Desenvolvimento de um Sistema Web para Reconhecimento de Imagens Utilizando Comunicação entre Módulos Distribuídos

Victor Marques de Souza, Vittória Dutra Teixeira Pirró

Pontíficia Universidade Católica de Minas Gerais - Poços de Caldas – MG – Brasil

victorms4@hotmail.com, vittoriadtp@gmail.com

Abstract *This article describes the development and integration of a web-based image recognition system. The solution involves three main modules: a frontend for user interaction, a backend implemented in Node.js for request handling, and a Python server using TensorFlow for image classification. Communication between these modules is achieved through REST APIs, enabling a seamless flow of data from image upload to prediction result presentation. This documentation details the technical aspects of integration, highlighting the architecture, data preprocessing, and inter-module communication processes.*

Resumo *Este artigo descreve o desenvolvimento e a integração de um sistema de reconhecimento de imagens baseado na web. A solução envolve três módulos principais: um frontend para interação com o usuário, um backend implementado em Node.js para gerenciamento de requisições e um servidor em Python utilizando TensorFlow para classificação de imagens. A comunicação entre os módulos é realizada por meio de APIs REST, permitindo o fluxo contínuo de dados desde o upload da imagem até a apresentação do resultado da predição. Esta documentação aborda os aspectos técnicos da integração, destacando a arquitetura, o pré-processamento de dados e os processos de comunicação entre os módulos.*

1. Introdução

O reconhecimento de imagens é uma aplicação crescente no campo da inteligência artificial, com usos variados, desde segurança e saúde até entretenimento. Este artigo apresenta o desenvolvimento de um sistema que possibilita aos usuários fazer *upload* de imagens para um *website*, que, por sua vez, processa essas imagens e retorna informações sobre o conteúdo.

O sistema foi projetado como uma aplicação modular, com um *frontend* em HTML, CSS e JavaScript, um *backend* em Node.js, e um servidor Python equipado com TensorFlow para realizar as classificações. Este artigo detalha como os módulos foram conectados, garantindo uma comunicação eficiente e um processamento robusto.

2. Levantamento Bibliográfico

No artigo *Convolutional Deep Belief Networks on CIFAR-10* [Alex Krizhevsky 2010], é explorado o treinamento de redes de crença profunda convolucionais (C*onvolutional Deep Belief Networks* - DBNs) aplicadas ao conjunto de dados CIFAR-10. A abordagem inclui uma etapa de pré-treinamento não supervisionado em um subconjunto maior de imagens antes de um ajuste fino supervisionado. Os métodos propostos enfrentam desafios como a modelagem de bordas de imagens e problemas de *overfitting*, introduzindo unidades ReLU modificadas e técnicas de regularização para melhorar o desempenho. A solução alcançou resultados de ponta na época, demonstrando a eficácia de modelos convolucionais com inicialização pré-treinada para tarefas de classificação de imagens.

O artigo A*n Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale* [Alexey Dosovitskiy et al. 2021], introduz o *Vision Transformer* (ViT), que substitui completamente as convoluções por atenção em tarefas de reconhecimento de imagens. Aplicado ao CIFAR-10 e outros *benchmarks*, o ViT divide imagens em blocos, trata-os como sequências de tokens e aplica diretamente transformadores para classificação. Os resultados demonstram que, com pré-treinamento em grandes conjuntos de dados, o ViT supera arquiteturas convolucionais tradicionais, destacando a eficácia da abordagem baseada em transformadores em problemas de visão computacional em larga escala.

O estudo *Reduction of Class Activation Uncertainty with Background Information* [H.M. Dipu Kabir 2024], propõe uma nova metodologia para melhorar a generalização em modelos de aprendizado profundo, introduzindo uma classe de fundo que ajuda na redução da incerteza nas ativações de classe. Aplicado ao CIFAR-10 e outros conjuntos de dados, o método mostrou-se eficiente em modelos baseados em *Vision Transformers* (ViTs), combinando transferência de aprendizado e aprendizado multitarefa. A abordagem resultou em menor custo computacional e melhor desempenho em comparação com métodos tradicionais, destacando-se como uma alternativa promissora para cenários de recursos limitados.

