

Trabalho de Conclusão II

**PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

***MOVETAGGER: UMA APLICAÇÃO PARA RECONHECIMENTO
DE FILMES POR MEIO DE IMAGENS***

**GUILHERME WEBBER MENDES
JOÃO VITOR DE ALBUQUERQUE MACHADO**

Trabalho de Conclusão apresentado
como requisito parcial à obtenção do
grau de Bacharel em Sistemas de
Informação na Pontifícia Universidade
Católica do Rio Grande do Sul.

Orientador: Prof. Duncan Ruiz

**Porto Alegre
2019**

MOVETAGGER: UMA APLICAÇÃO PARA RECONHECIMENTO DE FILMES POR MEIO DE IMAGENS

RESUMO

A indústria cinematográfica representa um dos principais segmentos da economia, da cultura e do entretenimento contemporâneo. No cinema, é comum a produção de filmes em franquias, sequências, *spin-offs* e até filmes de mesmo gênero que possuem enredo e fotografia semelhantes e podem não ser distinguidos caso o espectador não possua um conhecimento profundo das obras. Este trabalho tem como objetivo desenvolver uma aplicação móvel capaz de auxiliar o usuário na identificação de filmes através de imagens. Para isso serão utilizadas técnicas de *deep learning* em um processo de recuperação de imagens baseado em conteúdo (CBIR) para identificar representações nas fotos submetidas e encontrar similaridades nos *frames* de filmes armazenados em um banco de dados.

Palavras-Chave: Aprendizado profundo; Redes Neurais Convolucionais; recuperação de imagens baseado em conteúdo; pré-processamento de vídeo; .

ABSTRACT

The film industry represents one of the main segments of the culture, economy and contemporary entertainment. In movies, it's common the production of films franchises, sequences, spin-offs and even films of the same genre which have a similar plot and photo and can be not recognized if the viewer do not have a deep knowledge of the area. The objective of this work is to develop a mobile application capable to assist the user in identify movies through images. For such, deep learning algorithms on a content based image retrieval (CBIR) process will be used to identify representations on submitted photos and find similarities in frames of downloaded on a database.

Keywords: *Deep Learning; Convolutional Neural Networks; content based image retrieval; video preprocessing;*

LISTA DE FIGURAS

| | |
|--|----|
| Figura 2.1 – Imagem explicando o funcionamento dos quadros por segundo. | 15 |
| Figura 2.2 – Estrutura de uma RNA simples e de uma RNA utilizando Aprendizado profundo. | 16 |
| Figura 2.3 – Exemplo de rede neural com múltiplas camadas de convolução. | 17 |
| Figura 2.4 – Imagem representando uma arquitetura CBIR | 19 |
| Figura 2.5 – Imagem demonstrando o método de identificação de imagens baseada em conteúdo | 20 |
| Figura 2.6 – <i>Frame</i> do GIF de demonstração da aplicação The Take, em que mostra o reconhecimento de objetos. | 21 |
| Figura 2.7 – <i>Frame</i> do GIF de demonstração da aplicação The Take, em que mostra a aplicação identificado os objetos. | 21 |
| Figura 2.8 – <i>Frame</i> do GIF de demonstração da aplicação The Take, em que mostra os objetos similares encontrados no banco de dados. | 22 |
| Figura 2.9 – <i>Frame</i> do GIF de demonstração da aplicação The Take, em que mostra como realizar a compra do objeto detectado. | 22 |
| Figura 2.10 – Imagem de um espectrograma. | 23 |
| Figura 2.11 – Imagem do aplicativo Shazam para IOS. Aplicativo “ouvindo” o som ambiente. | 23 |
| Figura 2.12 – Imagem do aplicativo Shazam para IOS. Aplicativo mostrando a música encontrada. | 24 |
| Figura 4.1 – Modelagem da arquitetura final da solução | 26 |
| Figura 4.2 – Modelo de dados das coleções do banco de dados | 27 |
| Figura 5.1 – Tela inicial da aplicação MovieTagger. | 31 |
| Figura 5.2 – Imagem da câmera do telefone fotografando uma cena do trailer do filme Star Wars Wars Episódio V: O Império Contra-Ataca | 31 |
| Figura 5.3 – Tela inicial da aplicação MovieTagger após uma foto ser capturada pela câmera do telefone, com a imagem do trailer do filme Star Wars Wars Episódio V: O Império Contra-Ataca fotografada. | 32 |
| Figura 5.4 – Tela inicial da aplicação MovieTagger após a imagem capturada ser enviada para a API da solução, constando uma mensagem de carregamento até a resposta da API. | 33 |
| Figura 5.5 – Tela secundária da aplicação MovieTagger após o reconhecimento da imagem fotografada, em que detalhes do filme descoberto são mostradas. | 33 |

| | |
|--|----|
| Figura 5.6 – Tela secundária da aplicação MovieTagger após o reconhecimento da imagem fotografada, em que detalhes do filme descoberto são mostradas junto de detalhes de outros possíveis filmes descobertos. | 34 |
| Figura 5.7 – Arquitetura da rede neural convolucional InceptionV3 apontando a camada utilizada para extrair o vetor de características. | 35 |
| Figura 5.8 – Fórmula de distância de cosseno utilizada para medir similaridade entre dois vetores. | 35 |
| Figura 6.1 – Imagem contendo o <i>frame</i> do trailer do filme Star Wars Episódio: Ataque dos Clones, submetido diretamente à API e suas três imagens mais similares. | 38 |
| Figura 6.2 – Imagem contendo o <i>frame</i> do trailer do filme Star Wars Episódio VI: O Retorno de Jedi, submetido diretamente à API e suas três imagens mais similares. | 39 |
| Figura 6.3 – Imagem contendo o <i>frame</i> do trailer do filme Star Trek V: A última Fronteira, submetido diretamente na API e suas três imagens mais similares. | 40 |
| Figura 6.4 – Imagem contendo o pôster do filme Star Trek II: A Ira de Khan, submetido diretamente à API e suas três imagens mais similares. | 41 |
| Figura 6.5 – Imagem contendo o pôster do filme Star Wars V: O Império Contra-Ataca, submetido diretamente à API e suas três imagens mais similares. . . | 42 |
| Figura 6.6 – Imagem contendo telas da aplicação móvel em que uma foto do trailer do filme Star Wars Episódio III: A Vingança dos Sith foi capturada . . | 43 |
| Figura 6.7 – Imagem contendo telas da aplicação móvel em que uma foto do trailer do filme Star Trek I: O Filme foi capturada | 44 |
| Figura 6.8 – Imagem contendo telas da aplicação móvel em que uma foto do trailer do filme Star Wars Episódio IV: Uma Nova Esperança foi capturada . | 45 |

LISTA DE TABELAS

Tabela 2.1 – Quantidade de imagens por filmes presentes no banco de dados. . . 14

LISTA DE SIGLAS

FPS – *Frames per second*

RNA – Rede Neural Artificial

RNC – Rede Neural Convolucional

API – *Application Programming Interface*

HTTP – *Hypertext Transfer Protocol*

REST – *Representational State Transfer*

JSON – *JavaScript Object Notation*

LISTA DE ABREVIATURAS

CBIR. – *Content-based image retrieval*

SBIR. – *Semantic-based image retrieval*

MVP. – *Minimum Viable Product*

SUMÁRIO

| | | |
|----------|--|-----------|
| 1 | INTRODUÇÃO | 11 |
| 2 | CARACTERIZAÇÃO DO PROBLEMA | 13 |
| 2.1 | CENÁRIO DE ESTUDO | 13 |
| 2.1.1 | TRATAMENTO DE QUADROS DE FILMES | 14 |
| 2.1.2 | APRENDIZADO DE MÁQUINA | 15 |
| 2.1.3 | <i>CBIR</i> | 18 |
| 2.2 | TRABALHOS SIMILARES | 19 |
| 2.2.1 | DEEP LEARNING OF BINARY HASH CODES FOR FAST IMAGE RETRIEVAL [LYHC15] | 19 |
| 2.2.2 | THETAKE [The18] | 20 |
| 2.2.3 | SHAZAM [Sha18] | 22 |
| 2.3 | IDENTIFICAÇÃO DA OPORTUNIDADE | 24 |
| 3 | OBJETIVOS | 25 |
| 3.1 | OBJETIVO GERAL | 25 |
| 3.2 | OBJETIVOS ESPECÍFICOS | 25 |
| 4 | PROJETO | 26 |
| 4.1 | ARQUITETURA DA SOLUÇÃO | 26 |
| 4.1.1 | <i>FRAMES</i> DOS TRAILERS DOS FILMES | 26 |
| 4.1.2 | MODELO DE DADOS | 27 |
| 4.1.3 | SERVIDOR | 27 |
| 4.1.4 | CLIENTE | 28 |
| 4.1.5 | CONEXÃO CLIENTE SERVIDOR | 28 |
| 4.1.6 | ESTRUTURA DE IDENTIFICAÇÃO | 29 |
| 4.1.7 | EXTRAÇÃO DE <i>FRAMES</i> DE VALIDAÇÃO | 29 |
| 5 | SOLUÇÃO | 30 |
| 5.1 | APLICAÇÃO MÓVEL | 30 |
| 5.2 | SERVIDOR | 34 |
| 5.3 | SUBMISSÃO DE NOVOS FILMES | 36 |

| | | |
|----------|--|-----------|
| 6 | TESTES DA SOLUÇÃO E RESULTADOS | 37 |
| 6.1 | SUBMISSÃO DIRETA DE IMAGENS DE VALIDAÇÃO | 37 |
| 6.2 | SUBMISSÃO DE PÔSTERES | 40 |
| 6.3 | SUBMISSÃO DE FOTOS VIA APLICATIVO | 42 |
| 7 | CONSIDERAÇÕES FINAIS | 46 |
| | REFERÊNCIAS | 47 |
| | APÊNDICE A – Estrutura JSON retornada pelo servidor | 49 |

1. INTRODUÇÃO

Este documento descreve uma solução que utiliza um processo de recuperação de imagens baseado em conteúdo (CBIR) [EGE⁺99] utilizando *deep learning* [GBC16] para fazer a extração das características da imagem e um algoritmo de busca de imagem por similaridade que torne a aplicação escalável. Esta solução em forma de aplicativo de *smartphone* chamada de MovieTagger poderá ser utilizada por um público geral e ajudar na identificação de obras por meio de imagens retiradas de filmes, cartazes e cenas que não possuem dados de identificação claros para o usuário.

Tendo em vista que na atualidade, filmes se tornaram muito populares e são constantemente lançados, apesar das inúmeras variações percebe-se que acabam por existir filmes com conceitos, cenas e figurinos parecidos. A partir disso, e, sabendo que filmes são compostos de imagens em sequência, e com o avanço da capacidade computacional tornou-se possível a criação de algoritmos de reconhecimento visual de imagens por meio de métodos de *deep learning* para identificar esses padrões, foi realizada uma breve pesquisa a fim de encontrar uma solução que possibilitasse o reconhecimento de um filme por apenas uma de suas imagens e, não obtendo sucesso na pesquisa, reconheceu-se que há um nicho novo a ser explorado.

O reconhecimento do conteúdo de uma imagem por computação, é um problema desafiador que vem sendo pesquisado nas últimas décadas. Utilizando métodos de *deep learning*, pesquisas vêm sendo realizadas a fim de se obter um bom resultado na aplicação de algoritmos de reconhecimento visual na identificação de imagens. O reconhecimento se dá em sua maioria pela identificação do conteúdo das imagens (CBIR) ou baseado em semântica (SBIR) [ALE14].

O termo “recuperação de imagens baseado em conteúdo” tem sido usado para descrever o processo de recuperação de imagens a partir de uma coleção com base em suas características. As técnicas, ferramentas e algoritmos utilizados são originários de várias áreas, como estatísticas, reconhecimento de padrões, processamento de sinais e visão computacional [20116]. Dentre as principais técnicas utilizadas estão o histograma, a identificação por meio de formas e a identificação por meio de textura.

Portanto, este documento descreve um projeto que utiliza algoritmo de reconhecimento visual para fazer a identificação de filmes por imagens representativas através da utilização de uma base de dados contendo *frames* de filmes. Porém, na solução desenvolvida, foi utilizado *frames* de trailers, pois filmes possuem direitos autorais. A solução desenvolvida possui uma base de dados contendo esses *frames*, pré processados pela *InceptionV3* [SVI⁺15] utilizando os pesos iniciais como base da ImageNet e a criação de uma aplicação capaz de realizar a identificação por meio da captura de uma imagem.

Este trabalho está organizado da seguinte forma. Parte 2 descreve a caracterização do problema, apresentando o cenário de estudo, temas abordados neste trabalho e trabalhos similares. Parte 3 descreve os objetivos do trabalho. Parte 4 apresenta o projeto da solução proposto. Parte 5 a solução desenvolvida com base na proposta. Parte 6 traz os resultados e testes do sistema e por fim a parte 7 apresenta as considerações finais sobre este projeto.

2. CARACTERIZAÇÃO DO PROBLEMA

Neste capítulo serão apresentadas as características do problema de reconhecimento de contexto de imagem de filmes, as dificuldades encontradas no decorrer do projeto e a diferença das soluções já existentes.

