UNIVERSIDADE FEDERAL DE OURO PRETO

DEPARTAMENTO DE COMPUTAÇÃO

Luís Filipe Lima Alves Vieira

CORRESPONDÊNCIA DE IDENTIDADE DE USUÁRIOS EM PLATAFORMAS DE VÍDEO

Ouro Preto, MG
2019

Luís Filipe Lima Alves Vieira

UNIVERSIDADE FEDERAL DE OURO PRETO

DEPARTAMENTO DE COMPUTAÇÃO

Monografia II apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciência da Computação.

Orientador: M.Sc. Reinaldo Silva Fortes

Ouro Preto, MG
2019

V658c Vieira, Luís Filipe Lima Alves.
Correspondência de identidade de usuários em plataformas de vídeo [manuscrito] / Luís Filipe Lima Alves Vieira. - 2019.

37f.:

Orientador: Prof. MSc. Reinaldo Silva Fortes.

Monografia (Graduação). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Biológicas. Departamento de Computação.

1. Identidade. 2. Vídeos para Internet. 3. Mídias sociais. I. Fortes, Reinaldo Silva. II. Universidade Federal de Ouro Preto. III. Titulo.

CDU: 004.77

Catalogação: ficha.sisbin@ufop.edu.br

Luís Filipe Lima Alves Vieira

CORRESPONDÊNCIA DE IDENTIDADE DE USUÁRIOS EM PLATAFORMAS DE VÍDEO

Monografia II apresentada ao Curso de Ciência da Computação da Universidade Federal de Ouro Preto como parte dos requisitos necessários para a obtenção do grau em Bacharel em Ciência da Computação.

Aprovada em Ouro Preto, 17 de Julho de 2019.

M.Sc. Reinaldo Silva Fortes
Universidade Federal de Ouro Preto
Orientador

Dra. Amanda Sávio Nascimento e Silva
Universidade Federal de Ouro Preto - UFOP
Examinador

Dr. Guilherme Tavares de Assis
Universidade Federal de Ouro Preto - UFOP
Examinador

Dedico esse trabalho para os meus pais Roberto e Olinta.

Agradecimentos

Agradeço aos meus pais Roberto e Olinta e meus irmãos Vinicius e Danilo por todo o incentivo durante os anos de faculdade.

Agradeço a todos os docentes e servidores da Universidade Federal de Ouro Preto, em especial ao Departamento de Computação.

Agradeço a meu orientador Reinaldo S. Fortes por todo apoio e paciência durante a elaboração da minha monografia.

Agradeço a todos os meus amigos por todo apoio, companheirismo, conselhos, e troca de aprendizados.

Agradeço aos moradores, ex-alunos e homenageados da República Távola Redonda, por criarem e manterem a melhor república de Ouro Preto.

Você pode saber o nome desse pássaro em todas as línguas do mundo, mas quando tiver terminado, você não saberá absolutamente nada sobre o pássaro. Você só saberá sobre humanos em lugares diferentes, e o que eles chamam de pássaro. (FEYNMAN, 1992)

Resumo

Plataformas de vídeo e redes sociais são sites denominados mídias sociais e, atualmente, compõem os sites mais acessados pela população. Considerando o tópico de correspondência de identidade, constatou-se que os trabalhos presentes na literatura são construídos, majoritariamente, sobre bases de dados de redes sociais. Este trabalho tem como objetivo a aplicação de técnicas de correspondência de identidade em dados de plataformas de vídeo, uma vez que os modelos de dados presentes em sites de mídias sociais compartilham um conjunto de atributos similares. Para isso, o trabalho propõe uma técnica de correspondência de identidade voltada para as plataformas de vídeo com métodos próprios e alguns dos métodos utilizados pela literatura em bases de redes sociais. O trabalho realiza a coleta nas bases de dados das plataformas de vídeo e, em seguida, aplica os métodos da técnica de correspondência de identidade. Tanto a base coletada como a base gerada pela aplicação da correspondência de identidade são maiores em quantidade de registros em comparação com os trabalhos presentes na literatura.

Palavras-chave: Correspondência de Identidade. Plataformas de Vídeo Online. Mídias Sociais.

Abstract

Video platforms and social networks are sites denominated social media and, nowadays, are the ones most accessed by the population. Considering the topic of identity matching, it was found that the works presented in the literature are mostly constructed on datasets of social networks. This work aims to apply identity matching techniques to video platform data, since the data models presented in social media sites share a set of similar attributes. For this, the work proposes an identity matching technique for the video platforms with our own methods and some of the methods used by the literature on social networks. The work performs a data retrieval of the video platforms' datasets and then applies the identity matching techniques. Both the base collected and the base generated by the application of the identity matching are larger in quantity of records compared to the works presented in the literature.

Keywords: identity matching, online video platforms, social media.

Lista de Ilustrações

Lista de Tabelas

Lista de Algoritmos

Sumário

1 Introdução

As principais Plataformas de Vídeo Online (*Online Video Platforms*, OVP), como YouTube, Dailymotion, Vimeo[1], surgiram por volta de 2005, visto que isso só foi possível devido à integração de vídeos aos navegadores, popularização das câmeras digitas e da internet banda larga (ALLOCCA, 2018). As plataformas proporcionaram às pessoas o compartilhamento e a reprodução de conteúdos em áudio e vídeo sem a necessidade de realizar o *download*. Também contam com espaços para a criação de comunidades e interação entre pessoas. Essas funcionalidades rapidamente atraíram a atenção do público. Segundo o (IWS, 2019), o número de pessoas conectadas na internet é de 4,3 bilhões o YouTube a maior plataforma possui cerca de 1,9 bilhão de usuários (YOUTUBE, 2019b).

Os usuários das plataformas podem ser divididos em dois grupos: os consumidores e os criadores de conteúdo, ambos os grupos podem consumir conteúdo de terceiros. Entretanto, os criadores também produzem o seu próprio conteúdo para um determinado grupo de consumidores que é estabelecido pelos algoritmos de recomendação presentes em cada uma das plataformas. O objetivo desses algoritmos é a retenção de usuários para a plataforma e para a criação de um público alvo compatível com os conteúdos compartilhados pelos criadores na plataforma (ZHOU; KHEMMARAT; GAO, 2010). Dessa maneira tanto os criadores quanto os consumidores estarão com seus interesses satisfeitos na plataforma. A audiência de um criador está limitada aos usuários daquela plataforma e, por isso, o criador pode recorrer às demais plataformas para atrair um público maior e aumentar a sua presença online. Entretanto, as métricas de visualização e aprovação serão distintas em cada plataforma.

Alguns estudos (SOLTANI; ABHARI, 2013; LONG; JUNG, 2015) propõem técnicas para o reconhecimento de usuários comuns em mídias sociais[2] distintas, problema esse conhecido como correspondência de identidade, que busca demonstrar a distribuição de usuários em grupos formados pela adoção dessas mídias sociais. A Figura 1.1 ilustra a organização desses grupos de usuários.

[1] <www.youtube.com>, <www.dailymotion.com>, <www.vimeo.com>
[2] Sites de Redes Sociais e Plataformas de Vídeo.

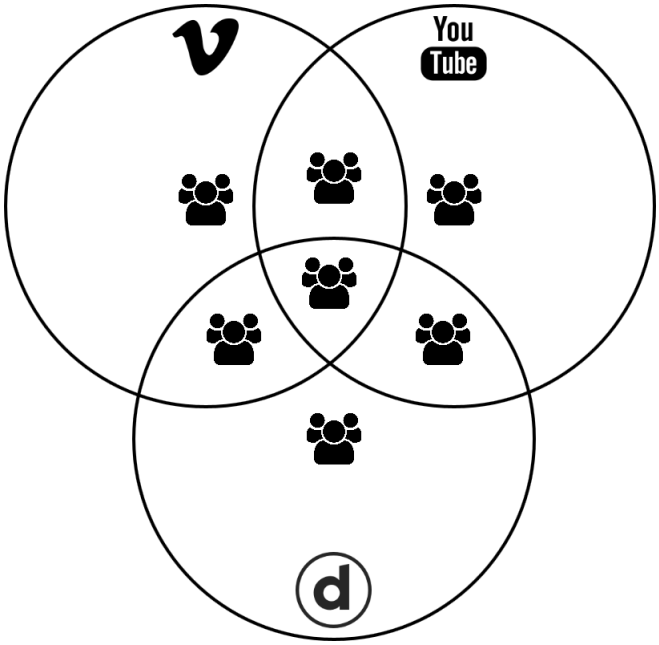

Figura 1.1 – Grupos de Usuários

Considerando que um usuário pode criar um perfil em mais de um site, esse usuário compõe um dos grupos de interseções, por exemplo, Alice criou um perfil no Dailymotion e outro no YouTube, Bob criou um perfil no Dailymotion, Vimeo e YouTube, conforme a Figura 1.1 Alice estaria no grupo coberto pelo círculo do Dailymotion e YouTube, e Bob no grupo central coberto pelos círculos das três plataformas.