3. Desenvolvimento

3.1. Arquitetura Geral

O sistema foi desenvolvido com uma arquitetura cliente-servidor, composta por três camadas principais:

- ***Frontend***: Interface gráfica onde o usuário pode carregar imagens e visualizar os resultados das predições;
- ***Backend* (Node.js)**: Responsável por receber os uploads do frontend, encaminhar as imagens para o servidor Python e retornar os resultados ao usuário;

- **Servidor Python**: Processa as imagens enviadas, realiza as predições utilizando um modelo de aprendizado profundo e retorna os resultados ao *backend*.

3.2. Comunicação entre os módulos

A comunicação entre os módulos é feita via APIs REST:

- ***Frontend* para Backend**: As imagens são enviadas em requisições POST, utilizando o *middleware* Multer para gerenciar os uploads;
- ***Backend* para Servidor Python**: O *backend* encaminha as imagens para o servidor Python utilizando o módulo Axios, que realiza requisições HTTP com as imagens no corpo da requisição;
- **Servidor Python para *Backend***: O servidor Python retorna os resultados da predição (classe e confiança) em formato JSON, que o backend processa e repassa ao *frontend*.

3.3. Funcionamento dos módulos

O *frontend* é desenvolvido em HTML e JavaScript, ele apresenta um formulário simples para *upload* de imagens. A exibição dos resultados é feita dinamicamente, com uma interface limpa e responsiva.

O *backend* é desenvolvido em Node JS, ele é implementado para servir tanto arquivos estáticos, quanto gerenciar as requisições enviadas pelo frontend. Utiliza Multer para manipular os uploads e Axios para comunicação com o servidor Python.

O servidor Python carrega um modelo de classificação de imagens pré-treinado utilizando TensorFlow e o conjunto de dados CIFAR-10, que foi desenvolvido pelo *Canadian Institute For Advanced Research* (CIFAR). Este conjunto de dados possui dez classes, sendo elas, avião, carro, pássaro, gato, veado, cachorro, sapo, cavalo, navio, caminhonete. E como retorno para o *backend*, ele envia a classe predita e a confiança associada em formato JSON.

3.4. Fluxo de Dados

1. O usuário faz *upload* de uma imagem no *frontend*.
2. O *backend* recebe a imagem e a envia para o servidor Python.
3. O servidor Python processa a imagem, realiza a predição e retorna os resultados ao *backend*.
4. O *backend* transmite os resultados para o *frontend*, onde são exibidos ao usuário.

4. Considerações Finais

Este artigo apresentou o desenvolvimento de um sistema de reconhecimento de imagens integrado, destacando o processo de comunicação entre os módulos. O sistema demonstrou ser eficaz ao permitir que os usuários obtenham resultados rápidos e precisos em uma interface amigável.

Futuras melhorias incluem a integração de suporte a múltiplos modelos de reconhecimento e a expansão da funcionalidade para lidar com *uploads* em lote. A modularidade do projeto facilita a adaptação a novos requisitos e tecnologias, promovendo escalabilidade e manutenção.

Referências

Krizhevsky, A. (2010) "C*onvolutional Deep Belief Networks on CIFAR-10*", *University of Toronto*, Canada.

Kabir, H. M. D. (2024) "*Reduction of Class Activation Uncertainty with Background Information*", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, DOI: 10.1109/CVPR.2024.01234.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2021) "*An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale*", In *International Conference on Learning Representations (ICLR)*

ProgrammingKnowledge (2023) *Python Flask Tutorial - FULL COURSE*. Disponível em: https://www.youtube.com/watch?v=2YOBmELm_v0. Acesso em: 30 de Novembro 2024.

FreeCodeCamp.org (2019) *Learn Flask for Python - Full Tutorial*. Disponível em: https://www.youtube.com/watch?v=Z1RJmh_OqeA. Acesso em: 30 de Novembro de 2024.

Corey Schafer (2018) *Python Flask Tutorial: Full-Featured Web App Part 1 - Getting Started*. Disponível em: https://www.youtube.com/watch?v=MwZwr5Tvyxo. Acesso em: 30 de Novembro de 2024.