2.1 Cenário de estudo

Filmes são produções audiovisuais onde existe uma sucessão de imagens acompanhadas por sons. A arte de fazer filmes é uma das principais manifestações artísticas do mundo moderno sendo considerada uma das sete artes, junto da música, artes cênicas, pintura, escultura, arquitetura e literatura.

Tendo em vista a grande popularização deste tipo de obra nos dias de hoje e a quantidade crescente de lançamentos anuais, alguns gêneros acabam por ter semelhanças em diversos aspectos que, sem um conhecimento profundo sobre as obras, tendem a gerar dificuldade quando há a necessidade de identificá-las e diferenciá-las.

Logo, a fim de facilitar a identificação de conteúdos provenientes de filmes e descobrir seu longa de origem, a aplicação MovieTagger propõe-se a fazer o reconhecimento visual de filmes pela utilização de métodos de *deep learning* aplicados em uma imagem submetida pelo usuário, retornando o nome e informações básicas do filme reconhecido.

Para testar a solução proposta neste trabalho, e otimizar a capacidade atual de utilização de hardware e de algoritmos, foi utilizado trailers de 6 filmes da franquia Star Wars e 6 filmes da franquia Star Trek. Ao explorar este escopo de trailers, pretendemos mostrar que a solução será capaz de funcionar também para filmes completos. A quantidade de imagens disponíveis para cada filme pode ser vista na Tabela 2.1, estas imagens são resultados de um pré-processamento dos *frames* em que os créditos iniciais e créditos finais foram excluídos manualmente, junto de *frames* totalmente pretos e totalmente brancos. O número de *frames* utilizados é aproximadamente 50% do total de *frames* do filme.

| Filme | Nº Imagens para treinamento | Nº Imagens para validação |
|---|-----------------------------|---------------------------|
| Star Wars Episode IV: A New Hope (1977) | 655 | 72 |
| Star Trek: The Motion Picture (1979) | 692 | 76 |
| Star Wars Episode V: - The Empire Strikes Back (1980) | 1.219 | 134 |
| Star Trek II: The Wrath of Khan (1982) | 1.266 | 140 |
| Star Wars Episode VI: Return of the Jedi (1983) | 1.105 | 121 |
| Star Trek III: The Search for Spock (1984) | 649 | 72 |
| Star Trek IV: The Voyage Home (1986) | 1.343 | 149 |
| Star Trek V: The Final Frontier (1989) | 776 | 86 |
| Star Trek VI: The Undiscovered Country (1991) | 897 | 99 |
| Star Wars Episode I: The Phantom Menace (1999) | 983 | 108 |
| Star Wars Episode II: Attack of the Clones (2002) | 1.156 | 129 |
| Star Wars Episode III: Revenge of the Sith (2005) | 1.430 | 157 |

Tabela 2.1 – Quantidade de imagens por filmes presentes no banco de dados.

Por fim, para solucionar o problema de identificação de filmes sem muitas informações, evitando a busca manual pela internet, e evitando reassistir a um filme por completo para descobrir a que filme o trecho ou imagem visto pertencem, a solução irá automatizar o processo, utilizando duas áreas estudadas: tratamento de quadros de filmes e aprendizado de máquina.

2.1.1 TRATAMENTO DE QUADROS DE FILMES

Os quadros de um filme são conhecidos como fotogramas. Cada quadro é um pedaço do filme registrado como imagem em que alinhando-se em sequências maiores de 11 quadros, conseguimos ter a ilusão de movimentação nas imagens. Desde 1929 a

cadência de gravação de filmes é padronizada em 24 FPS (*frames per second*), ou seja, quando assistimos a um filme estamos vendo em cada segundo cerca de 24 imagens em sequência. Essa taxa para os filmes é suficiente, mas dependendo do efeito que se quer proporcionar, quanto mais quadros por segundo maior a qualidade de movimento que se tem no vídeo resultante.

A Figura 2.1 mostra a diferença visual entre os quadros por segundo (FPS), em que com 3 FPS temos três quadros diferentes de um mesmo carro, dobrando esses quadros por segundo conseguimos reproduzir o dobro de imagens do mesmo carro no mesmo período de 1 segundo, e dobrando novamente conseguimos reproduzir doze imagens de um mesmo carro no mesmo segundo.

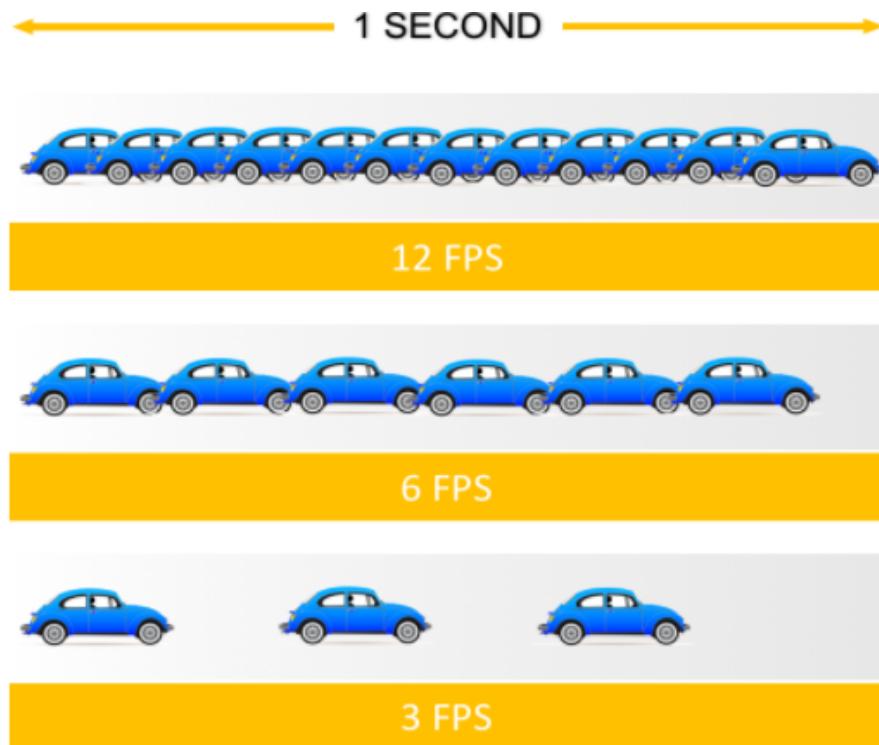


Figura 2.1 – Imagem explicando o funcionamento dos quadros por segundo. *Fonte: Frames Per Second "FPS" explained [Fli16]*

2.1.2 Aprendizado de Máquina

Deep Learning

Deep learning ou aprendizado profundo é uma abordagem de aprendizado de máquina que utiliza métodos baseados em aprendizagem representativa de dados com múltiplos níveis de representação como mostrado na Figura 2.2. Ou seja, ajuda o computador a aprender através de experiências e a entender o mundo em uma hierarquia de conceitos.

Essa hierarquia de conceitos faz com que o computador quebre conceitos complexos em conceitos mais simples, conseguindo assim, adquirir conhecimento sem que um humano codifique as instruções para que se produza um resultado. [GBC16]

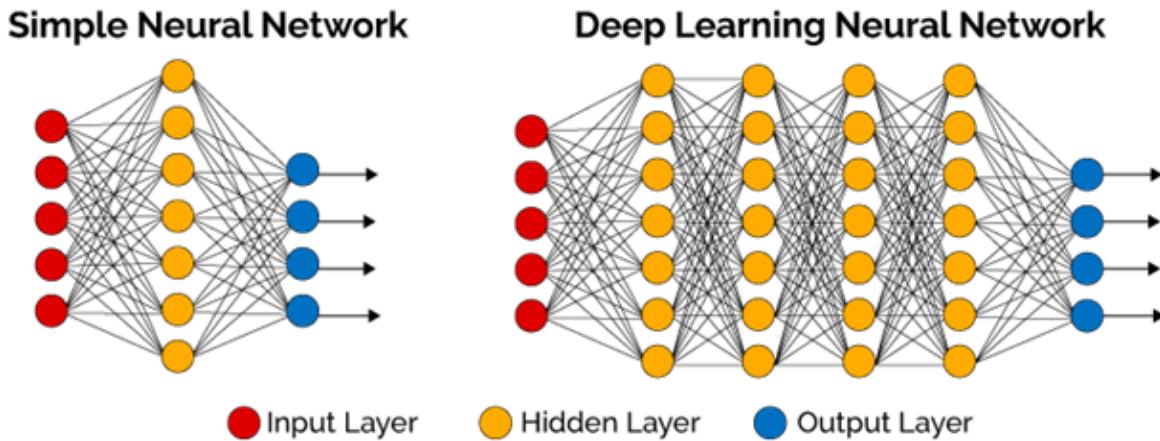


Figura 2.2 – Estrutura de uma RNA simples e de uma RNA utilizando Aprendizado profundo.
Fonte: *Deep learning in digital pathology* [Gom18]

As aplicações modernas de *deep learning* estão sendo utilizadas com sucesso em diversos problemas de aprendizado supervisionado, tendo bastante destaque na área de visão computacional. A variação de aplicações no campo é vasta, porém grande parte dela é focada em imitar capacidades humanas, como o reconhecimento e detecção de objetos e rostos.

Redes Neurais Artificiais

As redes neurais artificiais ou simplesmente RNA's são estruturas projetadas para modelar a forma como um cérebro humano realiza uma tarefa particular utilizando uma interligação maciça de células computacionais simples, denominadas de “neurônios” ou unidades de processamento. Esses neurônios funcionam recebendo n entradas de estímulos (variáveis) que tem seu valor ajustado a um peso atribuído e passam por uma função de ativação, que define o valor da saída da unidade. A saída de um neurônio pode estar ligada a entrada de outro, formando assim uma rede.

O conhecimento de uma rede neural acontece a partir de seu ambiente por intermédio do processo de aprendizagem. Esse conhecimento é sintetizado na forma de pesos sinápticos que modificam o valor de entrada das variáveis do neurônio (pesos).

A arquitetura de uma RNA é composta por três componentes. São eles:

- Camada de entrada
- Camada oculta

- Camada de saída

A camada de entrada possui esse nome por ser onde os neurônios recebem as variáveis externas e se inicia o processamento. A camada oculta são os neurônios intermediários que possuem como entrada a saída de outros neurônios. Por fim, a camada de saída é a que apresenta o resultado, ou seja, classe em que o dado foi classificado.

Quando usamos uma RNA *feedforward* (a ligação entre os neurônios não é cíclica) a informação inicial propaga da primeira camada pelas camadas ocultas até a saída. Durante o treinamento, esse processo pode continuar até produzir um custo escalar. O algoritmo de *backpropagation* permite que a informação do custo propague de volta pela RNA para recalcular os gradientes, ajustando os pesos e fazendo a melhoria da classificação.

Redes Neurais Convolucionais

Redes Convolucionais são RNA's que utilizam convolução ao invés de uma multiplicação de matrizes pelo menos em uma de suas camadas e são utilizadas para dados que possuem uma topologia estilo grade. Essas redes são mais fáceis de treinar, generalizam melhor do que outras e estão sendo usadas amplamente no campo de visão computacional. Um exemplo de como a técnica funciona pode ser encontrado na Figura 2.3.

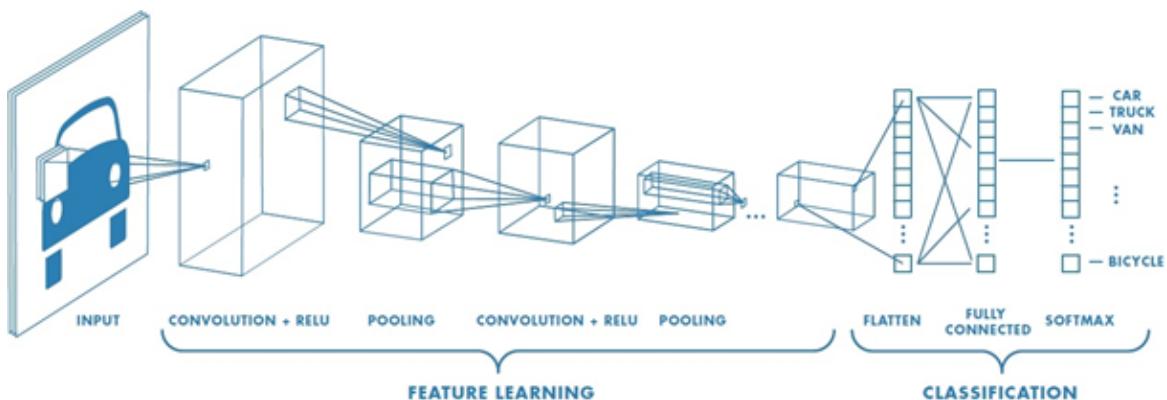


Figura 2.3 – Exemplo de rede neural com múltiplas camadas de convolução. Fonte: *Convolutional Neural Network* [Mat17]

O processo de convolução ou correlação é uma operação feita sobre duas funções de valores reais. Em imagens, trata-se do deslocamento de um filtro bidimensional sobre as camadas da imagem de entrada para calcular o produto vetorial entre as entradas do filtro e da imagem. Esse processo é repetido várias vezes, deslocando o filtro e calculando. O resultado final da convolução são imagens chamadas *feature maps*.