Nas seções a seguir ressalta-se a importância do trabalho na Seção 1.1, seguida pelos objetivos do trabalho na Seção 1.2, seguida da Seção 1.3 que apresenta uma visão geral do trabalho.

1.1 Justificativa

O crescimento das mídias sociais geraram um aumento na quantidade de informações compartilhadas. (BELLO-ORGAZ; JUNG; CAMACHO, 2016) ressaltam a importância do desenvolvimento de novas ferramentas e técnicas de correspondência de identidade devido o aumento do volume de dados. Os avanços apresentados na literatura, (SOLTANI; ABHARI, 2013; LONG; JUNG, 2015), utilizam bases de dados de redes sociais para a correspondência de usuários. Esses trabalhos demonstram que uma base de dados mais expressiva pode ser gerada com registros de usuários unificados. O método proposto por (SOLTANI; ABHARI, 2013) até o momento é o único que utilizou a base de dados do YouTube como parte do seu método de similaridade baseado em análise semântica. Entretanto, não realiza nenhuma comparação de registros da base do YouTube com registros nas demais bases.

1.2 Objetivos

O objetivo geral do trabalho consiste em desenvolver um método de correspondência de identidade voltado para usuários criadores de conteúdo em plataformas de vídeo online (Dailymotion, Vimeo e YouTube). Assim, os criadores de conteúdo que possuem conta em mais de uma plataforma podem ser detectados e agrupados. Dessa maneira, propõem-se os seguintes objetivos específicos: Implementar coletores de dados para as plataformas de vídeo online; Criação de uma base de dados de usuários e seus respectivos conteúdos por meio da coleta de conteúdo em cada plataforma; Implementar uma técnica de correspondência de identidades para as três plataformas; Desenvolver um método de similaridade específico para a correspondência de identidade em plataformas de vídeo.

1.3 Organização do Trabalho

Este trabalho está estruturado na seguinte maneira: Capítulo 2 apresenta trabalhos relacionados; Capítulo 3 apresenta o desenvolvimento da ferramenta proposta; O Capítulo 4 apresenta os resultados obtidos pela ferramenta; o Capítulo 5 apresenta as conclusões e os trabalhos futuros.

2 Trabalhos Relacionados

Um dos primeiros registros do problema da Correspondência de Identidade foi proposto por (NEWCOMBE et al., 1959) tratava-se de um estudo de acompanhamento de indivíduos contaminados por radiação e os efeitos em seus descendentes. A técnica é composta por métodos de exploração, similaridade e correspondência. O método de exploração busca os registros de uma base em outras bases conforme um conjunto de atributos índices[1], enquanto o método de similaridade determina o grau de similaridade[2] entre os registros para então o método de correspondência determinar a ligação entre eles.

Cinquenta anos depois, (VOSECKY; HONG; SHEN, 2009) publicam um dos primeiros trabalhos de correspondência de identidade em redes sociais utilizando as bases do Facebook[3] e do StudiVZ[4]. Para isso o autor propõe um método de coleta de dados que extrai informações dos perfis nas redes sociais e armazena em um formato de dicionário. A técnica de (VOSECKY; HONG; SHEN, 2009) é similar a de (NEWCOMBE et al., 1959) no método de exploração. Entretanto, distinguia-se na utilização de três métodos de similaridade. Os métodos de similaridade são: *exato*, para valores predeterminados; *parcial*, quando o valor de um dos registros pode compor outro[5]; e *difuso*[6], para atributos sujeitos a erros, como nomes e sobrenomes por exemplo.

A abordagem de (RAAD; CHBEIR; DIPANDA, 2010) dispensa a coleta, devido à geração automática de perfis e propõe quatro métodos de similaridades. Cada um dos métodos calculam a similaridade de quatro grupos distintos de atributos; são eles: distância de edição para atributos numéricos e baseados em URL, algoritmo de Jaro para atributos únicos e sem sentido, TF-IDF para atributos múltiplos sem sentido e análise semântica explicíta para atributos baseados em semântica.

A técnica de (SOLTANI; ABHARI, 2013) é proposta por um método de coleta, e quatro métodos de similaridade, sendo dois deles baseados nos métodos exato e difuso de (VOSECKY; HONG; SHEN, 2009), um outro baseado no método de análise semântica proposto por (RAAD; CHBEIR; DIPANDA, 2010) e o último, uma análise do conjunto de amigos comuns. O método de análise semântica de (SOLTANI; ABHARI, 2013) busca categorizar as publicações nas redes sociais com o auxílio de APIs, uma delas é a YouTube Data API V3, utilizada por este trabalho. O método de análise de círculo de amigos consiste em encontrar nas outras plataformas um subconjunto de amigos semelhantes nos registros de amizades.

---

1 Algumas técnicas levam em consideração a distribuição estatística desses atributos.

2 Valores da similaridade de atributos calculados por algoritmos como distâncias de edição para atributos textuais ou subtração e divisões para numéricos

3 <www.facebook.com>

4 Uma rede social alemã similar ao Facebook <www.studivz.net>

5 *String* e *substring*.

6 no inglês: *fuzzy*

O trabalho mais recente e também considerado como o estado da arte é o de (LONG; JUNG, 2015). A técnica consiste de um método de coleta baseado no consentimento dos usuário em fornecer as informações, em dois métodos de similaridade e em um método correspondência próprio. O primeiro método de similaridade é baseado na distância de edição e na medição de força [7] da string, enquanto o segundo é similar ao de (SOLTANI; ABHARI, 2013) e analisa o círculo social dos registros. O método de correspondência é o algoritmo do método húngaro, o mesmo utilizado na cobertura de vértices em grafos.

2.1 Plataformas de Vídeo Online

Plataformas de Vídeo Online são serviços de mídias sociais que oferecem ao público duas funcionalidades da Web 2.0: compartilhamento de conteúdos em vídeo e redes sociais através de uma infraestrutura de servidores e redes de distribuição de conteúdos[8] (SAXENA; SHARAN; FAHMY, 2008).

Para compartilhar vídeos nas plataformas, é necessário possuir uma conta ativa na plataforma, realizar o *upload* do vídeo e escolher o tipo de listagem [9]. As plataformas também disponibilizam um serviço de gerenciamento de vídeos enviados, onde o criador pode editar informações sobre o vídeo e ter acesso a estatísticas de cada vídeo.

A Tabela 2.1 apresenta um comparativo das principais características das três plataformas exploradas neste trabalho. As seções a seguir apresentam uma breve descrição de cada uma delas.

2.1.1 YouTube

O YouTube começou a ser desenvolvido em fevereiro de 2005 por Steve Chen, Jawed Karim e Chad Hurley, ficando pronto somente em 25 de abril do mesmo ano, data registrada no primeiro vídeo da plataforma chamado “Me at the zoo” (ALLOCCA, 2018).

Para se destacar dos demais competidores, os desenvolvedores criaram algoritmos para mapear os interesses dos usuários e personalizar a experiência de cada um deles na plataforma de acordo com a experiência individual e da multidão. Segundo (ALLOCCA, 2018) os primeiros algoritmos atuavam sobre um conjunto de tags presentes em cada vídeo agrupando-os com outros com as mesmas tags, com o crescimento da plataforma veio a adoção de algorítimos de aprendizado de máquina que incorporou os chamados hábitos de visualização que são padrões do comportamento da multidão utilizados para recomendação de vídeos para os usuários.

[7] Semelhante ao cálculo da força de senhas em sites.
[8] no inglês: Content Distribution Networks CDNs
[9] Os vídeos podem ser listados como públicos de forma que qualquer usuário pode encontrar através da busca ou como privados onde é necessário possuir o link do vídeo para ter acesso

2.1.2 Dailymotion

Lançado em 15 de Março de 2005 por Benjamin Bejbaum e Olivier Poitrey, dois ex-funcionários de uma empresa de hospedagem de conteúdo na web, o Dailymotion é uma plataforma de vídeo criada na França com um mecanismo de codificação de vídeo e hospedagem própria.

Outras plataformas como o YouTube utilizavam mecanismos de codificação de vídeo de terceiros, os desenvolvedores do Dailymotion optaram por desenvolver o seu próprio devido ao alto custo de aquisição. O mecanismo de codificação do Dailymotion além de ter gerado uma economia para empresa, mantém o aspecto estereofônico da trilha sonora do vídeo intacto (LE-MONDE, 2006).

