5 mar. 2019. DOI https://doi.org/10.3390/electronics8030292. Disponível em: https://www.mdpi.com/2079-9292/8/3/292. Acesso em: 23 out. 2024.

ANAIS DO ENCONTRO UNIFICADO DE COMPUTAÇÃO DO PIAUÍ (ENUCOMPI), 2023, Piauí. *Uso de Reconhecimento Óptico de Caracteres para Extração de Textos em Imagens de Redações* […]. [S. l.: s. n.], 19/10/2023. Disponível em: https://sol.sbc.org.br/index.php/enucompi/article/view/26617. Acesso em: 19 out. 2024.

BICKHAM, Barbara. *The Future of AI: How Pattern Recognition is Driving Innovation in Machine Learning.* [*S. l.*], 29 dez. 2023. Disponível em: https://www.trailyn.com/the-future-of-ai-how-pattern-recognition-is-driving-innovation-in-machine-learning/. Acesso em: 26 set. 2024.

COMUNIDADE, BootPress. *Guia definitivo da Inteligência Artificial (IA) e da Realidade Aumentada (RA).* [*S. l.*], 23 set. 2023. Disponível em: https://botpress.com/pt/blog/ultimate-guide-to-artificial-intelligence-ai-and-augmented-reality-ar. Acesso em: 24 set. 2024.

DE OLIVEIRA, Helder; PITERI, Marco; ARTERO, Almir. Desenvolvimento de uma Plataforma de Software para o Reconhecimento Óptico de Caracteres. *CNMAC — Congresso Nacional de Matemática Aplicada e Computacional,* [*s. l.*], 10 mar. 2016. Disponível em: https://www.researchgate.net/publication/266317024_Desenvolvimento_de_uma_Plataforma_de_Software_para_o_Reconhecimento_Optico_de_Caracteres. Acesso em: 19 out. 2024.

DUTOIT, Thierry. High-quality text-to-speech synthesis: an overview. *JOURNAL OF ELECTRICAL AND ELECTRONICS,* [*s. l.*], 1 jan. 1997. Disponível em: https://www.semanticscholar.org/paper/High-quality-text-to-speech-synthesis-%3A-an-overview-Dutoit/36cb0e3d47a76bc1e87529cbb2fd965a37ade12a. Acesso em: 20 out. 2024.

EMMERT-STREIB, Frank; YANG, Zhen; FENG, Han; TRIPATHI, Shailesh; DEHMER, Matthias. Machine Learning and Artificial Intelligence. *An Introductory Review of Deep Learning for Prediction Models With Big Data,* [*s. l.*], 27 fev. 2020. DOI https://doi.org/10.3389/frai.2020.00004. Disponível em: https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2020.00004/full. Acesso em: 24 out. 2024.

ENCONTRO DE GESTÃO E TECNOLOGIA (ENGETECH), 2021, São Paulo. *RECONHECIMENTO ÓPTICO DE CARACTERES E TRANSCRIÇÃO DE TEXTOS MANUSCRITOS* […]. [S. l.: s. n.], 24/11/2021. Disponível em: https://www.fateczl.edu.br/engetec/engetec_2021/4_EnGeTec_paper_167.pdf. Acesso em: 19 out. 2024.

FARIAS, WALISSON. *APERFEIÇOANDO O RECONHECIMENTO ÓPTICO DE CARACTERES EM IMAGENS DE DOCUMENTOS PESSOAIS.* 2023. Trabalho de Conclusão de Curso (TCC) (Bacharelado) - Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, [*S. l.*], 2023. Disponível em: http://dspace.sti.ufcg.edu.br:8080/jspui/bitstream/riufcg/34848. Acesso em: 19 out. 2024.

FOLTER, Rodrigo. *O que é OCR? | Reconhecimento ótico de caracteres.* [*S. l.*], 7 jun. 2022.