O ponto chave da camada de convolução em RNCs é que os filtros são compostos de parâmetros que podem ser aprendidos. Estes parâmetros podem modelar o compor-

tamento dos filtros para que cada um seja ativado ao ver determinado comportamento na imagem de entrada, como contornos, formas e até objetos.[FdS16]

Transfer Learning

Transfer Learning é um método usado em *deep learning* onde um modelo construído para realizar uma tarefa é reusado como base para criar um segundo modelo. Esse método é bastante usado nas áreas de linguagem natural e visão computacional dado que a utilização dos modelos pré-treinados economiza muito do tempo e uso de recursos computacionais utilizados para treinar uma rede além de apresentar maiores performances quando utilizados com o mesmo propósito, visto que o segundo modelo refina os resultados do primeiro.

Em visão computacional, são comuns os desafios onde são usadas redes pré-treinadas para classificação de imagens em grandes números de classes, como é o caso dos desafios propostos pelo *ImageNet*. Os modelos gerados pelos pesquisadores que participam dos desafios são comumente disponibilizados na comunidade para reuso, onde podem ser baixados e incorporados em novos modelos que utilizam fotos como entrada.

Essa reutilização de modelos é efetiva pois os modelos usados como base são treinados com grandes *datasets* de imagens com uma grande variedade de classificações, o que faz com que os modelos aprendam a extrair *features* das imagens com eficiência.[Bro17]

2.1.3 CBIR

Avanços em armazenamento de dados e tecnologias de aquisição de imagens permitiram a criação de grandes bancos de dados de imagens. Nesse cenário, foi necessário criar sistemas de informação apropriados para gerenciar com eficiência essas coleções. A abordagem mais comum para isso é chamada de *Context Based Image Retrieval*(cbir) ou recuperação de imagens baseado em contexto [TF]. A abordagem CBIR funciona de modo antagônico a abordagem comum de busca de imagens baseada em semântica (SBIR), onde essa utiliza de metadados para fazer o reconhecimento de imagens, como por exemplo *tags*. O problema dessa abordagem é que ela pode consumir muito tempo de preparação dos dados, visto que a coleção de imagens deve passar por um processo de rotulação.

Uma solução típica utilizando CBIR requer a construção de um descritor de imagem, que é caracterizado por um algoritmo de extração de características e uma medida de similaridade para comparar as imagens. A medida de similaridade é uma função matemática que retorna o grau de similaridade dado um par de imagens representadas por seus vetores de característica.

A arquitetura de um sistema CBIR pode ser vista na figura 2.4.

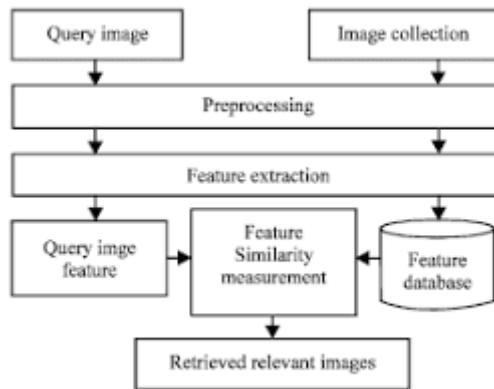


Figura 2.4 – Imagem representando uma arquitetura CBIR

2.2 Trabalhos similares

Nesta seção serão abordados algumas aplicações e trabalhos similares ao proposto.

2.2.1 Deep Learning of Binary Hash Codes for Fast Image Retrieval [LYHC15]

No estudo proposto por Kevin Lin, Huei-Fang Yang, Jen-Hao Hsiao e Chu-Song Chen é apresentado um método para a identificação de imagens baseada em conteúdo, mesmo problema a ser estudado neste projeto, porém sem um foco, como a identificação de filmes.

O funcionamento do método está representado na Figura 2.5. Primeiro é realizado um pré-treinamento das imagens em uma RNC no *ImageNet*. Após isso é adicionado uma camada na rede que faz uso das representações derivadas do primeiro passo a fim de se obter os *hashes* das imagens. E por fim é realizado um aprendizado para a busca de *hashes* como códigos binários, com o intuito de localizar imagens com códigos semelhantes. [LYHC15]

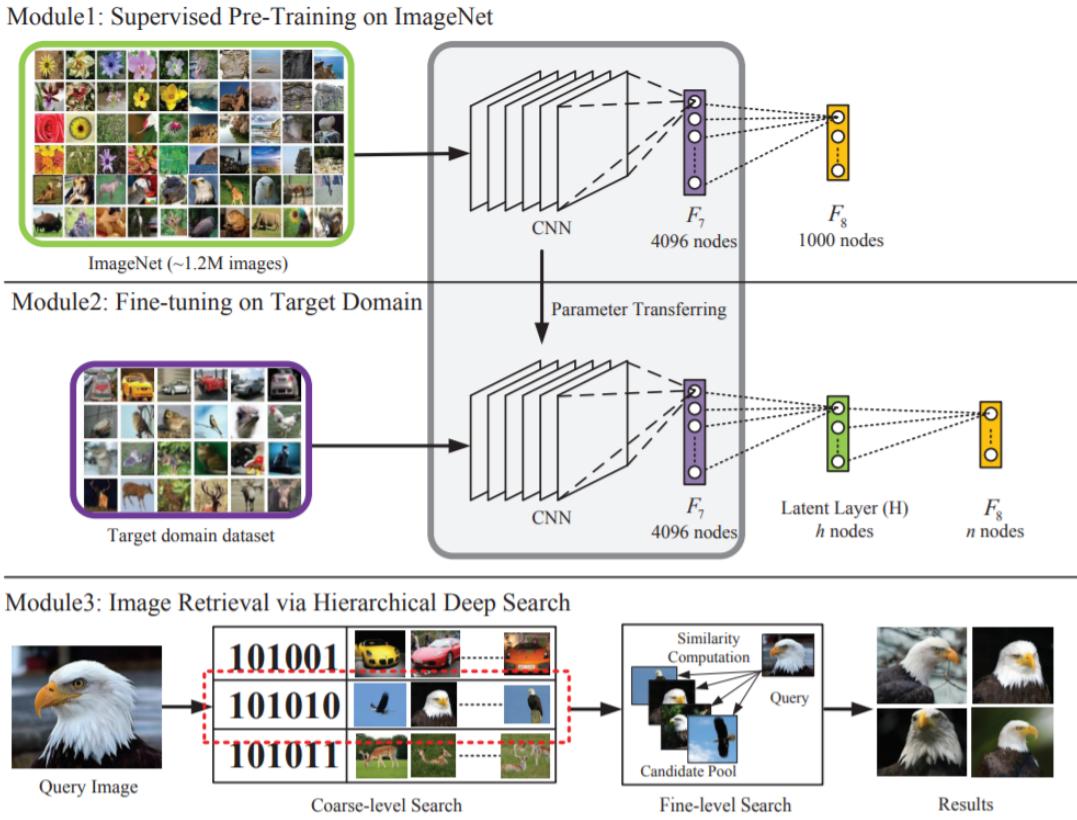


Figura 2.5 – Imagem demonstrando o método de identificação de imagens baseada em conteúdo
Fonte: *Deep Learning of Binary Hash Codes for Fast Image Retrieval* [LYHC15]

2.2.2 TheTake [The18]

TheTake é um aplicativo baseado no reconhecimento de contexto de imagem para a identificação de objetos e pessoas em um filme, problema semelhante ao que propomos resolver.

Baseando-se em inteligência artificial que entenda vídeos, o algoritmo da aplicação tem por objetivo detectar onde e quando as pessoas e os produtos aparecem em cada *frame* do vídeo. Possuindo uma base de dados com mais de 10000 produtos, a proposta do The Take é a identificação de objetos, como roupas, sapatos e acessórios usados pelos atores a fim de retornar onde comprar estes produtos e respectivos preços.

Primeiro o algoritmo identifica as pessoas e os objetos que elas estão usando naquele quadro, como pode ser visto na Figura 2.6. Em seguida, o algoritmo faz a identificação dos objetos na base de dados, ilustrado na Figura 2.7. Ao encontrá-lo, ele retorna o objeto exato e seus semelhantes como mostrado na Figura 2.8. Por último, ele oferece a opção de escolher a loja em que o objeto se encontra para o usuário realizar a compra, apresentado na Figura 2.9.



Figura 2.6 – Frame do GIF de demonstração da aplicação The Take, em que mostra o reconhecimento de objetos. Fonte: *Application based in artificial intelligence that understands video* [The18]

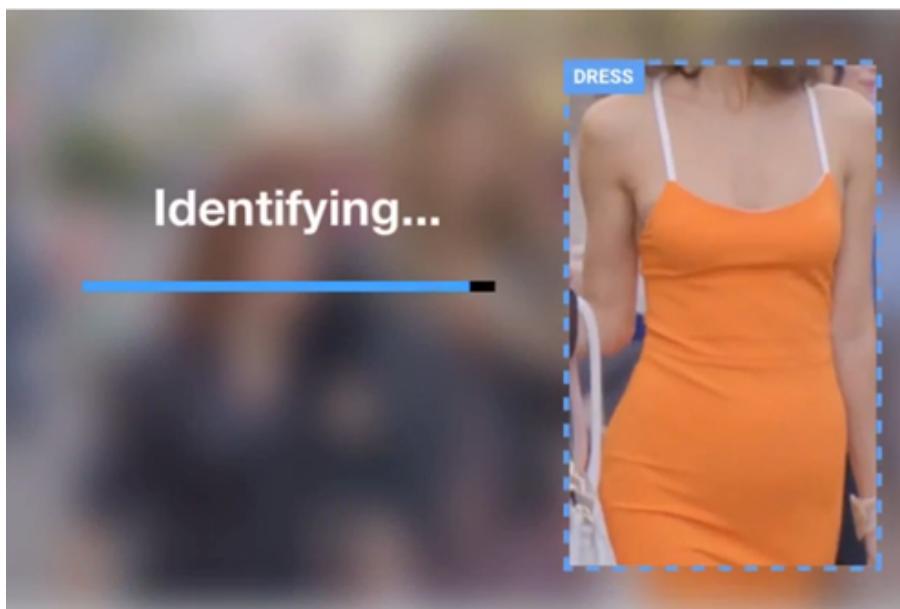


Figura 2.7 – Frame do GIF de demonstração da aplicação The Take, em que mostra a aplicação identificado os objetos. Fonte: *Application based in artificial intelligence that understands video* [The18]



Figura 2.8 – Frame do GIF de demonstração da aplicação The Take, em que mostra os objetos similares encontrados no banco de dados. Fonte: *Application based in artificial intelligence that understands video* [The18]

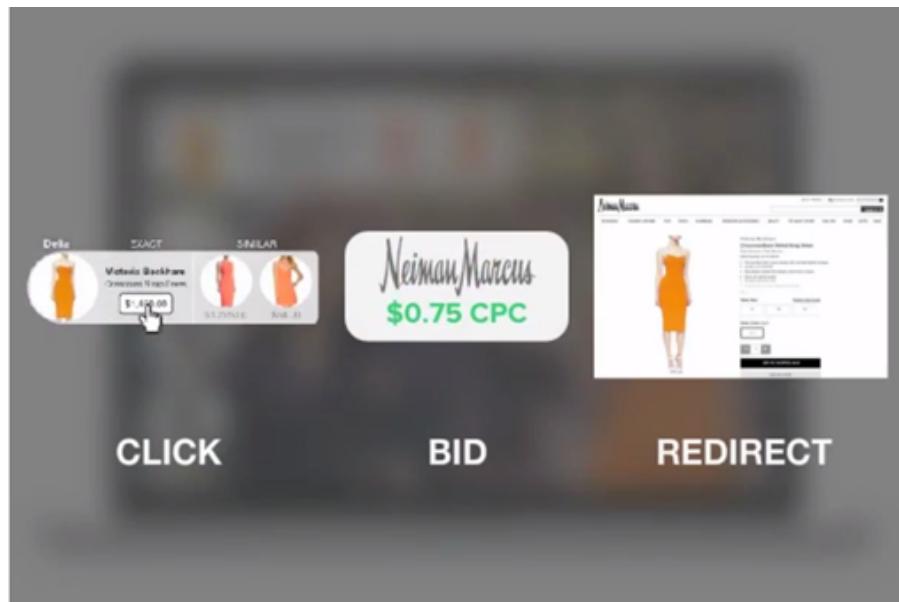


Figura 2.9 – Frame do GIF de demonstração da aplicação The Take, em que mostra como realizar a compra do objeto detectado. Fonte: *Application based in artificial intelligence that understands video* [The18]

2.2.3 Shazam [Sha18]

Shazam é um aplicativo multiplataforma que se baseia no reconhecimento da frequência de um som para a identificação da música ao qual o som pertence. Shazam

tem propósito similar ao desta proposta, porém com o uso de áudio. A aplicação tem como base o reconhecimento de um áudio por meio do microfone do dispositivo e a transformação deste áudio em um espectrograma (ver Figura 2.10) que fornece uma espécie de impressão digital do áudio. Utilizando inteligência artificial, a aplicação consegue comparar esta impressão digital às impressões digitais de suas mais de 11 milhões de músicas presentes em seu banco de dados.