2.1.3 Vimeo

Diferente do YouTube e do Dailymotion, o Vimeo teve seu desenvolvimento direcionado para os criadores de conteúdo, foi projetado por dois cineastas Jakob Lodwick e Zach Klein que procuravam divulgar seu trabalho na web.

O foco da plataforma se encontrava em atrair um público artístico que formou uma comunidade mais profissional e por isso segundo (LARSSON, 2013): “no Vimeo é mais provável que você receba críticas construtivas na seção de comentários”.

A plataforma oferece planos pagos e é isenta de publicidade, também é conhecida por ter uma alta qualidade, sendo a primeira a adotar a tecnologia HD e possuir alta qualidade de vídeo, atualmente atua até com 8K (VIMEO, 2019a).

	Dailymotion	Vimeo	YouTube
Visitantes Únicos Mensais	32,6 Milhões	37,7 Milhões	230,5 Milhões
Duração Média dos Acessos	161 segundos	244 segundos	514 segundos
Preço	Gratuito	a partir de 28 reais mensais	Gratuito
Resolução de Vídeo Máxima	1080p FHD	8K 4320p UHD	8K 4320p UHD
Framerate Máximo	25 FPS	60 FPS	60 FPS
Bitrate	1,2 - 4 Mbps p/ 1080p	50-80 Mbps p/ 4320p	44-56 Mbps p/ 4320p
Limite de Upload	96 vídeos ou duas horas de conteúdo por dia	Varia de acordo com o plano	100 vídeos
Tamanho Máximo de Vídeo	2GB	Limitado a um armazenamento total de 7TB por conta premium.	128 GB ou 12 horas

Fonte: Informações sobre acessos: (ALEXA, 2019).
Informações Técnicas sobre as plataformas de vídeo: (DAILYMOTION, 2019b; VIMEO, 2019a; YOUTUBE, 2019a).

Tabela 2.1 – Comparativo das Plataformas de Vídeo

3 Desenvolvimento

Neste capítulo, é descrito o desenvolvimento da ferramenta proposta, que envolve métodos de coleta, correspondência de identidade e visualização de dados. A aplicação proposta divide-se em três módulos: Coleta, Correspondência de Identidade e Visualização de Dados. A Figura 3.1 oferece uma visão geral dos componentes da ferramenta, descritos a seguir.

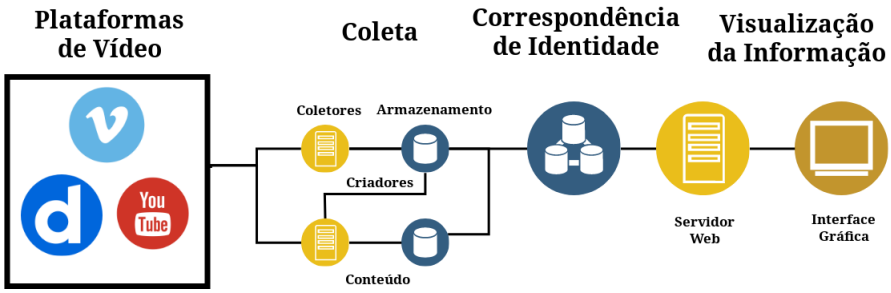

Figura 3.1 – Módulos da Ferramenta

Para cada plataforma de vídeo existem coletores e bases de dados específicos, que se dividem em usuário e conteúdo. Os coletores de usuário realizam a coleta e armazenam os resultados na base de dados da plataforma, em seguida, os coletores de conteúdo recuperam os usuários e coletam o conteúdo enviado por cada um deles, novamente, armazenando o resultado na base de dados da plataforma. Com as bases de dados geradas a correspondência de identidade busca unificar os dados, agrupando os registros semelhantes. Por fim, a visualização da informação é composta por um servidor Web cuja finalidade é disponibilizar os dados em uma interface gráfica Web com funcionalidades de pesquisa, perfil pessoal e visualização de vídeo em múltiplas plataformas.

O restante deste capítulo está organizado da seguinte forma. A Seção 3.1 descreve os coletores de dados e a maneira como foram implementados. A Seção 3.2 descreve a técnica de correspondência de identidade e os métodos de exploração, similaridade e correspondência. Por fim, a Seção 3.3 apresenta uma especificação para o módulo de visualização da informação contida na base de dados unificada. Devido à limitação de tempo, o módulo de visualização não foi implementado no escopo deste trabalho.

3.1 Coleta de Dados

A Coleta de Dados é realizada por requisições GET do protocolo HTTP, cada requisição refere-se a um tipo de recurso (usuário ou vídeo) e, para ser realizada, necessita de atributos que determinam um conjunto de registros compatíveis com a requisição. Em certos casos, o número de registros extrapolam o limite máximo da plataforma ou do coletor e por isso a resposta

da requisição é dividida em páginas pelo servidor, que são posteriormente coletadas em uma sequência de requisições GET. Cada uma dessas requisições contém os mesmos atributos, com os mesmos valores, se diferenciando apenas no atributo de página, que é atualizado pela resposta obtida pela requisição anterior. Os registros coletados são armazenados em um banco de dados. Como cada plataforma de vídeo tem um modelo de dados diferente, é necessário implementar coletores compatíveis com a API de dados de cada plataforma e armazenar os dados recuperados em bases de dados diferentes.

Além da compatibilidade dos coletores entre cada plataforma, a base de dados necessária para a execução do algoritmo de correspondência de identidade deve ser composta de dados sobre usuários e vídeos. Na implementação de cada API usuários e vídeos são implementados como recursos diferentes e, consequentemente, requisições diferentes, devido a isso foram implementados um coletor para cada tipo de recurso nas três plataformas.

Todas as implementações de coletores foram feitas utilizando a linguagem de programação Python com os pacotes das plataformas que permitem acessar as APIs de dados e realizar as operações. Todas os registros coletados são armazenados em um banco de dados não relacional MongoDB (Apêndice A).

3.1.1 Coleta de Usuários

O primeiro passo da execução do coletor de usuários, como apresentado no Algoritmo 3.1, é um laço de repetição em uma quantidade de iterações determinada pelo atributo `requests` (linha 2). Em seguida é realizada uma requisição GET da API de dados (linha 3), o armazenamento dos dados respondidos pelo servidor na base de dados de usuários (linha 4) e a verificação se existe uma próxima página (linha 5), se sim, o valor da próxima página é atualizado (linha 6), se não, ocorre uma interrupção e a coleta termina (linha 9).

Algoritmo 3.1: Coleta de usuários

Entrada: max-results, order, page, q, region-code, requests, FIELDS
Resultado: Base de dados da plataforma

```
1  início
2      para i ∈ {1, ...requests} faça
3          resposta = api.get('user', max-results, order, page, q, region-code, FIELDS);
4          db.basePlataforma.insert(resposta['data']);
5          se resposta['nextPage'] ≠ Nulo então
6              page = resposta['nextpage'];
7          fim
8          senão
9              Pare;
10         fim
11     fim
12 fim
```

Os atributos listados no método `api.get`, com exceção de `FIELDS`, estão listados na Tabela 3.1 e estão implementados nos coletores de usuários de todas as plataformas. O propósito dos atributos de coleta é a definição de critérios para a execução do coletor. O atributo `FIELDS` é uma *string* que contém o nome de todos os atributos presentes nos modelos de dados das plataformas e serve ao propósito de obter os valores desses atributos. O modelo de dados de cada plataforma está disponível nos Apêndices B, C e D.

Parâmetro	Descrição	Valor Padrão
`region-code`	Trata-se de um código de localização de dois dígitos padrão ISO 3166-1 Alpha 2. Não disponível na plataforma Vimeo.	US
`max-results`	Quantidade de resultados retornados por requisição, pode ser um valor entre 1 e 100	YouTube 50, Vimeo e Dailymotion 100.
`requests`	Número de requisições a serem feitas.	1
`page`	Número da página de resultados a ser retornados, deve ser um número maior igual a 1.	1
`q`	Termo textual para ser consultado	Termo vazio.
`order`	Critérios de ordenação da requisição	`Date`.

Tabela 3.1 – Parâmetros de Coleta de Dados

3.1.2 Coleta de Conteúdo

A coleta de conteúdo é dependente da base de dados gerada pela coleta de usuários. As requisições de coleta de conteúdo são realizadas para cada registro da base de usuários, consequentemente o número de requisições por conteúdo é muito maior e por isso a implementação da coleta de conteúdo armazena em sua base de dados informações sobre as requisições realizadas.

