Relatório de Classificação de Componentes de Computador

Gabriel Duarte Santos

Sumário

1 Introdução

Este projeto visa desenvolver um sistema que identifique componentes de computador a partir de imagens, com o intuito de auxiliar tanto no reconhecimento de cada componente quanto na verificação dos itens necessários para montar um computador completo. Com uso de aprendizado de máquina e redes neurais convolucionais, o sistema identifica peças como CPU, GPU, placa-mãe, memória RAM, entre outras, sendo útil em sistemas de inventário ou para organizar peças de hardware.

Para facilitar o uso, uma interface gráfica (GUI) foi implementada para que o usuário possa verificar e listar as peças, confirmando a presença de todos os componentes essenciais para uma montagem completa. O modelo ResNet50V2 foi escolhido para a versão final, com pré-processamento das imagens incluindo data augmentation, e ajustes em parâmetros como batch size e número de épocas para otimização.

2 Montagem da Base de Dados

Um dos primeiros desafios do projeto foi encontrar uma base de dados suficiente para componentes como CPU, GPU, placa-mãe, memória RAM e outros itens essenciais para a montagem de um computador. Abaixo estão os desafios e estratégias utilizadas para compor a base:

2.1 Desafios Enfrentados

- **Limitação de componentes**: As bases de dados disponíveis continham a maioria dos componentes, mas itens como fontes de alimentação e SSDs estavam ausentes. Em virtude disso, não foi possível incluir fontes de alimentação entre as classes de componentes identificados pelo modelo, e adicionei SSDs à classe de HDD.
- **Volume de dados**: As bases encontradas apresentavam um número reduzido de imagens, com média de 350 imagens por classe. Este número exigiu expansão para melhorar o desempenho do modelo, o que levou à adoção de algumas estratégias para aumentar o volume da base.

2.2 Expansão da Base

Para expandir a base de dados:

- **Captura de fotos próprias**: Capturei imagens dos componentes que possuo, aumentando a diversidade de ângulos e condições de iluminação.
- **Coleta manual de imagens na internet**: Dentre as fontes utilizadas, o Unsplash foi a que mais me auxiliou, disponibilizando imagens gratuitas e com facilidade para download.

Com essas etapas, a base de dados passou a ter 500 imagens por classe, o que contribuiu para a melhora no desempenho do modelo ao fornecer uma amostra mais variada e representativa. Caso fosse possível expandir ainda mais o banco de dados de forma eficaz, a precisão do modelo poderia ser potencialmente maior. No entanto, dentro das limitações de tempo e disponibilidade de imagens gratuitas, essa foi a abordagem mais viável.

3 Pré-Processamento de Dados

Antes de iniciar o treinamento do modelo, realizei algumas etapas de pré-processamento, essenciais para garantir a consistência e a qualidade das imagens:

- **Redimensionamento**: Todas as imagens foram redimensionadas para uma resolução padrão (224x224), garantindo que todas as entradas tivessem o mesmo formato.
- **Normalização**: Valores dos pixels entre 0 e 1 para facilitar o aprendizado.
- **Data Augmentation**: Buscando resolver o problema com a limitação de volume de dados, utilizei técnicas de data augmentation para aumentar virtualmente a base e aprimorar a robustez do modelo: Aplicando transformações como flip horizontal, rotação, zoom e translação para variações visuais e generalização do modelo.

4 Modelos Testados

Diferentes arquiteturas foram testadas ao longo do projeto, a fim de selecionar o modelo mais adequado para a tarefa de classificação de peças de computador. Abaixo estão as principais tentativas e os resultados observados:

4.1 Modelo Simples com Camadas Convolucionais

Inicialmente, foi desenvolvido um modelo simples com camadas convolucionais empilhadas, mas que resultou em baixa precisão e velocidade devido à capacidade limitada para extrair características visuais complexas. No entanto, foi muito útil para compreender melhor a tarefa de classificação.

4.2 VGG16

Foi possível melhorar a precisão em comparação ao modelo inicial, mas o tempo de execução foi um ponto crítico. Mesmo com uma boa precisão, o modelo VGG16 não diferenciou com exatidão todas as classes, indicando limitações em sua capacidade para este conjunto de dados específico.

4.3 EfficientNet

Oferece uma proposta otimizada para balancear eficiência e precisão. No entanto, ao tentar reaproveitar as mesmas configurações do código anterior do VGG16, os resultados iniciais foram muito insatisfatórios, levando à procura de outro modelo.