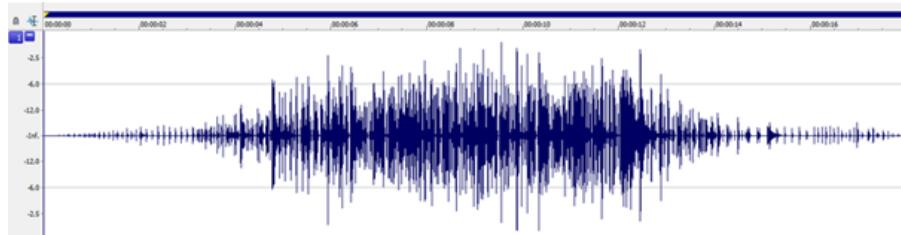


Figura 2.10 – Imagem de um espectrograma. Fonte: *Criatividade como Propriedade Emergente na Composição Algorítmica.*[CdSF13]

O funcionamento da aplicação se dá da seguinte forma: (1) o aplicativo utiliza o gravador do dispositivo para captar o som do ambiente por alguns segundos (Figura 2.11), (2) o transforma em um espectrograma e em uma impressão digital, enquanto isso (3) ele já faz a busca em sua base de dados até achar um valor com uma porcentagem de certeza alta o suficiente para saber a qual música o áudio se refere, (4) ao concluir a busca o aplicativo retorna o nome da música o álbum, e em quais aplicativos essa música está disponível para que o usuário possa escutar (Figura 2.12).

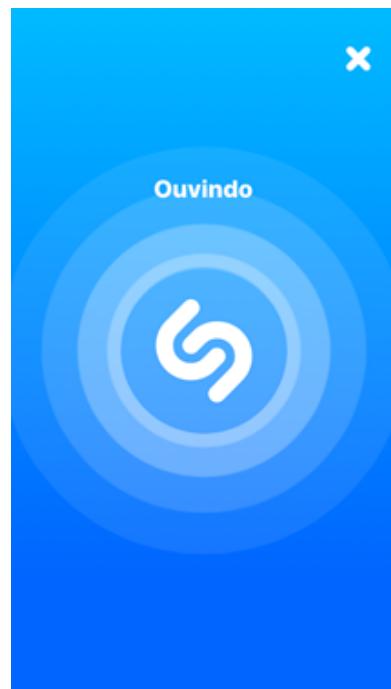


Figura 2.11 – Imagem do aplicativo Shazam para IOS. Aplicativo “ouvindo” o som ambiente.



Figura 2.12 – Imagem do aplicativo Shazam para IOS. Aplicativo mostrando a música encontrada.

2.3 Identificação da oportunidade

A partir do problema apresentado e das buscas por possíveis soluções existentes, foi percebido que o nicho do nosso problema ainda encontra-se em exploração, e que nossa proposta da criação de uma aplicação capaz de identificar o filme ao qual uma imagem pertence ainda não existe.

O aplicativo TheTake, é o que mais se aproxima da nossa solução. Porém ao ser testado, foi identificado que não está em funcionamento para a nossa região e que o seu contexto é o de identificação de objetos e pessoas sem necessariamente identificar o filme. Portanto identificamos que nossa proposta ainda não existe de forma eficaz e este ponto incentiva a criação da solução.

3. OBJETIVOS

Neste capítulo serão discutidos os objetivos gerais e específicos deste trabalho bem como os requisitos da solução.

3.1 Objetivo Geral

Com o atual destaque da área de *Deep Learning*, podemos dizer que nos próximos anos nossas vidas serão impactadas por diversas aplicações utilizando essa tecnologia e que nos próximos cinco a dez anos, ferramentas de desenvolvimento de *Deep Learning*, bibliotecas e linguagens de programação irão se tornar componentes padrão no desenvolvimento de qualquer software [Mit17].

Este trabalho tem como objetivo principal o aprimoramento dos conhecimentos adquiridos ao longo do curso de Sistemas de Informação e no desenvolvimento de aplicações móveis, assim como agregar conhecimentos em *Deep Learning*.

Entretanto o foco principal está em prover uma solução móvel para reconhecimento de filmes por meio de imagens com uma boa precisão que possa ser usado por qualquer pessoa portando um *smartphone*.

3.2 Objetivos específicos

Os objetivos específicos deste trabalho são:

- Proporcionar ao usuário uma aplicação móvel simples e com boa usabilidade capaz de identificar filmes baseando-se em uma foto tirada pelo celular.
- Testar um modelo de recuperação de imagens baseado em conteúdo utilizando uma rede neural como extrator de características.
- Prover um algoritmo de busca de imagem por similaridade escalável.
- Proporcionar ao administrador da ferramenta uma forma simples de fazer o carregamento de um novo filme para identificação.

4. PROJETO

Neste capítulo são apresentado as ferramentas, a arquitetura e o conjunto de dados da solução proposta neste trabalho.

4.1 Arquitetura da Solução

A arquitetura da solução é composta de diversos componentes que interagem entre si como pode ser visto na figura 4.1. Dentro da arquitetura pode-se dizer que existem duas grandes estruturas que compartilham componentes entre si. Essas estruturas podem ser identificadas como a parte de consulta do usuário e a parte de identificação de novos filmes.

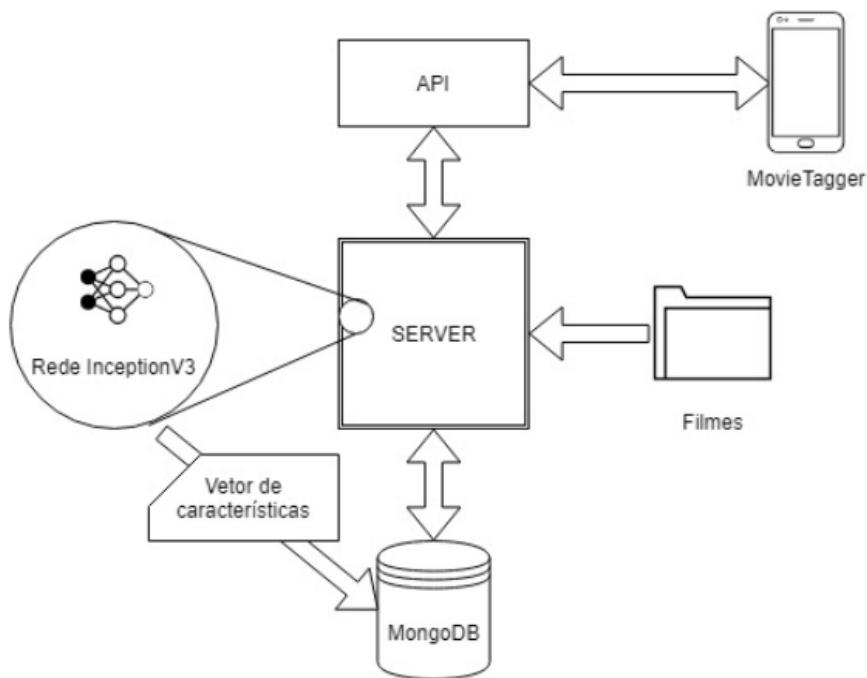


Figura 4.1 – Modelagem da arquitetura final da solução

4.1.1 Frames dos Trailers dos Filmes

Para realizar a extração dos *frames* dos trailers dos filmes, utilizou-se uma linguagem chamada Processing [RF14], utilizada para artes eletrônicas e projetos visuais, com bibliotecas já inclusas de manipulação de *frames* de vídeos. Sendo assim com algumas linhas de código foi possível definir a quantidade de *frames* por segundo que deveriam ser

extraídas de cada trailer. Optamos por cerca de 12 *frames* a cada segundo de vídeo, e manualmente realizamos a exclusão de *frames* totalmente pretos, com créditos no inicio e com créditos no fim do trailer, o que satisfez nossas expectativas para a criação do banco contendo estas imagens.

4.1.2 Modelo de Dados

MongoDB é um banco de dados NoSQL orientado a documentos que armazena seus dados de forma flexível em formato JSON. Por sua definição, o MongoDB lida com base de dados distribuídas [Mon18]. A ferramenta foi escolhida pois suporta tanto escalonamento vertical quanto horizontal utilizando *replica sets*(instâncias espelhadas) e *sharding*(dados distribuídos) [DUA17], além de ser de fácil integração pois a manipulação de JSONs é de baixa complexidade. Abaixo na figura 4.2 segue o modelo de dados utilizado pela aplicação.

| Index | Movies |
|---|---|
| <code>_id: ObjectId</code> <code>imageName: texto</code> <code>features: [float]</code> | <code>_id: ObjectId</code> <code>movield: texto</code> <code>movieName: texto</code> <code>poster: texto (base64)</code> <code>synopsis: texto</code> |

Figura 4.2 – Modelo de dados das coleções do banco de dados

4.1.3 Servidor

Para desenvolvimento do servidor, foi escolhida a linguagem Python por sua simplicidade e por sua abundância de bibliotecas e *frameworks* relacionados a área de aprendizado de máquina. Para fazer com que o ambiente possa ser embarcado em um servidor web, um *micro-framework* (*framework* minimalista) chamado Flask pode ser utilizado. O Flask foi desenvolvido em Python e publicado sob a licença BSD e se baseia na *template-engine* Jinja e na biblioteca WerkZeug [ROC14].

Como a aplicação não possui uma interface web o Jinja não será utilizado no projeto (vale ressaltar que caso futuramente seja desenvolvido uma interface web, o servidor já está pronto para realizar uma renderização de páginas devido a utilização desta *template-engine*).

Já o WerkZeug é uma biblioteca para desenvolvimento de apps WSGI (a especificação universal de como deve ser a interface entre um app Python e um web server). Ela possui a implementação básica deste padrão para interceptar requests e lidar com response, controle de cache, cookies, status HTTP, roteamento de urls e também conta com uma poderosa ferramenta de debug.

4.1.4 Cliente

Para o desenvolvimento do cliente optou-se por desenvolver uma aplicação móvel, permitindo uma maior facilidade em tirar fotos para o reconhecimento pela solução. Com isso, escolheu-se Ionic: um SDK de software livre para o desenvolvimento de aplicações híbridas em que um único código criado, poderá ser compilado para três plataformas diferentes de dispositivos móveis, são elas IOS, Android e Windows Phone [Sai17].

Ionic se utiliza de HTML, CSS e Angular para criar a lógica e o layout da aplicação, já, para acessarmos alguns recursos nativos do telefone como a câmera, precisamos recorrer a algumas bibliotecas e plugins do próprio SDK, que permitem a sua implementação utilizando Angular. Para este projeto utilizamos algumas destas bibliotecas, como a Camera: plugin usado para permitir o acesso à câmera nativa do sistema operacional do usuário da aplicação. Outro recurso externo utilizado foi a biblioteca HTTP do Ionic [Sai17], para permitir a comunicação da aplicação com a API do servidor, utilizando o método POST, em que se passa o base64 da imagem e o nível de similaridade mínimo de retorno da solução.

4.1.5 Conexão Cliente Servidor

Web Services fornecem um padrão de interoperação entre diferentes aplicações [Wor04]. Em um web service, protocolos de comunicação como o HTTP, originalmente concebidas para a comunicação homem-a- máquina, são utilizadas para a comunicação máquina-a-máquina, por meio de formatos de arquivo como XML e JSON. Atualmente, essas conexões são feitas por meio de Web API (*Application Programming Interface*). Aplicações que utilizam REST (*Representational State Transfer*, ou em português, Transferência de Estado Representacional) não exigem protocolos de web services baseados em XML o que torna seu desenvolvimento mais simples e mais difundido atualmente. Por este motivo foi es-

colhido o formato RESTful para o desenvolvimento desta aplicação e JSON para troca de informações.

4.1.6 Estrutura de Identificação

Para fazer o reconhecimento e recuperação da imagem o procedimento de recuperação de imagens baseado em conteúdo foi escolhido. A motivação da escolha se deve ao fato de que o conceito é amplamente usado na área de visão computacional. Esse procedimento consiste basicamente de duas principais funcionalidades: Extração de características da imagem e processamento das consultas.

O primeiro passo consiste em analisar uma imagem, e extrair características de interesse, no projeto, essa essa extração é feita pelo armazenamento de um vetor de características retirado de uma camada intermediária de uma rede neural convolucional.

O segundo passo é processar a consulta do usuário, através de comparações, isto é, comparar as características extraídas da imagem de consulta, com as imagens presentes na base, e através de alguma métrica, como a distância euclidiana, determinar as imagens mais próximas daquela usada como parâmetro, quanto menor a distância entre duas imagens, mais semelhantes elas serão [20116].

4.1.7 Extração de *Frames* de Validação

Para a validação da solução, após a extração de cerca de 12 *frames* por segundo de cada trailer, e da exclusão manual de alguns *frames* que poderiam prejudicar a solução, resolveu-se extrair 10% dos *frames* restantes de cada trailer dos filmes antes de sua submissão à rede e a base de dados final, para que, estes *frames* não estivessem presentes na base.