O primeiro passo da execução do coletor de conteúdo, como apresentado no Algoritmo 3.2, é recuperar as requisições não concluídas na base de dados (linha 2) e a partir destas recuperar os usuários (linha 3) ao qual pertencem. Em seguida para cada usuário recuperado (linha 5), um laço de repetição (linha 6) realiza uma requisição de conteúdo utilizando o método GET (linha 7), o conteúdo recuperado é inserido na base de dados (linha 8), a informação referente a requisição desse usuário é atualizada, e o valor da próxima página é atualizado (linha 9). Em caso de uma falha o coletor continua no ponto em que parou devido aos registros de requisições, o que contribui na redução do número de requisições realizadas pela API.

3.2 Correspondência de Identidade

Para ingressar em um site de mídia social o usuário deve criar uma conta, para isso ele repassa informações pessoais para a base de dados do site. Considerando a existência de múltiplos sites na Web é possível que o usuário possua contas em vários deles. Com as informações repas-

Algoritmo 3.2: Coleta de conteúdo

Entrada: max-results, users, requests
Resultado: Base de dados da plataforma

```
1  início
2      requisicoes = db.baseRequisicoes.read(completa = Falso);
3      usuarios = db.basePlataforma.read(id ∈ requisicoes.id);
4      para usuario ∈ usuarios faça
5          enquanto resposta['nextPage'] ≠ Nulo faça
6              resposta = api.get('video', usuario.id, page, FIELDS);
7              db.basePlataforma.insert(resposta['dados']);
8              db.baseRequisicoes.update(usuario.id, resultado['page'],
                 tamanho(resultado['videos']));
9              page = resposta['nextPage'];
10         fim
11     fim
12 fim
```

sadas a cada um dos sites no momento do cadastro é possível determinar a existência de registros de um mesmo usuário em variadas plataformas. Esse problema, denominado **Correspondência de Identidade**, foi solucionado pela primeira vez por (NEWCOMBE et al., 1959) através de uma técnica que é baseada na implementação de três métodos: *exploração*, *similaridade* e *correspondência*. Cada um desses métodos estão detalhados nas subseções a seguir.

A implementação dos métodos de exploração e similaridade atuam apenas com duas bases de dados. No total existem três bases de usuário, uma de cada plataforma e por isso, uma combinação de duas bases entre as três é realizada para a execução desse métodos, a Figura 3.2 apresenta um esquemático dessas combinações.

Para cada combinação de bases, uma coleção é criada no MongoDB para armazenar os resultados dos métodos de exploração e similaridade. Essas coleções chamadas de pares de bases são geradas pela combinação das bases das plataformas, são elas: YouTube-Dailymotion `YTDM`, YouTube-Vimeo `YTVM` e Dailymotion-Vimeo `DMVM`.

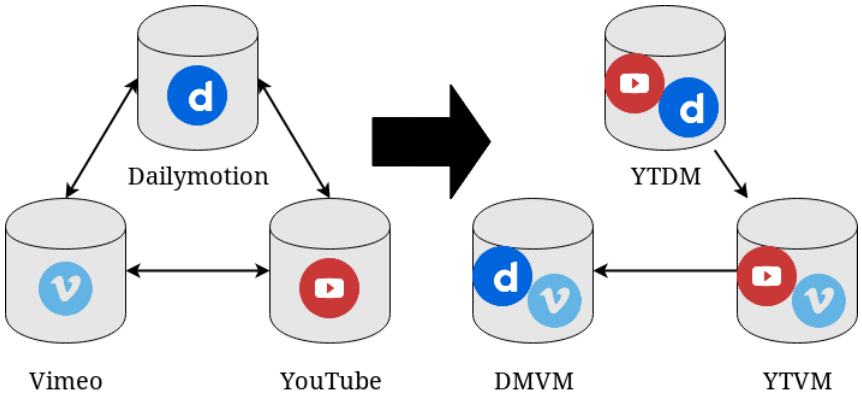

Figura 3.2 – Correspondência de Identidade com três bases de dados distintas

Algoritmo 3.3: Correspondência de Identidade

```
   Entrada: dm, vm, yt
   Resultado: Base de dados unificada
1  início
2      ytdm = gerar_par_base(yt, dm);
3      ytvm = gerar_par_base(yt, vm);
4      dmvm = gerar_par_base(dm, vm);
5      correspondencia(ytdm, ytvm, dmvm);
6  fim
```

Algoritmo 3.4: Gerar Par de Base

```
   Entrada: base_a, base_b
   Resultado: Par de Base A B
1  início
2      para reg_base_a ∈ base_a faça
3          compativeis = exploracao(reg_base_a, base_b);
4          para compativel ∈ compativeis faça
5              grau_similaridade = similaridade(reg_base_a, compativel);
6              db.base_a_b.insert(reg_base_a, compativel, grau_similaridade);
7          fim
8      fim
9      retorna db.base_a_b
10 fim
```

O Algoritmo 3.3 apresenta uma visão geral da técnica de correspondência de identidade. As bases de dados de cada plataforma são utilizadas para a construção de três pares de base (linhas $2-4$), conforme descrito pelo Algoritmo 3.4 adiante. Em seguida, são repassadas ao método de correspondência de identidade (linha 5) onde serão relacionadas com os registros mais similares. A implementação do método de correspondência é detalhada na Subseção 3.2.3

O par de base é construído da aplicação dos métodos de exploração e similaridade, para isso, a entrada do Algoritmo 3.4 é composta por duas bases A (`base_a`) e B (`base_b`). O algoritmo se inicia em um laço de repetição (linha 2) encarregado de selecionar um registro (`reg_base_a`) membro da base A por iteração. Em seguida, o método de exploração (linha 3) seleciona os registros da base B compatíveis com o registro da base A. Para cada registro compatível encontrado, um laço de repetição (linha 4) executa o método de similaridade (linha 5) e insere (linha 6) o resultado da similaridade em uma nova base (`base_a_b`). Os resultados inseridos na nova base são compostos pelo identificador dos registros e pelo grau de similaridade, descrito na Subseção 3.2.2.

3.2.1 Método de Exploração

Para retornar um conjunto de registros da base B compatíveis com cada registro da base A, o método de exploração realiza pesquisas nas bases de dados pelo conjunto de atributos índices.

A implementação do método de exploração recorre ao operador `$text` do MongoDB que permite realizar pesquisas de `strings` com técnicas de processamento de textos. Segundo (MONGODB, 2019) o operador `$text` realiza a stemização [1] das palavras, e o processamento de palavras vazias[2], ou seja realiza a redução das palavras em sua forma base e isola as palavras mais comuns da língua para realizar a pesquisa. O resultado dessa técnica é um conjunto de registros que compartilha valores comuns nos atributos índices.

Os atributos índices utilizados referem-se ao nome utilizado pelo usuário nas plataformas, definido nos atributos: `screenname` da base de dados Dailymotion (Apêndice B), `name` do Vimeo (Apêndice C) e `title` do YouTube (Apêndice D).

3.2.2 Método de Similaridade

O método de similaridade é instanciado para cada registro da base A acompanhado de um conjunto de registros compatíveis extraídos da base B pelo método de exploração.

O objetivo do método de similaridade é definir o grau de similaridade entre dois registros. Na literatura, (RAAD; CHBEIR; DIPANDA, 2010; LONG; JUNG, 2015), a técnica mais utilizada aplica algoritmos de similaridade de `strings` como a distância de edição ou distância Levenshtein. O algoritmo da distância de edição determina uma grau de similaridade embasado no número de operações necessárias para equiparar um *string* ao outro.

Uma outra característica dos trabalhos de correspondência de identidade é a implementação de vários métodos de similaridade. Neste trabalho, são implementados dois métodos. O primeiro baseado na distância de edição entre os mesmos atributos índices utilizado pelo método de exploração. O segundo é um método que busca nas bases de dados o conteúdo em comum entre os registros e retorna um percentual do conteúdo comum.

A implementação do método de similaridade baseado na distância de edição segue o paradigma da programação dinâmica, onde através de uma matriz é calculado o número mínimo de operações para igualar uma *string* a outra. O resultado então é aplicado na Equação 3.1 para definir um grau de similaridade entre os atributos dos dois registros.

$$p(s_1, s_2) = 1 - \frac{distancia\_edicao(s_1, s_2)}{max(|s_1|, |s_2|)} \tag{3.1}$$

onde:

[1] no inglês: *stemming*
[2] no inglês: *stop words*

- s_1, s_2 são as *strings* a serem comparadas.
- `distancia_edicao` trata-se do número de operações necessárias para assemelhar as duas *strings*.
- $|s_1|, |s_2|$ é o comprimento das duas strings $s_1 e s_2$
- `max` é uma função que retorna o atributo de maior valor, neste caso, retornando o tamanho da maior *string*.