4.4 ResNet50V2 (Modelo Final)

Selecionado como arquitetura final, o modelo ResNet50V2 obteve uma precisão equivalente à do VGG16, mas com uma otimização muito superior em termos de tempo de execução. Após ajustes finais, o modelo alcançou uma boa acurácia, equilibrando precisão e desempenho.

5 Treinamento e Avaliação do Modelo

- **Batch Size**: Foi utilizado 32, equilibrando a eficiência do processamento com a estabilidade do aprendizado.
- **Épocas**: Treinado por 20 épocas com *Early Stopping* identificando quando o treinamento começasse a estabilizar.
- **Otimizador**: Adam, com learning rate inicial de 0.001, ajustado ao longo do treinamento para melhorar a convergência.

O modelo atingiu 90% de acurácia geral, com ótimo desempenho nas classes principais, mas algumas confusões em componentes visualmente similares. A Tabela 1 mostra a acurácia detalhada por classe.

Tabela 1: Acurácia por classe do modelo final.

Classe	**Acurácia (%)**
Gabinete	88
CPU	88
GPU	92
HDD/SSD	87
Headset	97
Teclado	94
Monitor	91
Placa-mãe	92
Mouse	91
Memória RAM	94

6 Métricas de Avaliação

Para avaliar o desempenho do modelo final, foram utilizadas métricas de precisão detalhadas por classe, incluindo o F1-Score e o Recall, além de uma matriz de confusão. Esses indicadores permitem uma compreensão mais profunda do quão bem o modelo distingue entre diferentes classes de componentes de hardware.

6.1 F1-Score e Recall por Classe

A Figura 1 apresenta os valores de F1-Score e Recall para cada classe de componente. Como é possível observar, a maioria das classes apresenta bons valores, indicando que o modelo é efetivo em classificar corretamente os componentes na maioria das vezes. Para o F1-Score, é possível observar que as classes *Headset* e *Mouse* apresentam os valores mais altos, indicando que o modelo teve maior facilidade em identificar corretamente esses componentes. Porém, as outras classes também tiveram bons desempenhos, exceto pela classe *Case*, que teve um valor um pouco mais baixo em comparação com as demais.

Em relação ao Recall, a maioria das classes também alcança bons resultados, revelando que o modelo é eficaz em identificar a maioria dos componentes corretamente sem omitir muitos verdadeiros positivos.

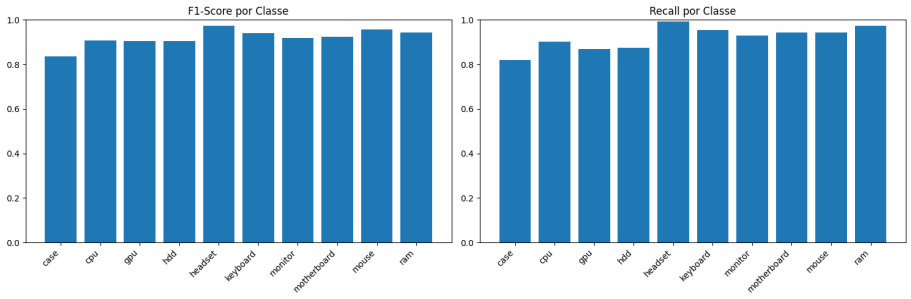

Figura 1: F1-Score e Recall por Classe do Modelo Final.

6.2 Matriz de Confusão

A Figura 2 apresenta a matriz de confusão, que mostra os acertos e erros de classificação. A matriz demonstra que, embora o modelo tenha alcançado uma alta acurácia, algumas classes, como *Case* e *HDD*, foram confundidas com outras em alguns casos. Algumas confusões que o modelo comete podem ser explicadas pela presença de componentes semelhantes ou que aparecem juntos em muitas imagens. Por exemplo, muitas fotos de Cases mostram também a Placa Mãe instalada dentro deles, o que pode levar o modelo a confundir essas duas classes. Além disso, componentes como HDD e CPU têm aparência semelhante em alguns ângulos, o que dificulta a distinção.