A extração dos 10% dos *frames* foi realizada com um script criado em NodeJS, que é uma plataforma de desenvolvimento server-side que interpreta JavaScript como linguagem, fácil e de rápido desenvolvimento, possuindo uma vasta gama de bibliotecas para o desenvolvimento de scripts e servidores, já que pode acessar as bibliotecas presentes no NPM (Node Package Manager ou traduzido para o português Gerenciador de Pacotes do Node) que é um repositório de código aberto de bibliotecas e pacotes. No script foi-se utilizado uma biblioteca chamada FileSystem que permite realizar comandos do sistema operacional de manipulação de arquivos, como excluir, copiar e criar arquivos e diretórios.

5. SOLUÇÃO

Este capítulo trata da solução desenvolvida a partir deste trabalho, descrevendo de forma funcional e ilustrativa as aplicações resultantes. Esta solução é um mínimo produto viável baseado no projeto, portanto alguns módulos do projeto não estão representados nesta solução.

5.1 Aplicação Móvel

Esta seção descreve as características da aplicação móvel, suas funcionalidades, definição de layout e telas.

A aplicação recebeu o nome de MovieTagger, a partir do nome, criaram-se o logotipo, ícones, imagens e cores necessárias para caracterizar a identidade visual da aplicação.

Para validar a solução a aplicação ficou dividida em duas telas, a primeira tela vista na Figura 5.1 possui dois status, o primeiro, é possível abrir a câmera do telefone o que se resume na Figura 5.2 para realizar a captura de uma foto de algum trailer dos filmes presentes no banco, após a captura a aplicação mostra a primeira tela novamente com o seu segundo status visto na Figura 5.3, em que o reconhecimento pode ser feito, há ainda um slider, que permite ao usuário só retornar um resultado se estiver dentro do range de similaridade (0% à 100%) selecionado.

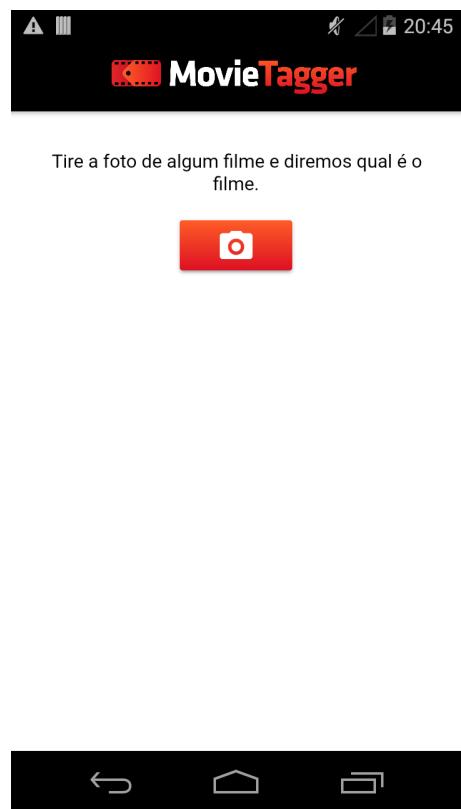


Figura 5.1 – Tela inicial da aplicação MovieTagger.



Figura 5.2 – Imagem da câmera do telefone fotografando uma cena do trailer do filme Star Wars Wars Episódio V: O Império Contra-Ataca



Figura 5.3 – Tela inicial da aplicação MovieTagger após uma foto ser capturada pela câmera do telefone, com a imagem do trailer do filme Star Wars Episódio V: O Império Contra-Ataca fotografada.

Após realizar a captura de uma imagem é possível pressionar o botão 'reconhecer', o que fará com que a aplicação faça uma chamada POST utilizando o protocolo HTTP para a API da solução enviando um JSON contendo uma propriedade chama "image" e seu valor, um base64 da imagem capturada pela câmera. Enquanto a requisição é realizada, um loading é mostrado na tela como na Figura 5.5. Por fim ao ser reconhecida a que filme a imagem pertence, a API retorna à aplicação um JSON contendo os dados necessário para visualmente mostrar na tela os resultados, o que se caracteriza na segunda tela da aplicação vista na Figura 5.5, em que o primeiro filme retornado com a similaridade mais alta é mostrado em destaque, seu nome seguido da porcentagem de similaridade, um pôster e abaixo uma pequena sinopse do filme.

No fim da segunda tela da aplicação ainda é possível ver os três resultados mais prováveis retornados da aplicação Figura 5.6, onde algumas informações importante além do nome e da porcentagem de similaridade aparecem, como o nome da imagem ao qual a solução descobriu a similaridade é mostrado. Um exemplo do JSON de retorno e da API da solução a aplicação móvel pode ser vista no apêndice A.

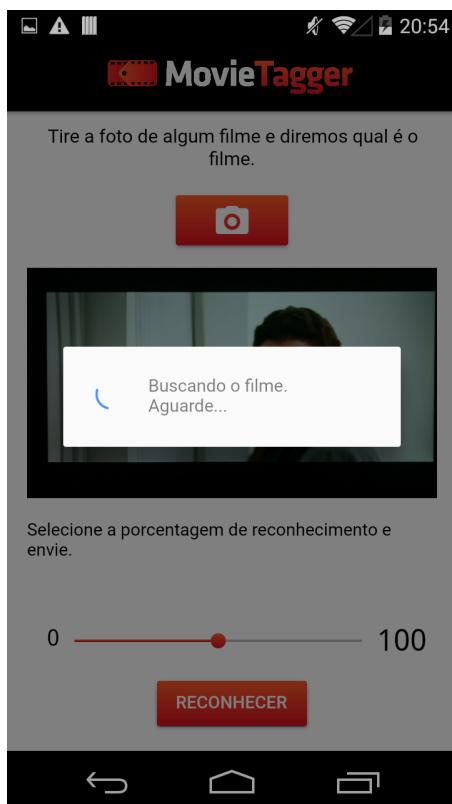


Figura 5.4 – Tela inicial da aplicação MovieTagger após a imagem capturada ser enviada para a API da solução, constando uma mensagem de carregamento até a resposta da API.



Figura 5.5 – Tela secundária da aplicação MovieTagger após o reconhecimento da imagem fotografada, em que detalhes do filme descoberto são mostradas.



Figura 5.6 – Tela secundária da aplicação MovieTagger após o reconhecimento da imagem fotografada, em que detalhes do filme descoberto são mostradas junto de detalhes de outros possíveis filmes descobertos.

5.2 Servidor

Esta seção descreve as características do servidor e ilustra como a aplicação soluciona o problema proposto. O principal objetivo do servidor é fazer a extração dos vetores de características e realizar a pesquisa de similaridade com os dados armazenados no banco de dados.

Para atingir os objetivos da proposta, foi implementado um servidor com uma API RESTful consumida pela aplicação móvel.

A extração do vetor de características foi implementada utilizando a rede neural convolucional InceptionV3, submetendo as imagens ao processamento da rede e armazenando a saída da camada *avg_pool* apontada na figura 5.7. O vetor possui o tamanho 1x2048 e é remodelado para 2048x1 apenas para melhorar a visualização durante o desenvolvimento.

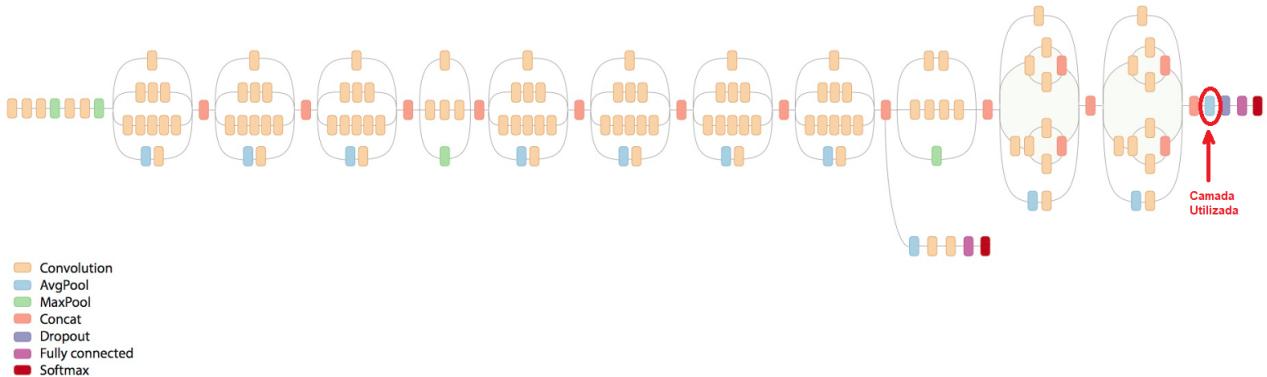


Figura 5.7 – Arquitetura da rede neural convolucional InceptionV3 apontando a camada utilizada para extrair o vetor de características.

Seguindo a estrutura de identificação de novos filmes, o vetor é armazenado juntamente com o nome do arquivo submetido (por convenção, o nome dos arquivos inicia com o nome do filme a qual a imagem pertence, para facilitar na hora da pesquisa das informações do filme) em formato JSON em uma coleção do banco de dados utilizando a API PyMongo.

Acoplada ao servidor foi desenvolvida uma API RESTful. Esta API possui um método POST que recebe um JSON, onde uma de suas propriedades se chama "*image*" e seu valor é o base64 de uma imagem submetida a comparação. Este base64 é convertido em um arquivo *.jpg* utilizando métodos do Keras e então é submetido ao modelo da rede InceptionV3 para ter seu vetor de características extraído.

Para fazer a comparação e elencar as imagens com maior similaridade no banco, é utilizado o cálculo de distância de cosseno descrita na fórmula 5.8 entre o vetor da imagem submetida e todos os vetores armazenados no banco. Acreditamos que esta comparação com todas as imagens do banco não seja a forma mais otimizada para fazer a comparação, porém dado o escopo de mínimo produto viável e o número de imagens armazenadas no banco, ela se mostrou viável.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum_{i=1}^n A_i B_i}{\sqrt{\sum_{i=1}^n A_i^2} \sqrt{\sum_{i=1}^n B_i^2}},$$

where \mathbf{A}_i and \mathbf{B}_i are components of vector \mathbf{A} and \mathbf{B} respectively.

Figura 5.8 – Fórmula de distância de cosseno utilizada para medir similaridade entre dois vetores.

A motivação para escolher a fórmula de distância de cosseno para comparação se deve ao fato de que ela despreza magnitudes. Sendo assim, imagens que representam um *frame* que forem submetidas com alteração em seu esquema de cores, brilho, luminosidade e angulação serão reconhecidas.

Após a comparação com todos os *frames* existentes no banco de dados, teremos uma lista de objetos JSON contendo o nome da imagem comparada, o percentual de similaridade e o nome do filme ordenados por similaridade de forma descendente. Assim, são extraídos os três primeiros objetos e as informações de seus respectivos filmes são adicionadas e então essa lista é retornada ao cliente. Um exemplo de estrutura retornada ao cliente pode ser vista no apêndice A.

5.3 Submissão de Novos Filmes

Esta seção descreve o processo de como a aplicação recebe novos filmes para armazenamento na base de dados.

Para a submissão de novos filmes foi desenvolvido um script Python desacoplado do servidor. O script é rodado por linha de comando passando como parâmetro o caminho de uma pasta no computador contendo os *frames* do novo filme em arquivos no formato JPG. Para a identificação do nome do filme, foi usado como convenção prefixar o nome do filme à qual a imagem pertence.

Assim que o comando é executado, o script processa um arquivo de cada vez. O arquivo é submetido a extração do vetor de características, e utilizando o PyMongo, é inserido um novo documento no banco de dados contendo o nome do filme (retirado do nome do arquivo) e o vetor de características.

Para inserir as informações do filme, como o nome, sinopse e um cartaz de cinema, atualmente o usuário administrador deve fazer uma inserção manual no banco. Futuramente será implementada uma interface melhor para fazer a administração dos dados dos filmes, porém por se tratar de um mínimo produto viável, e como foram usados apenas 12 trailers para validação da aplicação, esta parte foi mantida simples.

6. TESTES DA SOLUÇÃO E RESULTADOS

Este capítulo relata os testes e experimentos realizados com a solução, alguns destes realizados por submissão direta à API de imagens de validação, que foram retiradas aleatoriamente dos *frames* de cada trailer, portanto estas imagens e alguns pôsteres usados nos testes e na validação da solução não estão presentes no conjunto do banco de dados. Outros testes foram feitos pelo uso da aplicação móvel, utilizando fotografias de tela de televisores e outros monitores em que os trailers dos filmes estavam passando.

6.1 Submissão Direta de Imagens de Validação

Esta seção relata testes realizados na solução utilizando os dados de validação (10% dos *frames* que foram retirados aleatoriamente do conjunto geral de dados) e submetidos para comparação e validação.

Teste 1

Na Figura 6.1 a primeira imagem pertencente a uma cena do trailer de Star Wars Episódio II: Ataque dos Clones foi submetida diretamente à API da solução, com isso a API retornou as três imagens com maior índice de similaridade. O resultado foi positivo, conseguimos ver que as três imagens mais similares pertencem ao mesmo trailer do filme, Star Wars Episódio II: Ataque dos Clones, com uma similaridade de 75,98%, 75,92% e 75,39%.