Quando o grau de similaridade resultante da Equação 3.1 é próximo de 1 significa que trata-se de duas *strings* bem semelhantes, enquanto próximo de 0 significa o oposto.

Neste trabalho a correspondência entre dois registros é estabelecida através do grau de similaridade entre os atributos índices e a quantidade de vídeos em comum. O Algoritmo 3.5 demonstra o cálculo dessa medida nos pares de bases. A execução do algoritmo inicia-se com o cálculo da distância de edição entre os registros dos pares de bases com seus respectivos registros compatíveis (linhas $2-4$). Com os valores de obtidos pela distância de edição, os registros compatíveis presente nos pares de bases são selecionados (linhas $5-6$) conforme um valor mínimo estabelecido de 0,65. Em seguida, o método de similaridade de conteúdo é aplicado nos registros selecionados chegando no grau de similaridade.

3.2.3 Método de Correspondência

O método de correspondência recebe como entrada as bases de dados das plataformas e os pares de bases resultantes dos métodos de similaridade. Os dados referentes a estes pares de bases correspondem aos graus de similaridade calculados. O Algoritmo 3.6 realiza a verificação de correspondência de identidade.

Algoritmo 3.5: Execução dos Métodos de Similaridade

Entrada: ytdm, ytvm, dmvm, dm_videos, vm_videos, yt_videos
Resultado: Inserção do Atributo Grau de Similaridade nos Pares de Bases

```
1  início
2      ytdm = similaridade_de(ytdm);
3      ytvm = similaridade_de(ytvm);
4      dmvm = similaridade_de(dmvm);
5      ytdm = selecao(ytdm);
6      ytvm = selecao(ytvm);
7      dmvm = selecao(dmvm);
8      ytdm = similaridade_conteudo(ytdm, yt_videos, dm_videos);
9      ytvm = similaridade_conteudo(ytvm, yt_videos, vm_videos);
10     dmvm = similaridade_conteudo(dmvm, dm_videos, vm_videos);
11     retorna unificar_bases(ytdm, ytvm, dmvm);
12 fim
```

Algoritmo 3.6: Correspondência de Identidade

```
   Entrada: yt, dm, vm, ytdm, ytvm, dmvm
   Resultado: Base Unificada
1  Inicio
2      para cada R ∈ yt e R ∉ ytdm e R ∉ ytvm faça db.unificado.insert(R);
3      para cada R ∈ dm e R ∉ dmvm e R ∉ ytdm faça db.unificado.insert(R);
4      para cada R ∈ vm e R ∉ dmvm e R ∉ ytvm faça db.unificado.insert(R);
5      para cada R ∈ ytdm faça
6          unificado = getRegistroUnificado(R);
7          db.unificado.insert(unificado);
8      fim
9      para cada R ∈ ytvm faça
10         unificado = getRegistroUnificado(R);
11         se unificado in db.unificado então
12             db.unificado.update(unificado);
13         fim
14         senão
15             db.unificado.insert(unificado);
16         fim
17     fim
18     para cada R ∈ dmvm e R ∉ db.unificado faça
19         unificado = getRegistroUnificado(R);
20         db.unificado.insert(unificado);
21     fim
```

O Algoritmo 3.6 inicia realizando inserções dos registros presentes nas bases de dados das plataformas que não possuem correspondência (linhas $2-4$). Em seguida, um laço de repetição (linha 5) realiza para cada registro correspondente presente no par de base `YTDM` a seleção do registro unificado (linha 6) e a inserção desse registro na base (linha 7). Da mesma forma, entretanto com o par de base `YTVM` é realizado uma seleção do registro unificado (linha 10), que em seguida passa por uma verificação (linha 11) na base de registros unificados, se a verificação for positiva o registro unificado selecionado atualiza (linha 12) o registro unificado da base que passa a conter a correspondência entre as três plataformas, se negativo o registro unificado é inserido na base (linha 15). E por fim, para o par de base `DMVM`, os registros unificados (linha 19) são inseridos (linha 20) na base unificada. Observe que os registros que pertencem aos três pares de bases já foram contemplados nos dois laços anteriores, restando, para o último laço, apenas os registros que pertencem exclusivamente ao par `DMVM`.

3.3 Visualização dos Dados

O objetivo do método de visualização de dados é disponibilizar os dados oriundos da base de usuários unificada em uma interface web. Todos os usuários presentes nas bases de dados devem ser disponibilizados, com destaque para aqueles que possuem presença em mais de

uma plataforma. As Figuras 3.3, 3.4 e 3.5 apresentam um protótipo de interface gráfica para a visualização de dados. Por limitação de tempo a interface não foi implementada, ficando como sugestão de trabalhos futuros.

A página principal (Figura 3.3) contém, na primeira linha, alguns vídeos em destaque e, na segunda, alguns usuários. O clique em um item de usuário carrega a tela da Figura 3.4, no caso de um vídeo, a tela da Figura 3.5 é carregada.

O conteúdo em vídeo dos usuários é listado em uma interface (Figura 3.4) de modo que é possível obter informações sobre as plataformas onde determinados vídeos estão disponíveis. A métrica da quantidade de inscritos é calculada através da soma da quantidade de usuários presentes nos grupos de adoção, a tabela mostra a distribuição desses usuários pelas plataformas. Clicando em um vídeo a tela da Figura 3.5 é carregada.

Um vídeo compartilhado entre as plataformas é disponibilizado em uma interface (Figura 3.5) onde o usuário poderá ter acesso ao conteúdo na plataforma de sua preferência ao navegar pelo menu de abas. As métricas de aceitação e visualização também serão serão somadas e são obtidas através de requisições HTTP submetidas pela interface gráfica.