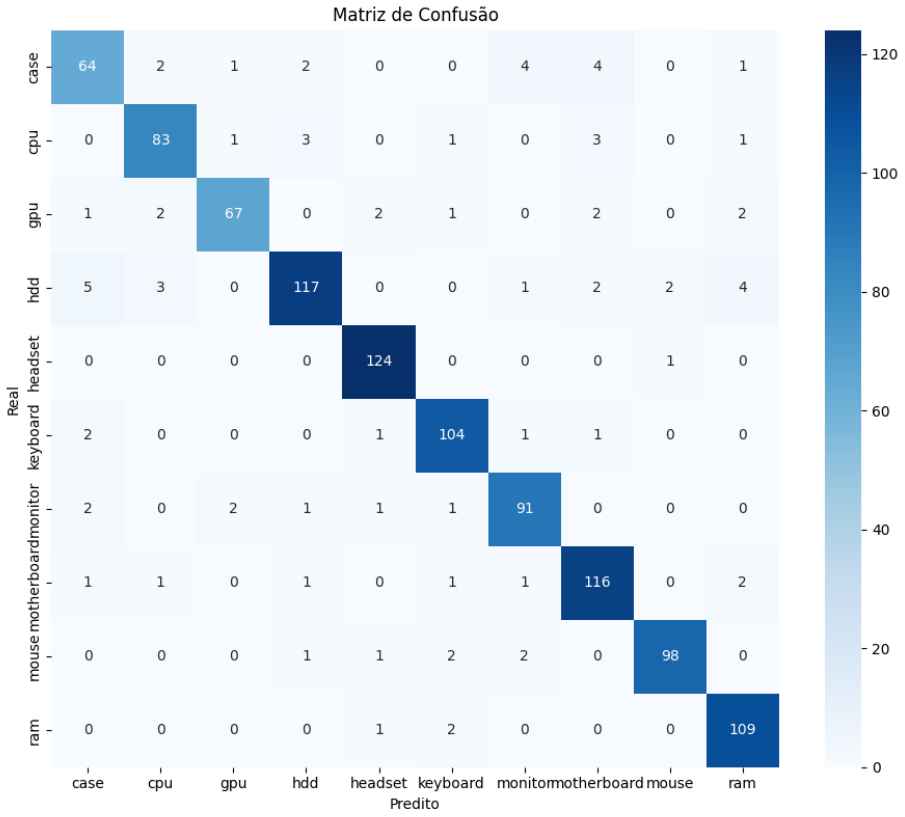

Figura 2: Matriz de Confusão para o modelo final.

6.3 Análise dos Resultados

Com base nos gráficos de F1-Score, Recall e na matriz de confusão, concluise que o modelo possui uma performance muito boa, com uma acurácia geral de 90%, sendo mais preciso na identificação das classes mais distintas visualmente, como *RAM*, *Headset* e *Keyboard*. No entanto, há uma leve dificuldade na classificação de componentes com aparência similar, como *CPU* e *HDD*. Esse resultado mostra a capacidade do modelo em lidar com a maioria das classes, mas que um aumento na diversidade de imagens para classes problemáticas poderia reduzir os erros de classificação.

Essas métricas contribuíram para a escolha do modelo ResNet50V2 como a solução final deste projeto, por conta de sua alta precisão e capacidade de generalizar bem para as diferentes classes de componentes de hardware.

7 Interface Gráfica de Usuário (GUI)

A aplicação desenvolvida inclui uma interface gráfica de usuário (GUI) para facilitar a verificação e o controle dos componentes de hardware necessários para a montagem de um computador.