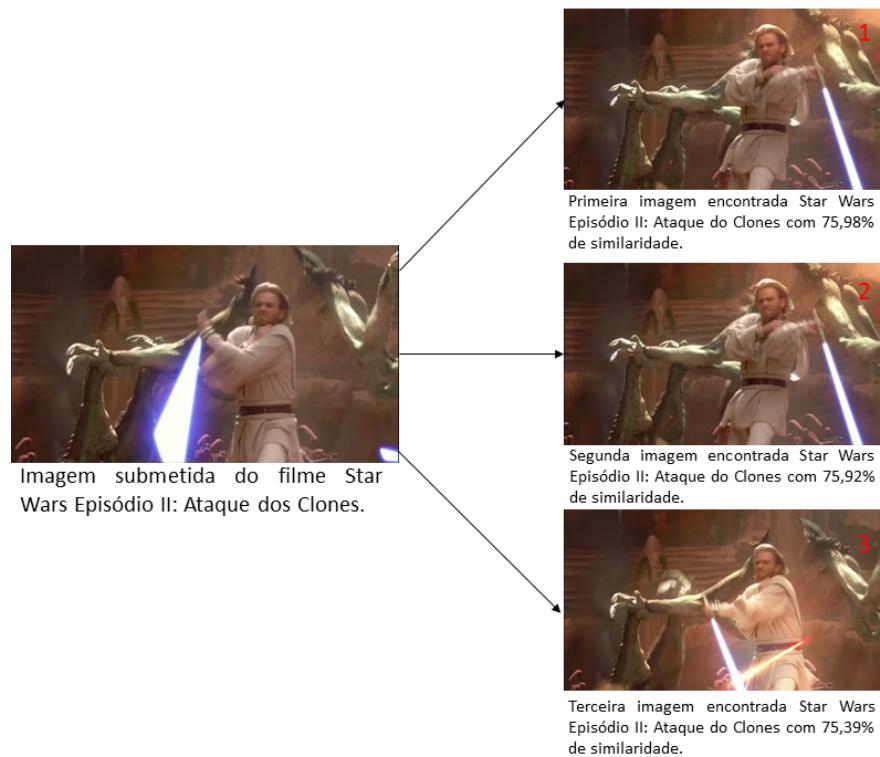


Figura 6.1 – Imagem contendo o *frame* do trailer do filme Star Wars Episódio II: Ataque dos Clones, submetido diretamente à API e suas três imagens mais similares.

Teste 2

Na Figura 6.2 a primeira imagem pertencente a uma cena do trailer de Star Wars Episódio VI: O Retorno de Jedi, esta imagem ao ser submetida diretamente à API da solução, retornou as três imagens mais similares à primeira. Percebe-se que apesar de as três imagens retornadas não serem do mesmo trailer do filme, notamos que as duas primeiras estão corretas quanto ao trailer original da primeira imagem, Star Wars Episódio VI: O Retorno de Jedi, com uma similaridade de 74,84% e 73,78% respectivamente. Consideraremos então um resultado positivo, já que a imagem retornada ao qual o trailer do filme pertencente não é o mesmo da imagem submetida veio em terceiro lugar, com uma porcentagem de 73,39% de similaridade, abaixo das outras duas.

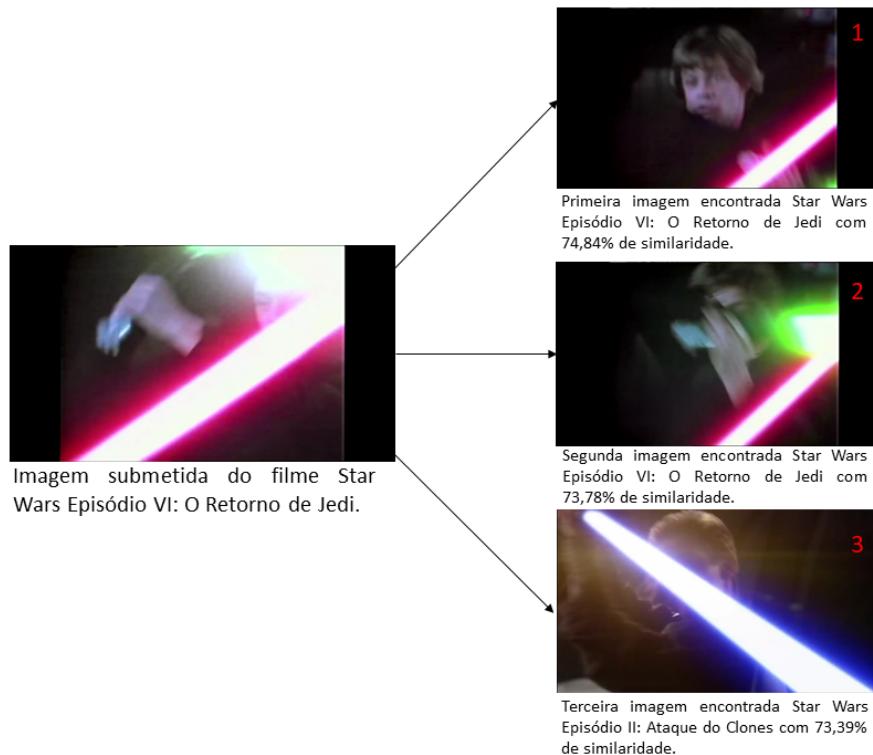


Figura 6.2 – Imagem contendo o *frame* do trailer do filme Star Wars Episódio VI: O Retorno de Jedi, submetido diretamente à API e suas três imagens mais similares.

Teste 3

Neste último teste percebe-se que na Figura 6.3 a primeira imagem pertence a uma cena do trailer de Star Trek V: A Última Fronteira, submetida à API da solução, obtendo-se três imagens bem similares a primeira, com porcentagens de similaridade de 92,96%, 92,52% e 92,46%. A porcentagem alta reflete na aparição desta cena durante um período mais longo de tempo no trailer. Conclui-se então que este teste foi positivo, já que, o índice de similaridade das três imagens em comparação com a submetida foram altos e todas elas são do mesmo trailer, Star Trek V: A última Fronteira.



Figura 6.3 – Imagem contendo o *frame* do trailer do filme Star Trek V: A última Fronteira, submetido diretamente na API e suas três imagens mais similares.

6.2 Submissão De Pôsteres

Esta seção relata alguns testes realizados na solução utilizando pôsteres dos filmes presentes na base de dados, sendo que, estes pôsteres não estão contidos na base de dados.

Teste 1

Na Figura 6.4 a primeira imagem pertence a um pôster do filme, Star Trek II: A Ira de Khan, o qual foi submetido a API da solução, obtendo como retorno três imagens presentes no trailer do mesmo filme, Star Trek II: A Ira de Khan. Percebe-se um resultado positivo deste teste, mesmo o pôster não estando presente na base de dados, conseguimos resultados de 65,97%, 65,64% e 65,36% de similaridade nas imagens, possíveis devido a elementos similares, como a tipografia neste caso, presente em cenas do trailer do filme.

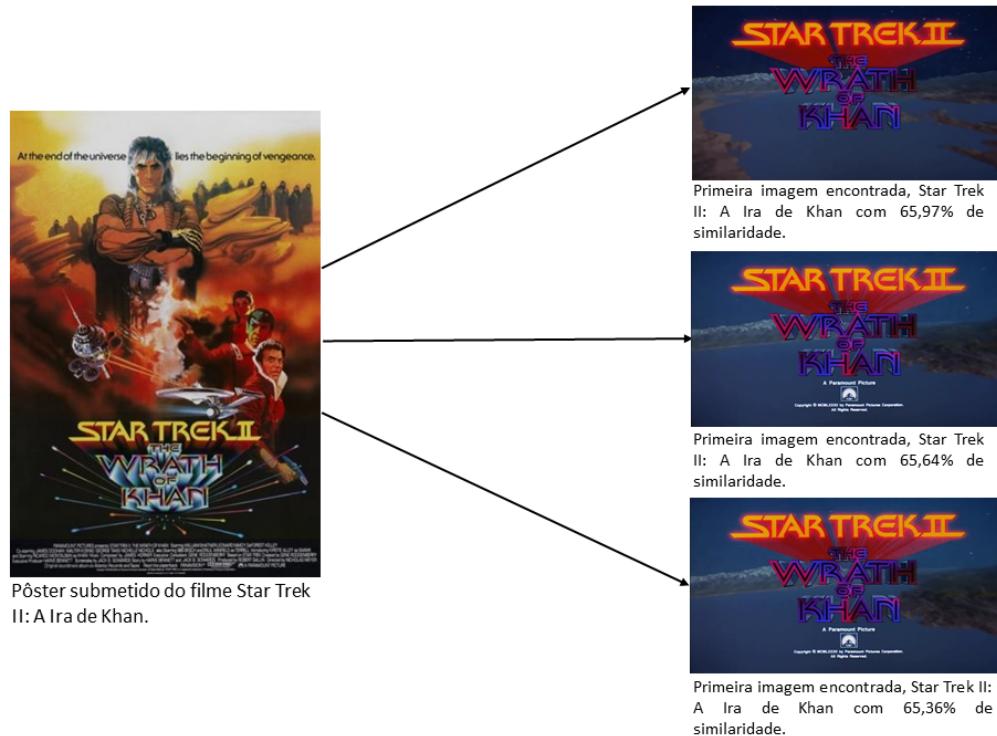


Figura 6.4 – Imagem contendo o pôster do filme Star Trek II: A Ira de Khan, submetido diretamente à API e suas três imagens mais similares.

Teste 2

Na Figura 6.5 a primeira imagem pertence a um pôster do filme, Star Wars V: O Império Contra-Ataca, submetido à API da solução, obtendo como retorno três imagens presentes no banco de dados, com elementos similares ao pôster do filme, porém não refletindo o filme correto, já que a primeira imagem - retornada com 70,03% de similaridade - se refere ao trailer do filme Star Wars Episódio IV: Uma Nova Esperança e as outras duas com 69,58% e 69,49% de similaridade se referem ao trailer do filme Star Wars Episódio I: A Ameaça Fantasma. Concluímos que este foi um teste de resultado negativo.

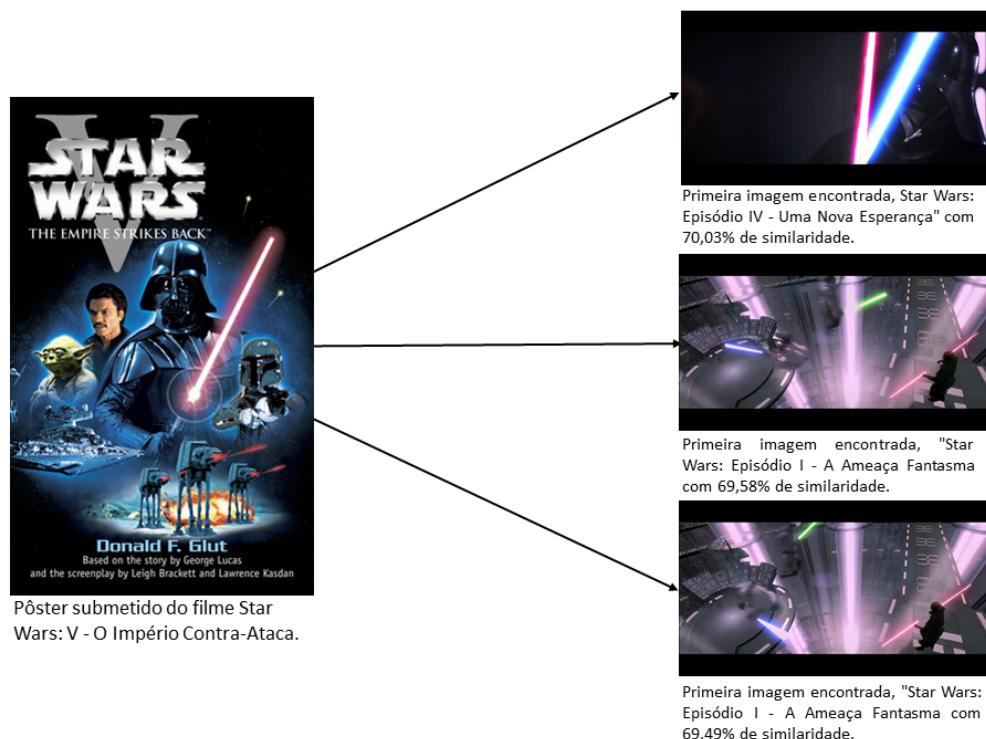


Figura 6.5 – Imagem contendo o pôster do filme Star Wars V: O Império Contra-Ataca, submetido diretamente na API e suas três imagens mais similares.

6.3 Submissão de Fotos via Aplicativo

Esta seção relata teste realizados via submissão de fotos à API da solução utilizando a aplicação móvel, o que se resume a fotos de televisores e monitores, não sendo totalmente fiéis as imagens presentes no banco de dados.