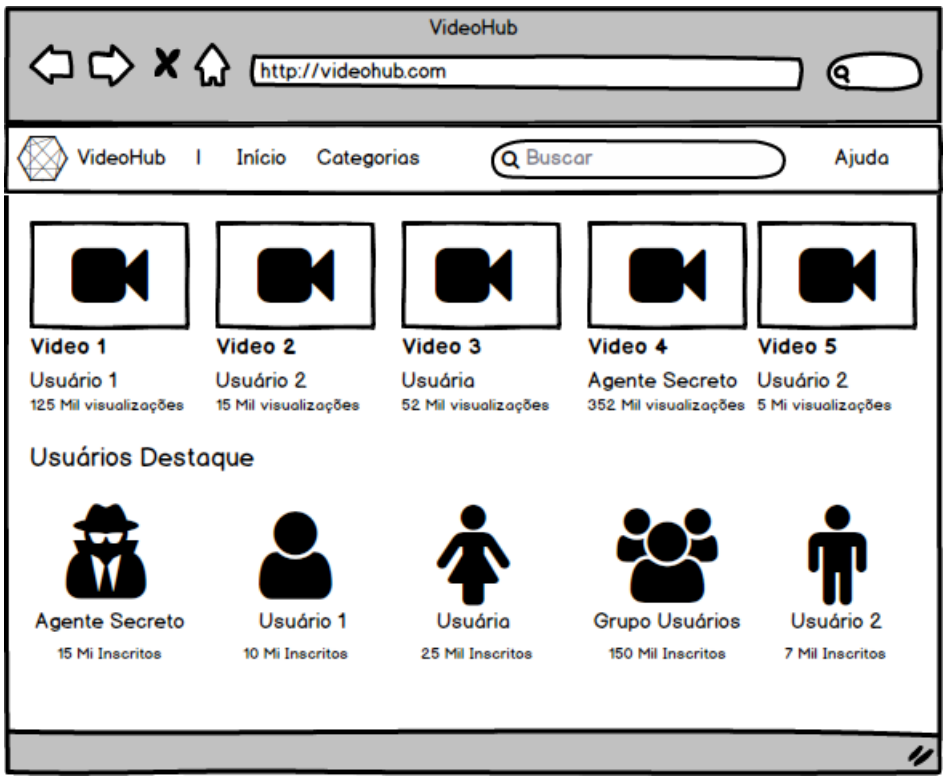

Figura 3.3 – Página Principal

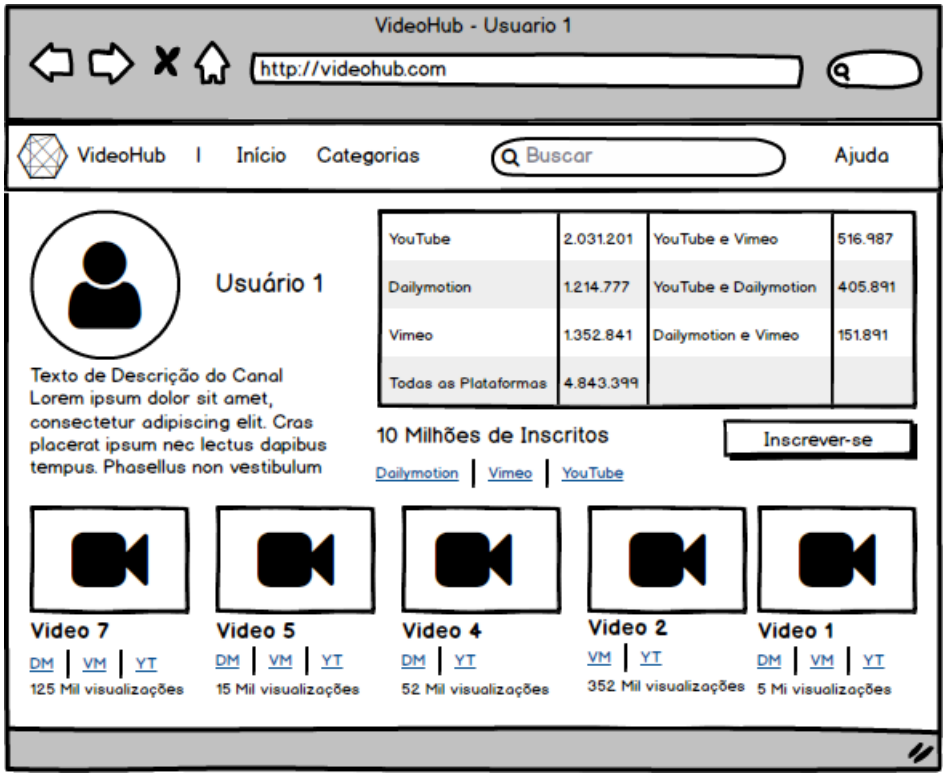

Figura 3.4 – Página do Usuário

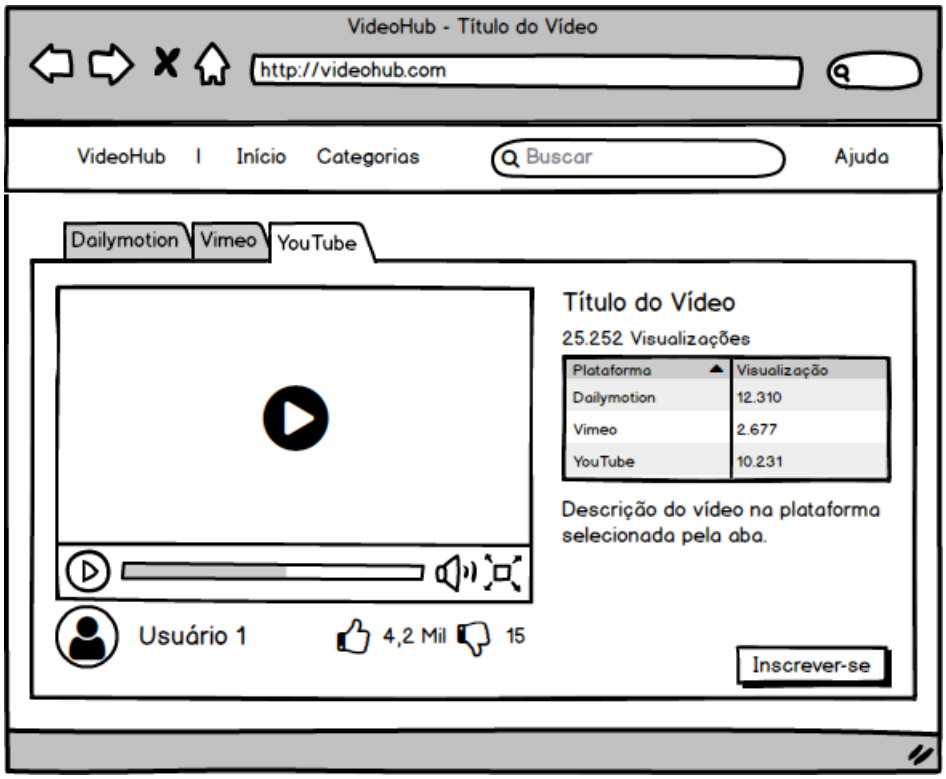

Figura 3.5 – Opção de vídeo em várias plataformas

4 Resultados

Neste capítulo são apresentados os resultados da coleta de dados, na Seção 4.1, e do método de correspondência de identidade, na Seção 4.2.

4.1 Coleta de Dados

Como já detalhado no Capítulo 3, existem dois tipos de coletores, de usuários e conteúdo. A coleta de dados de usuários das plataformas de vídeo é realizada com a passagem dos atributos de requisição como argumentos na execução do coletor. Diferentes definições desses atributos proporcionam requisições diversificadas e, consequentemente, respostas distintas, com menor número de redundância. A coleta de conteúdo é realizada a partir da base de dados de usuários construída pela coleta de usuários e resulta em uma base de dados de conteúdo e outra de requisições. Vale ressaltar que os coletores mesmo sendo bem definidos não previnem redundâncias durante a coleta e por isso cada base possui um mecanismo de unicidade dos registros, que atuam impedindo a inserção de registros redundantes.

Para a coleta de usuário são realizadas requisições utilizando os atributos de coleta presentes na Tabela 3.1. O primeiro passo da coleta de usuários em cada plataforma foi selecionar os mais populares, para isso o atributo `order` é definido com o valor específico de cada API, `popular` nas plataformas Vimeo e Dailymotion e `viewCount` no YouTube. O segundo passo consiste em adicionar o filtro de localização com o atributo `region-code`. O terceiro e último passo consiste em adicionar o atributo textual `q`.

Cada uma das plataformas possui uma limitação na quantidade de total de registros disponibilizados para um tipo de requisição. Devido a isso, cada um dos passos de coleta é realizado por um conjunto de requisições. Cada conjunto de requisições contém um atributo extra para reduzir o número de redundância nos registros coletados. No segundo e terceiro passo os valores repassados aos atributos `region-code` e `q` são diferentes. No segundo passo são realizadas requisições onde `region-code` tem como valor países onde as plataformas estão disponíveis. No terceiro passo são realizadas requisições onde `q` tem como valor as letras do alfabeto.

Os resultados da coleta de dados são apresentados na Tabela 4.1, a primeira linha representa o número de usuários coletados em cada plataforma, a segunda linha representa o número de requisições de conteúdo realizadas pelo coletor de conteúdo e a terceira o número de vídeos coletados. Conforme explicado na Seção 3.1 as requisições de conteúdo são realizadas para cada usuário, dessa forma os números da segunda linha representam o número de usuários que tiveram seus conteúdos coletados nas plataformas.

	Dailymotion	Vimeo	YouTube
Usuários	14.756	15.617	40.674
Requisições de Conteúdo	14.756	15.617	38.093
Conteúdo em Vídeo	3.722.415	652.83	3.407.943

Tabela 4.1 – Quantitativo dos dados coletados.

Devido a um limite imposto pela plataforma Dailymotion, a coleta conseguiu recuperar um valor máximo de 1000 registros de conteúdo por usuário. Na plataforma YouTube, para coletar um número de registros de conteúdo próximo ao número de usuários, a coleta teve que ser limitada a uma quantidade máxima de 250 registros de conteúdo por usuário. Vale lembrar que a base de requisições armazena informações sobre a última requisição de cada usuário sendo possível retomar a coleta dos conteúdos a partir deste ponto.

4.2 Correspondência de Identidade

A técnica de correspondência apresentada no Capítulo 3 é composta pelos métodos de exploração, similaridade e correspondência. Os métodos de exploração e similaridade atuam nos pares de bases `ytdm`, `ytvm` e `dmvm`.

O método de exploração define um conjunto de registros compatíveis `list` para cada registro, conforme apresentado na Figura 4.1. A Tabela 4.2 apresenta a quantidade média de registros compatíveis para cada registro gerado em cada par de base.

O método de correspondência recupera os registros dos pares de base com as pontuações, define os registros com correspondência e constrói a base de correspondência. A quantidade de registros com correspondência de cada grupo de usuário está listado na Tabela 4.2.