Disponível em: https://canaltech.com.br/internet/o-que-e-ocr-reconhecimento-otico-de-caracteres/. Acesso em: 19 out. 2024.
JUNIOR, Deógenes; WAZLAWICK, Raul. *Histórico de Eventos de Computação no Brasil.* [*S. l.*], 16 mar. 2021. Disponível em: https://horizontes.sbc.org.br/index.php/2021/03/historico-de-eventos-de-computacao-no-brasil/. Acesso em: 24 set. 2024.
KATADE, Gopal *et al*. Review on Text to Speech Synthesizer. *International Journal of Scientific Research in Science and Technology,* [*s. l.*], 1 jun. 2023. Disponível em: https://ijsrst.com/paper/11180.pdf. Acesso em: 20 out. 2024.
KRICHEN, Moez. Department of Information Technology. *Convolutional Neural Networks: A Survey*, [*s. l.*], 28 jul. 2023. DOI https://doi.org/10.3390/computers12080151. Disponível em: https://www.mdpi.com/2073-431X/12/8/151. Acesso em: 24 out. 2024.
LUG, Andre. *Introdução à criação da inteligência artificial.* [*S. l.*], 21 mai. 2023. Disponível em: https://andrelug.com/a-evolucao-da-inteligencia-artificial-do-pioneirismo-a-aplicacao-contemporanea/. Acesso em: 24 set. 2024.
MINAR, Matiur Rahman; NAHER, Jibon. Machine Learning. *Recent Advances in Deep Learning: An Overview,* [*s. l.*], 21 jul. 2018. DOI https://doi.org/10.48550/arXiv.1807.08169. Disponível em: https://arxiv.org/abs/1807.08169. Acesso em: 24 out. 2024.
O que é reconhecimento ótico de caracteres (OCR)? [*S. l.*], 12 jul. 2024. Disponível em: https://www.ibm.com/br-pt/think/topics/optical-character-recognition. Acesso em: 19 out. 2024.
O'SHEA, Keiron; NASH, Ryan. Neural and Evolutionary Computing; Machine Learning; Computer Vision and Pattern Recognition. *An Introduction to Convolutional Neural Networks,* [*s. l.*], 2 dez. 2015. DOI https://doi.org/10.48550/arXiv.1511.08458. Disponível em: https://arxiv.org/abs/1511.08458. Acesso em: 23 out. 2024.
PANT, Navya *et al*. *Speech to text and text to speech recognition systems: A review.* IOSR Journal of Computer Engineering (IOSR-JCE), [*s. l.*], 17 mar. 2018. Disponível em: https://www.academia.edu/download/87530288/E2002013643.pdf. Acesso em: 20 out. 2024.
PENG, Kelly. *A Tale of Two Technologies: How AI and AR Tech Are Changing Our World.* [*S. l.*], 11 ago. 2023. Disponível em: https://www.dataversity.net/a-tale-of-two-technologies-how-ai-and-ar-tech-are-changing-our-world/. Acesso em: 26 set. 2024.
Qual a diferença entre imagens bitmaps e vetoriais? [S. l.], 1 mar. 2016. Disponível em: https://imprensa.ufc.br/pt/duvidas-frequentes/diferenca-entre-imagens-bitmaps-e-vetoriais/. Acesso em: 2 nov. 2024.
TRAIN Custom Object Detection Models in Tensorflow Lite format and build Android Apps using Java and Kotlin | Android ML. [S. l.], 1 set. 2024. Disponível em: https://www.udemy.com/course/train-object-detection-models-for-android-java-kotlin/?couponCode=KEEPLEARNING. Acesso em: 28 set. 2024.
TENSORFLOW Lite. [S. l.], 9 jan. 2021. Disponível em: https://www.tensorflow.org/lite/guide?hl=pt-br. Acesso em: 2 nov. 2024.
UZAI, LUIS. *RECONHECIMENTO ÓPTICO DE CARACTERES MANUSCRITOS USANDO REDES NEURAIS CONVOLUCIONAIS.* 2018. Trabalho de Conclusão de Curso (TCC) (Bacharelado) - Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, [*S. l.*], 2018. Disponível em: https://repositorio.unesp.br/server/api/core/bitstreams/ee8add21-69d4-44cf-bb37-d46b92536ba9/content. Acesso em: 19 out. 2024.
WEITZMAN, Tyler. *A short history of text to speech.* [*S. l.*], 27 jun. 2022. Disponível em: https://speechify.com/blog/history-of-text-to-speech/. Acesso em: 20 out. 2024.
YOUNESI, Abolfazl; ANSARI, Mohsen; FAZLI, MohammadAmin; EJLALI, Alireza; SHAFIQUE, Muhammad; HENKEL, Jorg. Neural and Evolutionary Computing; Machine Learning. **A** *Comprehensive Survey of Convolutions in Deep Learning: Applications, Challenges, and Future Trends,* [*s. l.*], 28 fev. 2024. DOI https://doi.org/10.48550/arXiv.2402.15490. Disponível em: https://arxiv.org/abs/2402.15490. Acesso em: 23 out. 2024.