Teste 1

Na primeira captura de tela da Figura 6.6, observa-se a tela da aplicação móvel, após a captura de uma imagem do trailer do filme Star Wars Episódio III: A Vingança dos Sith. Ao ser pressionado o botão "reconhecer" a imagem capturada foi enviada para a API da solução, retornando as três imagens mais similares, neste caso, a solução identificou as três imagens sendo do mesmo filme, Star Wars Episódio III: A Vingança dos Sith, com similaridade de 85,45%, 85,35% e 84,85%. Sendo assim validamos como um teste positivo, já que o filmes corresponde ao da captura da imagem e os resultados de similaridade foram altos.

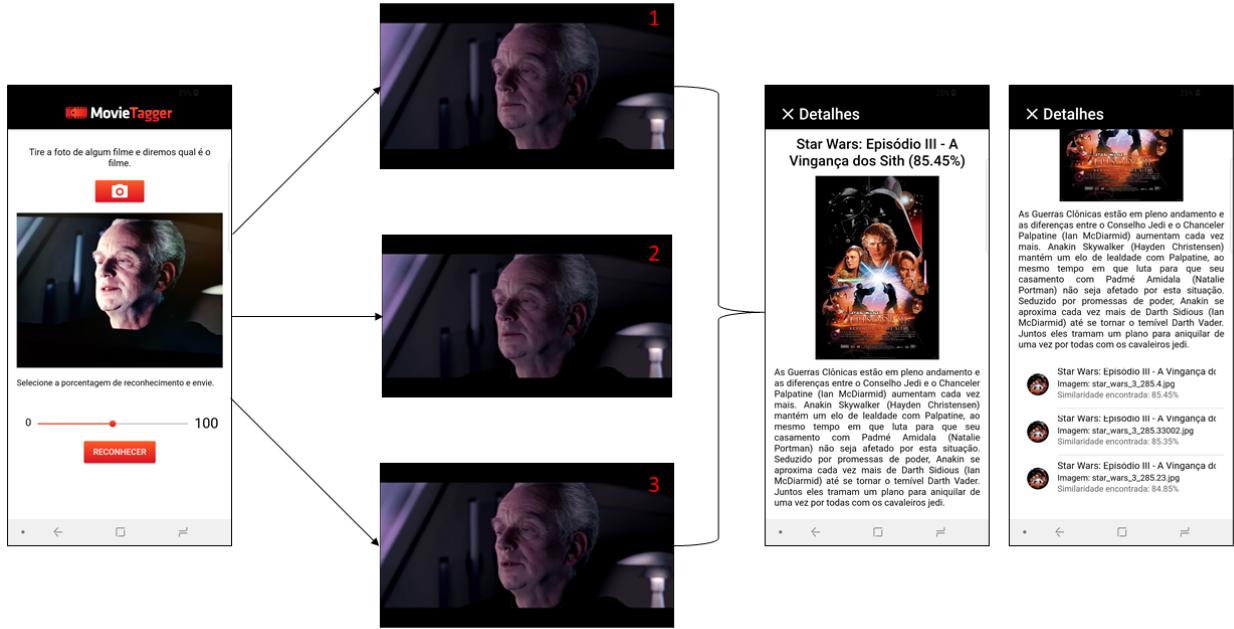


Figura 6.6 – Imagem contendo telas da aplicação móvel em que uma foto do trailer do filme Star Wars Episódio III: A Vingança dos Sith foi capturada

Teste 2

Neste segundo teste podemos ver que na Figura 6.7 a aplicação móvel foi utilizada para capturar uma imagem do trailer do filme Star Trek I: O Filme, e ao ser submetida na API da aplicação, esta foto retornou suas três mais similares, as três imagens são do mesmo filme da foto capturada, Star Trek I: O Filme, com similaridades de 76,94%, 76,93% e 75,36%. Conclui-se que este teste foi positivo, já que todas as imagens retornadas são bem similares à imagem capturada e pertencentes ao mesmo filme.

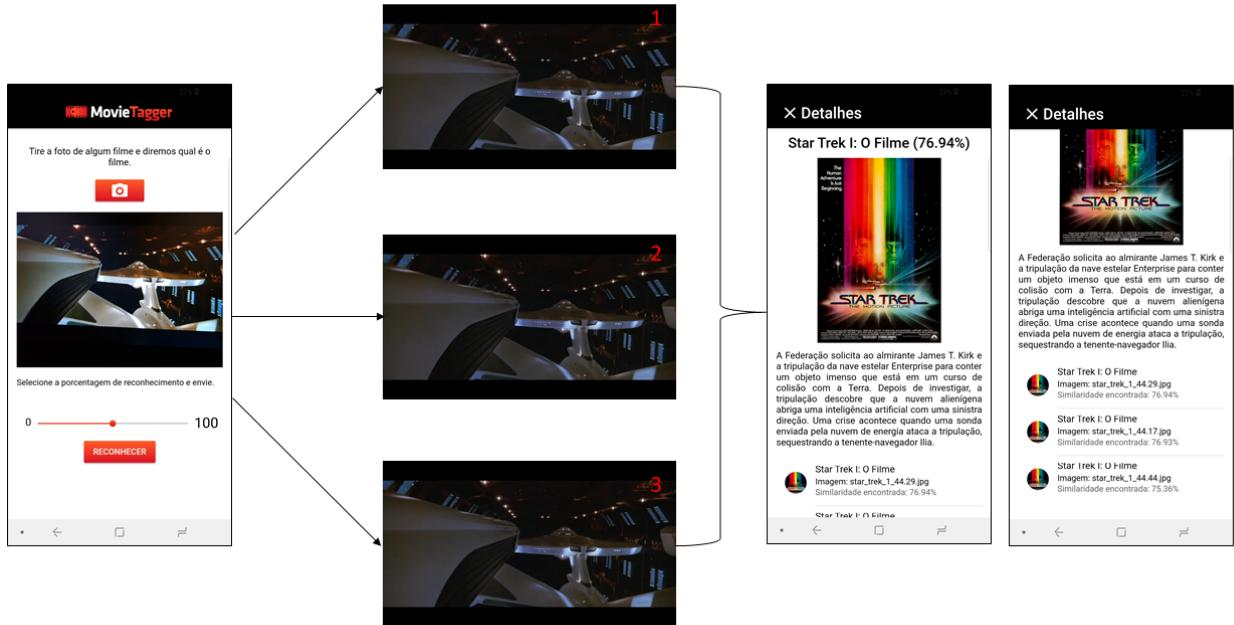


Figura 6.7 – Imagem contendo telas da aplicação móvel em que uma foto do trailer do filme Star Trek I: O Filme foi capturada

Teste 3

Neste teste podemos ver na Figura 6.8, que uma imagem do trailer do filme Star Wars Episódio IV: Uma Nova Esperança foi capturada pela aplicação móvel e submetida na API da solução, obtendo como retorno três imagens, porém de filmes diferentes, as duas primeiras com 71,24% e 68,91% de similaridade indicando que a imagem pertencia ao filme Star Wars Episódio VI: O Retorno de Jedi, e a última com 68,63% indicando pertencer ao filme, Star Trek IV: A Volta para Casa. As três imagens retornadas não eram do mesmo filme da imagem capturada, concluindo um caso de não sucesso do teste, porém percebemos que nas duas primeiras imagens retornadas o mesmo personagem capturado estava presente, porém em posições diferentes.

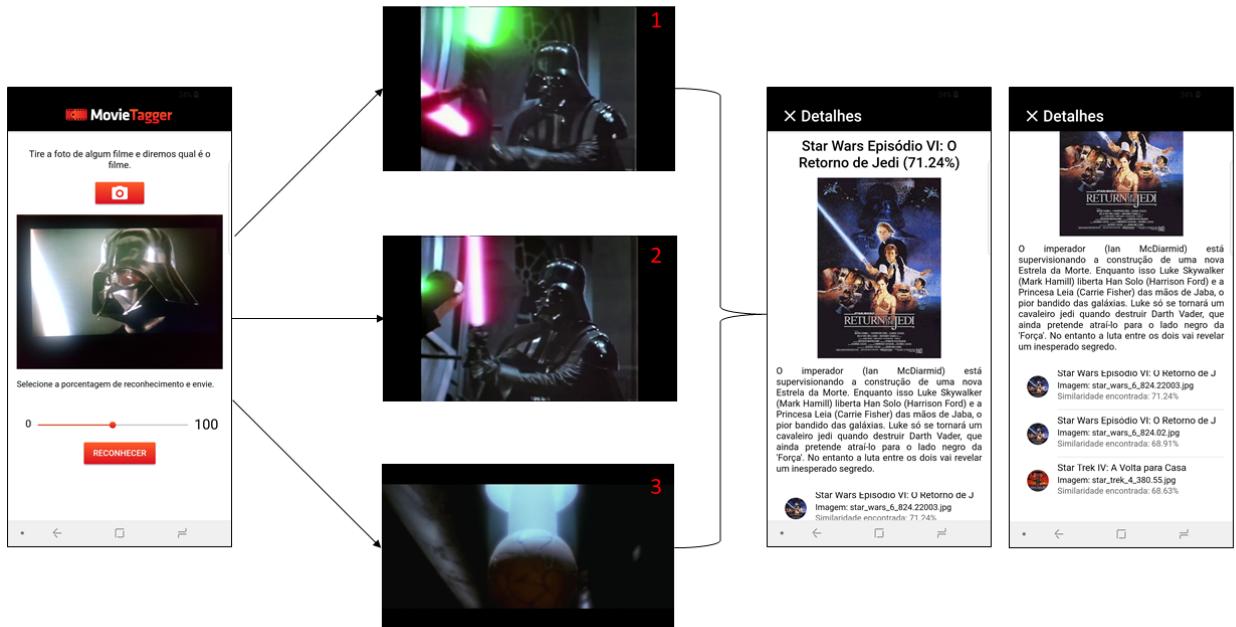


Figura 6.8 – Imagem contendo telas da aplicação móvel em que uma foto do trailer do filme Star Wars Episódio IV: Uma Nova Esperança foi capturada

7. CONSIDERAÇÕES FINAIS

Este trabalho traz como principal realização a absorção do conhecimento em métodos de *Deep Learning*, e de *Transfer Learning* para solucionar o problema proposto. Estes pontos foram atingidos a partir da criação de um servidor capaz de extrair as informações das imagens submetidas a ele e salvar em um banco de dados, para posteriormente ser consumido por uma API RESTful capaz de realizar a comparação de similaridade das imagens, e a criação de uma aplicação móvel capaz de validar a solução utilizando a captura de imagens pela câmera de um telefone e ao consumo da API utilizando o método POST do protocolo HTTP.

Destacamos a alta capacidade da aplicação em identificar os filmes, mesmo sendo desenvolvido apenas um MVP, a aplicação acertou o filme da imagem na maior parte dos testes, inclusive em situações onde a imagem submetida não existia no banco, como é o caso dos pôsters.

Entretanto podemos destacar pontos de melhoria, como questões de usabilidade e experiência do usuário, que poderiam trazer uma melhoria geral da solução, e o uso de ferramentas de estatísticas, para avaliar a aplicação conforme seu uso.

Outros pontos de melhoria que podemos destacar seriam um novo algoritmo para busca e comparação de frames mais eficiente, a utilização de outro método de seleção de frames, como a utilização de apenas um *frame* representativo por cena e uma utilização do banco de dados em uma infraestrutura distribuída.

Durante o uso do aplicativo MovieTagger e comentários sobre a solução, novas funcionalidades e ideias surgiram para o rumo da aplicação, como o uso para descobrir a que cena dos filmes uma foto qualquer poderia se parecer, então API da solução retornaria como resultado os *frames* mais similares, e não só o nome do filme, pois o interessante seria o usuário verem o que seria similar nos filmes com uma foto qualquer capturada.

Durante o processo de implementação e desenvolvimento, algumas dificuldades foram encontradas na utilização e instação do ambiente de desenvolvimento Python e do entendimento das bibliotecas de desenvolvimento utilizando redes neurais.

Em conclusão, a experiência de ter desenvolvido e trabalhado com métodos de *Deep Learning*, processos de desenvolvimento de visão computacional e desenvolvimento de aplicações utilizando Python e Ionic [Sai17] agregou muito valor para a formação acadêmica, profissional e pessoal.