As Figuras 4.2 e 4.3 representam dois registros da nova base de dados formada pela correspondência de identidade, o primeiro exemplo (Figura 4.2) contém três usuários de fato semelhantes, enquanto o segundo exemplo (Figura 4.3), existe a ocorrência de dois falsos positivos

	YTDM	YTVM	DMVM
Média de Registros Compatíveis	4,07	9,12	0,77
Média de Registros Similares	1,87	4,19	0,04

Tabela 4.2 – Resultados dos Métodos de Exploração e Similaridade.

	Registros
Todas as Bases	5536
YouTube e Dailymotion	5565
YouTube e Vimeo	6098
Vimeo e Dailymotion	940

Tabela 4.3 – Quantidade de registros de correspondência.

nos atributos `dailymotion` e `vimeo`. Cada um dos falsos positivos são usuários diferentes do usuário no atributo `YouTube`, mas que foram considerados usuários correspondentes a ele.

A causa dos falsos positivos e falsos negativos estão relacionados a uma grande quantidade de registros que compartilham valores comuns nos atributos índices, por exemplo: *News*, *Football*, *Cartoon*, *Sports*, etc. No caso da Figura 4.3 todos os registros contém como maior *substring* *Entertainment*, que possui baixa relevância na identificação e gera uma pontuação de similaridade maior.

Para reduzir a quantidade de falsos negativos é proposto como trabalho futuro a implementação de um método de similaridade baseado na métrica TF-IDF para o cálculo da relevância de cada *substring* presente nos atributos índices de cada registro do conjunto de registros compatíveis. Dessa forma, é possível que a influência da pontuação dessas *substrings* possa ser reduzida na pontuação final, potencialmente reduzindo a incidência de erros.

Os dados contidos nas variáveis: `similars_dmvm`, `similars_ytdm` e `similars_ytvm` foram ocultados para serem comportados no trabalho. Essas variáveis armazenam listas de registros candidatos que possuem uma similaridade próxima da selecionada.

```
{
  "_id":"5d245d22cc3770d6558bd3d2",
  "list":[
    {"ed_result":11,"score":0.5925925925925926,
      "dailymotion_id":"5cf51a3a15a6278e1ce6ac47",
      "name":"Epic Entertainment"},
    {"ed_result":10,"score":0.6296296296296297,
      "dailymotion_id":"5cdf53b1c66ae59bf7942fcd",
      "name":"Online Entertainment"},
    {"ed_result":12,"score":0.5555555555555556,
      "dailymotion_id":"5cdccf3751069b7c0bd094d0",
      "name":"Ubisoft Entertainment"},
    {"ed_result":12,"score":0.5555555555555556,
      "dailymotion_id":"5cdf25729b8bbc5ceae33e56",
      "name":"Audio Visual Entertainment"},
    {"ed_result":12,"score":0.5555555555555556,
      "dailymotion_id":"5cde0d927cdb35d92dec7658",
      "name":"ABS-CBN Entertainment"},
    {"ed_result":12,"score":0.5555555555555556,
      "dailymotion_id":"5cddf252001fb775cd91a20c",
      "name":"True Entertainment TV"},
    {"ed_result":12,"score":0.5555555555555556,
      "dailymotion_id":"5cdcd049e734c4271568b2d8",
      "name":"E! Entertainment UK"},
  ],
  "youtube_id":"5cf19380435a2a5b3535f052",
  "similars":[
    {"dailymotion_id":"5cf51a3a15a6278e1ce6ac47","ed_result":11,
      "final_score":0.2962962962962963,"score":0.5925925925925926,
      "similars_video": Array,"name":"Epic Entertainment"},
    {"dailymotion_id":"5cdf53b1c66ae59bf7942fcd","ed_result":10,
      "final_score":0.3148148148148148,"score":0.6296296296296297,
      "similars_video": Array,"name":"Online Entertainment"},
  ]
}
```


Figura 4.1 – Modelo de Dados de Par de Base YTDM

```
{
  "_id":"5d289098e1765f72e8879c33",
  "similars_ytvm": Array,
  "dailymotion": {
    "username": "bloomberg",
    "status": "active",
    "instagram_url": "https://instagram.com/bloombergbusiness",
    "verified": true,
    "description": "Bloomberg delivers business...",
    "twitter_url": "https://twitter.com/business",
    "facebook_url": "https://facebook.com/bloombergbusiness",
    "videos_total": 14076,
    "linkedin_url": null,
    "views_total": 3532586,
    "_id": "5cddf212001fb775cd91a199",
    "id": "x1i7u28",
    "website_url": "https://www.bloomberg.com",
    "screenname": "Bloomberg",
  },
  "vimeo_id": "5d20f22df25c208fe0f820d3",
  "dailymotion_id": "5cddf212001fb775cd91a199",
  "similars_ytdm": [],
  "youtube_id": "5cc8dbb3097198bbb5be10cb",
  "vimeo": {
    "bio": "Bloomberg, the global business ...",
    "account":"live_premium",
    "name":"Bloomberg LP",
    "pictures": {},
    "uri": "/users/10633866",
    "websites": [],
    "link": "https://vimeo.com/bloomberg",
    "location": null,
    "created_time": "2012-02-28T21:24:55+00:00",
    "_id": "5d20f22df25c208fe0f820d3",
    "metadata": { "connections": { "videos": { "total": 121 } } }
  },
  "youtube": {
    "description": "Welcome to the official ...",
    "title": "Bloomberg",
    "channelId": "UCUMZ7gohGI9HcU9VNsr2FJQ",
    "publishedAt": "2006-03-09T23:17:35.000Z",
    "liveBroadcastContent": "none",
    "channelTitle": "Bloomberg",
    "_id": "5cc8dbb3097198bbb5be10cb",
    "thumbnails": {}
  },
  "similars_dmvm": []
}
```


Figura 4.2 – Caso Positivo de correspondência de Usuário

```
{
  "_id":"5d24cf458b3e0a3785b6348a",
  "dailymotion": {
    "username":"MBCEntertainment",
    "status":"active",
    "instagram_url":null,
    "verified":true,
    "description":"A perfect blend of entertainment...",
    "twitter_url":null,
    "facebook_url":null,
    "videos_total":63486,
    "linkedin_url":null,
    "views_total":4159452,
    "_id":"5cde0d537cdb35d92dec75e8",
    "id":"x22puk5",
    "website_url":null,
    "screenname":"MBC Entertainment"
  },
  "similars_ytdm":[],
  "vimeo":{
    "bio":null,
    "account":"basic",
    "name":"C more Entertainment",
    "pictures":{},
    "uri":"/users/4333324",
    "websites":[],
    "link":"https://vimeo.com/cmorenordic",
    "location":"Stockholm",
    "created_time":"2010-07-23T11:17:04+00:00",
    "_id":"5cf1d1a98ef666d950c84a81",
    "metadata":{"connections":{"videos":{"total":28}}}},
    "youtube":{
      "description":"Marvel Entertainment, LLC, ...",
      "title":"Marvel Entertainment",
      "channelId":"UCvC4D8onUfXzvjTOM-dBfEA",
      "publishedAt":"2005-06-16T12:09:27.000Z",
      "liveBroadcastContent":"none",
      "channelTitle":"Marvel Entertainment",
      "_id":"5cc8dbb3097198bbb5be10c4",
      "thumbnails":{},
    }
  "similars_dmvm":[],
  "similars_ytvm":[]
}
```


Figura 4.3 – Caso Falso Positivo de Correspondência

5 Conclusão

Neste trabalho, foi proposto um método de correspondência de identidade voltado para usuários das plataformas de vídeo online, trata-se de uma abordagem inédita nesse tipo de base de dados. As plataformas de vídeo e redes sociais compartilham semelhanças em suas bases de dados e, devido a isso, métodos baseados em redes sociais possuem compatibilidade com as bases de dados de plataformas de vídeo.

Para a construção da base de dados o método implementado utilizou as APIs de dados de cada plataforma para a realização de requisições de coleta. A partir do volume de dados presente nas bases de dados construídas pelos coletores, constata-se que as bases de dados coletadas de plataformas de vídeo são maiores que as bases de dados de redes sociais coletadas por (SOLTANI; ABHARI, 2013). Isso se deve às políticas de privacidade das plataformas de vídeo serem menos rigorosas, uma vez que os dados coletados são de usuários populares e podem ser tratados pelos sistemas legais como pessoas públicas.

Com as bases de dados de usuários e conteúdos finalizadas, é possível a execução dos métodos presentes na técnica de correspondência de identidade. Apesar do bom desempenho do método de exploração na definição de um conjunto de registros compatíveis, o método de similaridade de conteúdo tem apresentado um tempo de execução alto. Isso se deve ao número de comparações realizadas entre o conteúdo das duas bases ser muito alto e às limitações da coleta de conteúdo apresentadas no Capítulo 4.