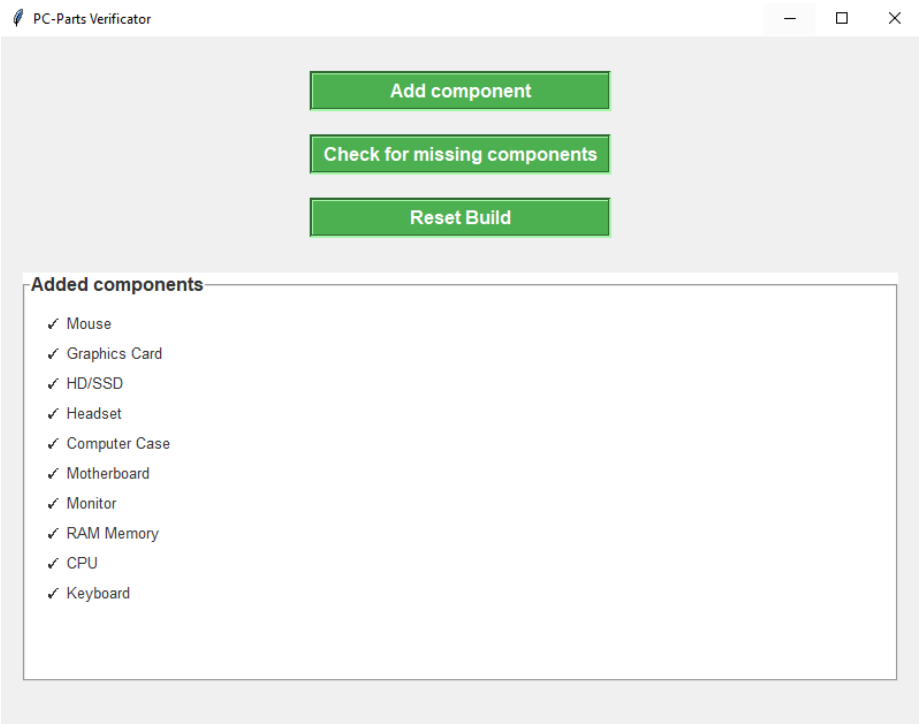

Figura 3: Interface Gráfica de Usuário (GUI).

Esta interface permite que o usuário adicione imagens de componentes, verifique a presença dos itens essenciais, e obtenha um relatório indicando se o conjunto está completo ou se ainda faltam peças. As funcionalidades incluem:

- **Adição e Classificação de Componentes**: O sistema utiliza o modelo treinado para identificar e classificar as imagens dos componentes inseridos. As imagens são redimensionadas e normalizadas para garantir uma classificação precisa. Caso o modelo identifique o componente com uma confiança mínima (ajustável), ele é adicionado à lista de componentes; caso contrário, uma mensagem de erro é exibida.
- **Verificação de Componentes Faltantes**: Outra funcionalidade é a verificação de se todos os componentes essenciais e opcionais foram adicionados ao sistema. Após a análise, uma lista é exibida com os componentes

obrigatórios e opcionais que ainda faltam, incluindo uma descrição de cada um para auxiliar o usuário a entender sua importância.

- **Reinicialização da Verificação (Resetar a Build)**: O usuário tem a opção de reiniciar o processo de verificação, removendo todos os componentes adicionados e restaurando o estado inicial da interface.

8 Exemplo de Uso

A interface gráfica juntamente com o modelo treinado facilita a identificação e organização das peças de um computador, auxiliando iniciantes, evitando erros comuns entre peças que várias pessoas se confundem, como CPU e Gabinete. Além disso, a verificação dos componentes ajuda a assegurar que o sistema contará com todos os elementos necessários para um funcionamento completo e eficiente.

9 Possíveis Melhorias

Com o objetivo de melhorar a acurácia do modelo, uma possível extensão seria o aumento contínuo da base de dados, adicionando mais imagens variadas para as classes que apresentam maior índice de confusão. Outra ideia para incrementar a funcionalidade do sistema, embora não faça parte dos objetivos principais deste projeto, seria incluir a possibilidade de o usuário informar o modelo específico de cada componente adicionado. Dessa forma, o sistema poderia realizar verificações de compatibilidade, por exemplo, entre a placa-mãe e o processador ou memória RAM. O que prorcionaria uma experiência mais completa, especialmente em montagem de computadores onde a compatibilidade é essencial.

10 Conclusão

Este projeto resultou em um sistema eficiente para a identificação de componentes de computador a partir de imagens, utilizando aprendizado de máquina e redes neurais convolucionais. A escolha do modelo ResNet50V2 como arquitetura final foi ótimo para atingir um equilíbrio entre precisão e velocidade, permitindo uma classificação confiável dos principais componentes, mesmo com a limitação de algumas classes e de uma base de dados de tamanho moderado. A criação de uma interface gráfica (GUI) para o usuário trouxe um valor e praticidade muito interessante para o sistema, permitindo uso prático em montagens e inventário de hardware. Este projeto demonstrou que a combinação de redes neurais avançadas e uma GUI bem desenvolvida oferece uma ferramenta interessante para o reconhecimento e organização de peças de hardware, proporcionando um suporte eficaz tanto na montagem quanto no gerenciamento de inventário de computadores.

Referências

[1] Unsplash. “Free Images for Everyone.” Acesso em: https://unsplash.com

[2] Kaggle. “Datasets for Data Science and Machine Learning.” Acesso em: https://www.kaggle.com/datasets/asaniczka/pc-parts-images-dataset-classification

[3] Google Dataset Search. “A ferramenta de busca de conjuntos de dados.” Acesso em: https://datasetsearch.research.google.com

[4] Google Imagens. “Plataforma de pesquisa de imagens.” Acesso em: https://images.google.com

[5] Stack Overflow. “Plataforma de perguntas e respostas para desenvolvedores.” Acesso em: https://stackoverflow.com