REFERÊNCIAS BIBLIOGRÁFICAS

- [20116] “Aplicação de técnicas de content-based image retrieval (cbir) em imagens radiográficas”, Dissertação de Mestrado, 2016, instituto de Informática - INF (RG).
- [ALE14] ALEXANDER, M. M. M., “A survey on image retrieval methods”, March 2014, Capturado em: <http://cogprints.org/9815/>.
- [Bro17] Brownlee, J. “A gentle introduction to transfer learning for deep learning”. Capturado em: <https://machinelearningmastery.com/transfer-learning-for-deep-learning/>, Mar 2018.
- [CdSF13] Coelho de Souza, R.; Faria, R. “A criatividade como propriedade emergente na composição algorítmica.”, vol. 10, 06 2013, pp. 21–34.
- [DUA17] DUARTE J, L. F. “Mongodb para iniciantes em nosql”, September 2017.
- [EGE⁺99] Eakins, J.; Graham, M.; Eakins, J.; Graham, M.; Franklin, T. “Content-based image retrieval”, *Library and Information Briefings*, vol. 85, 1999, pp. 1–15.
- [FdS16] Furtado, A. M. M.; de Souza, D. M. “Descoberta de Elenco de Trailers de Filmes Utilizando Reconhecimento Facial em Deep Learning”. EDIPUCRS, 2016, <http://revistaseletronicas.pucrs.br/ojs/index.php/graduacao/article/view/25672/14967>.
- [FlI16] FlipaClip. “Frames per second "fps"explained”. Capturado em: <https://support.flipaclip.us/knowledge-bases/3/articles/306-frames-per-second-fps-explained>, Feb 2018.
- [GBC16] Goodfellow, I.; Bengio, Y.; Courville, A. “Deep Learning”. MIT Press, 2016, <http://www.deeplearningbook.org>.
- [Gom18] Gomez, A. “Deep learning in digital pathology”. Capturado em: <http://www.global-engage.com/life-science/deep-learning-in-digital-pathology/>, Mar 2018.
- [LYHC15] Lin, K.; Yang, H.-F.; Hsiao, J.-H.; Chen, C.-S. “Deep learning of binary hash codes for fast image retrieval.” In: CVPR Workshops, 2015, pp. 27–35.
- [Mat17] MathWorks. “Convolutional neural network”. Capturado em: <https://www.mathworks.com/discovery/convolutional-neural-network.html>, Mar 2018.

- [Mit17] Mittal, V. “Top 15 deep learning applications that will rule the world in 2018 and beyond”. [Online; posted Oct 3, 2017], Capturado em: <https://medium.com/@vratulmittal/top-15-deep-learning-applications-that-will-rule-the-world-in-2018-and-beyond-7c6130c43b> October 2017.
- [Mon18] MongoDB. “What is mongodb?”, November 2018.
- [RF14] Reas, C.; Fry, B. “Processing: A Programming Handbook for Visual Designers and Artists”. The MIT Press, 2014.
- [ROC14] ROCHA, B. “What the flask? pt-1 introdução ao desenvolvimento web com python”, MY 2014.
- [Sai17] Saini, G. “Hybrid Mobile Development with Ionic”. Packt Publishing, 2017.
- [Sha18] Shazam. “Application that recognize musics”. Capturado em: <https://www.shazam.com>, April 2018.
- [SVI⁺15] Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. “Rethinking the inception architecture for computer vision”, *CoRR*, vol. abs/1512.00567, 2015, 1512.00567.
- [TF] Torres, R. D. S.; Falcão, A. X. “Content-based image retrieval: Theory and applications”, *Revista de Informática Teórica e Aplicada*, vol. 13, pp. 161–185.
- [The18] TheTake. “Application based in artificial intelligence that understands video”. Capturado em: <http://http://thetake.ai/>, April 2018.
- [Wor04] World Wide Web Consortium. “Web services architecture”, 2004, <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>.

APÊNDICE A – ESTRUTURA JSON RETORNADA PELO SERVIDOR

Este apêndice mostra um exemplo de JSON retornado pelo servidor. Para facilitar a leitura do arquivo, as propriedades que retornavam um texto em formato Base64 foram substituídas pelo texto "base64".

```

1 [{  

2     "imageName": "star_trek_2_270.66998.jpg",  

3     "movieInfo": {  

4         "nomeCompletoFilme": "Star Trek II: A Ira de Khan",  

5         "poster": "base64",  

6         "sinopse": "No s\u00e3culo XXIII, o almirante James T. Kirk (William Shatner) enfrenta uma crise de meia-idade enquanto se ocupa de treinar cadetes em simuladores. Paralelamente uma respeitada cientista, Carol Marcus (Bibi Besch), ex-namorada de Kirk, est\u00e1 envolvida no Projeto G\u00e3eanese, que pode dar vida a planetas sem nenhuma forma de vida e tamb\u00e9m ser uma arma terr\u00f5edvel em m\u00e1s erradas, pois se usada onde j\u00e1 havia vida todas as formas de vida ser\u00e3o extintas, dando lugar a uma 'nova matriz1'. O capit\u00e3o da Reliant, Terrell (Paul Winfield), e Chekov (Walter Koenig) sondam um planeta pensando que \u00e3o Ceti Alfa 6, para ver se este \u00e3o ideal para o projeto. Chekov percebe que algo est\u00e1 errado e avisa Terrell, mas logo os dois s\u00e3o feitos prisioneiros de Khan (Ricardo Montalban), que \u00e3o inimigo de Kirk h\u00e1 200 anos. Eles est\u00e3o mesmo \u00e3o em Ceti Alfa 5 e Khan mostra aos seus prisioneiros a \u00fanica forma de vida que restou no planeta. Ao pegar dois filhotes, avisa para Terrell e Chekov que colocar\u00e1 cada um deles nos seus ouvidos, pois os filhotes v\u00e3o se alojar no c\u00f3rtice cerebral e, como resultado, a v\u00edima fica extremamente suscet\u00e1vel a sugest\u00e3o e, posteriormente, a loucura e a morte. Enquanto isso, na Enterprise, Kirk parte em uma viagem de treinamento com os novos cadetes, mas a Dra. Marcus est\u00e1 sob tens\u00e3o, pois do Reliant Chekov entrou em contato com ela dizendo que retornar\u00e1 em tr\u00eas dias (e n\u00e3o em tr\u00eas meses). Al\u00e3o disto Chekov diz que deixou Ceti Alfa 6 e que quando chegar em Regula Um, onde est\u00e1 Carol, todo o material do G\u00e3eanese dever\u00e1 ser dado para a Reliant, para ser testado em Ceti Alfa 6. Ela considera isto tudo irregular, mas Chekov afirma que recebeu novas ordens do comandante Kirk. Carol tenta falar com Kirk, mas a transmiss\u00e3o sofre interfer\u00eancia e ela n\u00e3o tem como negar a veracidade da ordem. Entretanto, Kirk sente que algo errado est\u00e1 acontecendo e, como \u00e3o a \u00fanica nave do quadrante, ter\u00e1 de apurar o que est\u00e1 havendo, mesmo que parte da sua tripula\u00e7\u00e3o seja de novatos. Assim se v\u00e3o obrigado a reassumir a Enterprise, sendo que Khan por sua vez far\u00e1 tudo para conseguir o G\u00e3eanese, pois nas suas m\u00e1s se tornar\u00e1 algo terr\u00f5edvel para qualquer um que o desafie."  

7     },  

8     "movieName": "star_trek_2",  

9     "similarity": 0.659719588277513  

10 }, {  

11     "imageName": "star_trek_2_271.91998.jpg",  

12     "movieInfo": {  

13         "nomeCompletoFilme": "Star Trek II: A Ira de Khan",  

14         "poster": "base64",  

15         "sinopse": "No s\u00e3culo XXIII, o almirante James T. Kirk (William Shatner) enfrenta uma crise de meia-idade enquanto se ocupa de treinar cadetes em simuladores. Paralelamente uma respeitada cientista, Carol Marcus (Bibi Besch), ex-namorada de Kirk, est\u00e1 envolvida no Projeto G\u00e3eanese, que pode dar vida a planetas sem nenhuma forma de vida e tamb\u00e9m ser uma arma terr\u00f5edvel em m\u00e1s erradas, pois se usada onde j\u00e1 havia vida todas as formas de vida ser\u00e3o extintas, dando lugar a uma 'nova matriz1'. O capit\u00e3o da Reliant, Terrell (Paul Winfield), e Chekov (Walter Koenig) sondam um planeta pensando que \u00e3o Ceti Alfa 6, para ver se este \u00e3o ideal para o projeto. Chekov percebe que algo est\u00e1 errado e avisa Terrell, mas logo os dois s\u00e3o feitos prisioneiros de Khan (Ricardo Montalban), que \u00e3o inimigo de Kirk h\u00e1 200 anos. Eles est\u00e3o mesmo \u00e3o em Ceti Alfa 5 e Khan mostra aos seus prisioneiros a \u00fanica forma de vida que restou no planeta. Ao pegar dois filhotes, avisa para Terrell e Chekov que colocar\u00e1 cada um deles nos seus ouvidos, pois os filhotes v\u00e3o se alojar no c\u00f3rtice cerebral e, como resultado, a v\u00edima fica extremamente suscet\u00e1vel a sugest\u00e3o e, posteriormente, a loucura e a morte. Enquanto isso, na Enterprise, Kirk parte em uma viagem de treinamento com os novos cadetes, mas a Dra. Marcus

```

est\u00e1 sob tens\u00e3o, pois do Reliant Chekov entrou em contato com ela dizendo que retornar\u00e1 em tr\u00eas dias (e n\u00e3o em tr\u00eas meses). Al\u00e9m disto Chekov diz que deixou Ceti Alfa 6 e que quando chegar em Regula Um, onde est\u00e1 Carol, todo o material do G\u00e3eanesis dever\u00e1 ser dado para a Reliant, para ser testado em Ceti Alfa 6. Ela considera isto tudo irregular, mas Chekov afirma que recebeu novas ordens do comandante Kirk. Carol tenta falar com Kirk, mas a transmiss\u00e3o sofre interfer\u00eancia e ela n\u00e3o tem como negar a veracidade da ordem. Entretanto, Kirk sente que algo errado est\u00e1 acontecendo e, como \u00e9 a \u00fanica nave do quadrante, ter\u00e1 de apurar o que est\u00e1 havendo, mesmo que parte da sua tripula\u00e7\u00e3o seja de novatos. Assim se v\u00e1 obrigado a reassumir a Enterprise, sendo que Khan por sua vez far\u00e1 tudo para conseguir o G\u00e3eanese, pois nas suas m\u00e3os se tornar\u00e1 algo terr\u00edvel para qualquer um que o desafie."

```

16     },
17     "movieName": "star_trek_2",
18     "similarity": 0.656441178358514
19 }, {
20     "imageName": "star_trek_2_272.01.jpg",
21     "movieInfo": {
22         "nomeCompletoFilme": "Star Trek II: A Ira de Khan",
23         "poster": "base64",
24         "sinopse": "No s\u00e9culo XXIII, o almirante James T. Kirk (William Shatner) enfrenta uma crise de meia-idade enquanto se ocupa de treinar cadetes em simuladores. Paralelamente uma respeitada cientista, Carol Marcus (Bibi Besch), ex-namorada de Kirk, est\u00e1 envolvida no Projeto G\u00e3eanese, que pode dar vida a planetas sem nenhuma forma de vida e tamb\u00e9m ser uma arma terr\u00edvel em m\u00e3os erradas, pois se usada onde j\u00f3 havia vida todas as formas de vida ser\u00e1o extintas, dando lugar a uma 'nova matriz'. O capit\u00e3o da Reliant, Terrell (Paul Winfield), e Chekov (Walter Koenig) sondam um planeta pensando que \u00e9 Ceti Alfa 6, para ver se este \u00e9 ideal para o projeto. Chekov percebe que algo est\u00e1 errado e avisa Terrell, mas logo os dois s\u00e3o feitos prisioneiros de Khan (Ricardo Montalban), que \u00e9 inimigo de Kirk h\u00e1 200 anos. Eles est\u00e1o mesmo \u00e9 em Ceti Alfa 5 e Khan mostra aos seus prisioneiros a \u00fanica forma de vida que restou no planeta. Ao pegar dois filhotes, avisa para Terrell e Chekov que colocar\u00e1 cada um deles nos seus ouvidos, pois os filhotes v\u00e3o se alojar no c\u00f3rtice cerebral e, como resultado, a v\u00e3edtima fica extremamente suscet\u00e1vel a sugest\u00e3o e, posteriormente, a loucura e a morte. Enquanto isso, na Enterprise, Kirk parte em uma viagem de treinamento com os novos cadetes, mas a Dra. Marcus est\u00e1 sob tens\u00e3o, pois do Reliant Chekov entrou em contato com ela dizendo que retornar\u00e1 em tr\u00eas dias (e n\u00e3o em tr\u00eas meses). Al\u00e9m disto Chekov diz que deixou Ceti Alfa 6 e que quando chegar em Regula Um, onde est\u00e1 Carol, todo o material do G\u00e3eanesis dever\u00e1 ser dado para a Reliant, para ser testado em Ceti Alfa 6. Ela considera isto tudo irregular, mas Chekov afirma que recebeu novas ordens do comandante Kirk. Carol tenta falar com Kirk, mas a transmiss\u00e3o sofre interfer\u00eancia e ela n\u00e3o tem como negar a veracidade da ordem. Entretanto, Kirk sente que algo errado est\u00e1 acontecendo e, como \u00e9 a \u00fanica nave do quadrante, ter\u00e1 de apurar o que est\u00e1 havendo, mesmo que parte da sua tripula\u00e7\u00e3o seja de novatos. Assim se v\u00e1 obrigado a reassumir a Enterprise, sendo que Khan por sua vez far\u00e1 tudo para conseguir o G\u00e3eanese, pois nas suas m\u00e3os se tornar\u00e1 algo terr\u00edvel para qualquer um que o desafie."
25     },
26     "movieName": "star_trek_2",
27     "similarity": 0.6536371934627712
28 }]

```