Os trabalhos futuros envolvem a implementação de um método de similaridade paralelizável para conjuntos independentes de dados no método de similaridade. Implementação de um método de similaridade baseado na métrica TF-IDF. Por fim, a implementação do servidor web e da interface web para visualização dos dados descrita no Capítulo 3.

Referências

ALEXA. *The top 500 sites on the web*. 2019. Acessado em: 01/07/2019. Disponível em: <https://www.alexa.com/topsites>.

ALLOCCA, K. *Videocracy: How YouTube Is Changing the World... with Double Rainbows, Singing Foxes, and Other Trends We Can't Stop Watching*. [S.l.]: Bloomsbury Publishing USA, 2018.

BELLO-ORGAZ, G.; JUNG, J. J.; CAMACHO, D. Social big data: Recent achievements and new challenges. *Information Fusion*, Elsevier, v. 28, p. 45–59, 2016.

CHODOROW, K. *MongoDB: the definitive guide: powerful and scalable data storage*. [S.l.]: "O'Reilly Media, Inc.", 2013.

CROCKFORD, D. *The application/json media type for javascript object notation (json)*. [S.l.], 2006.

DAILYMOTION. *Data API overview*. 2019. Acessado em 27/05/2019. Disponível em: <https://developer.dailymotion.com/api>.

DAILYMOTION. *Melhores dicas / Guia para fazer o upload de vídeos*. 2019. Acessado em 27/05/2019. Disponível em: <https://www.dailymotion.com/upload/faq>.

FEYNMAN, R. P. *Surely you're joking, Mr. Feynman!* [S.l.]: Random House, 1992.

IWS, I. W. S. *World Internet Users Statistics and 2019 World Population Stats*. 2019. Acessado em: 26/03/2019. Disponível em: <https://internetworldstats.com/stats.htm>.

LARSSON, E. *5 Reasons to Choose Vimeo Instead of YouTube*. 2013. Acessado em: 09/05/2019. Disponível em: <https://mashable.com/2013/05/30/vimeo-over-youtube/>.

LE-MONDE. *Dailymotion, le YouTube à la française*. 2006. Acessado em: 09/05/2019. Disponível em: <https://web.archive.org/web/20061025035101/https://www.lemonde.fr/web/article/0,1-0@2-651865,36-822036,0.html>.

LONG, N. H.; JUNG, J. J. Privacy-aware framework for matching online social identities in multiple social networking services. *Cybernetics and Systems*, Taylor & Francis, v. 46, n. 1-2, p. 69–83, 2015.

MONGODB. *$text - Evaluation Query Operators*. 2019. Acessado em 25/07/2019. Disponível em: <https://docs.mongodb.com/manual/reference/operator/query/text/index.html#definition>.

NEWCOMBE, H. B. et al. Automatic linkage of vital records. *Science*, American Association for the Advancement of Science, v. 130, n. 3381, p. 954–959, 1959. ISSN 00368075, 10959203. Disponível em: <http://www.jstor.org/stable/1756667>.

RAAD, E.; CHBEIR, R.; DIPANDA, A. User profile matching in social networks. p. 297–304, 2010.

SAXENA, M.; SHARAN, U.; FAHMY, S. Analyzing video services in web 2.0: a global perspective. In: ACM. *Proceedings of the 18th International Workshop on Network and Operating Systems Support for Digital Audio and Video*. [S.l.], 2008. p. 39–44.

SOLTANI, R.; ABHARI, A. Identity matching in social media platforms. p. 64–70, 2013.

VIMEO. *Help Center/Video Compression Guidelines*. 2019. Acessado em: 10/05/2019. Disponível em: <https://vimeo.com/help/compression>.

VIMEO. *Vimeo API Reference*. 2019. Acessado em: 10/05/2019. Disponível em: <https://vimeo.com/help/compression>.

VOSECKY, J.; HONG, D.; SHEN, V. Y. User identification across multiple social networks. p. 360–365, 2009.

YOUTUBE. *API Reference*. 2015. Acessado em 11/05/2019. Disponível em: <https://developers.google.com/youtube/v3/docs/?hl=pt-br>.

YOUTUBE. *Configurações recomendadas de codificação de envio*. 2019. Acessado em 10/05/2019. Disponível em: <https://support.google.com/youtube/answer/1722171?hl=pt-BR>.

YOUTUBE. *Imprensa*. 2019. Acessado em: 26/03/2019. Disponível em: <https://www.youtube.com/intl/pt-BR/yt/about/press/>.

ZHOU, R.; KHEMMARAT, S.; GAO, L. The impact of youtube recommendation system on video views. *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement*, p. 404–410, 2010.

Apêndices

APÊNDICE A – MongoDB

MongoDB é um software de banco de dados não relacional orientado a documentos. Bancos de dados orientados a documentos similares ao MongoDB operam com o conceito de "documento" em seus registros, tal abordagem possibilita o armazenamento de relações hierárquicas mais complexas e dispensa a criação de um esquema (CHODOROW, 2013).

Um registro no MongoDB, chamado de **documento**, é uma estrutura de dados no formato atributo-valor semelhante a Figura A.1.

```
{
  "atributo 1" : "Valor 1",
  "atributo 2" : 2,
  "atributo 3" : [3, 4, 5]
}
```

Figura A.1 – Formato atributo-valor

O formato atributo-valor utilizado pelo MongoDB chamado de BSON é inspirado no JSON[1] (CROCKFORD, 2006) que é formado por uma listagem de pares de atributos e valores separados por vírgulas e delimitados entre chaves. Os atributos são representados como *strings* e separados dos valores por um caractere de dois pontos. O BSON possui seus próprios tipos como também suporta os mesmos tipos presentes no formato JSON. O exemplo A.2 mostra alguns dos tipos próprios do BSON como: *ObjectId* que é único e utilizado como chave privada de cada **documento**, *Date* que segue o padrão do MongoDB[2].

```
{
  _id: ObjectId("5099803df3f4948bd2f98391"),
  name: { first: "Alan", last: "Turing" },
  birth: new Date('Jun 23, 1912'),
  death: new Date('Jun 07, 1954'),
  contribs: [ "Turing machine", "Turing test", "Turingery" ],
  views : NumberLong(1250000)
}
```

Figura A.2 – Exemplo de documento do MongoDB.

[1] do inglês: *JavaScript Object Notation*

[2] O JSON segue o padrão estabelecido pelas linguagens de programação

APÊNDICE B – Modelo de Dados do Dailymotion

B.1 Usuários

Representado na API como recurso `user` (DAILYMOTION, 2019a).

B.1.1 Requisição

```
search_response = d.get('/users', {
    'fields': FIELDS,
    'country': options.region_code,
    'limit': options.max_results,
    'page': options.page,
    'sort': options.order
})
```

B.1.2 Resposta

Atributo	Tipo	Descrição
`description`	*String*	Texto de descrição do canal na plataforma.
`username`	*String*	Nome de usuário único que identifica o canal.
`verified`	*Bool*	Atributo que indica se o usuário confirmou sua identidade.
`website_url`	*String*	URL de site na web.
`witter_url`	String	URL no Twitter.
`facebook_url`	*String*	URL no Facebook.
`instagram_url`	*String*	URL no Instagram.
`linkedin_url`	*String*	URL no Linkedin.
`screenname`	*String*	Nome utilizado no canal do usuário.
`id`	*String*	Identificador único.
`status`	*String*	Atributo que indica o estado da conta do usuário na plataforma.
`videos_total`	*Number*	Número de vídeos enviados para a plataforma

views_total	*Number*	Número de visualizações na plataforma

Tabela B.1 – Atributos de user na Plataforma Dailymotion

B.2 Conteúdo

Representado na API como recurso video (DAILYMOTION, 2019a).

B.2.1 Requisição

```
search_response = d.get('/user/' + user_id + '/videos', {
    'fields': FIELDS,
    'page': nextPage,
    'limit': n_results
})
```

B.2.2 Resposta

Atributo	Tipo	Descrição
available_formats	*Array*	Lista os formatos no qual o vídeo está disponível.
allow_embed	*Bool*	Indica se o vídeo pode ser incorporado a páginas web.
country	*String*	Código ISO 3166-1 Alpha 2 do País onde o vídeo foi publicado.
allowed_in_playlists	*Bool*	Indica se o vídeo pode ser adicionado a playlists.
created_time	*Number*	Data e hora que esse vídeo foi carregado.
description	*String*	Descrição do vídeo na plataforma.
duration	*Number*	Duração do vídeo em segundos.
geoblocking	*Array*	Lista de países onde este vídeo não está acessível.
geoloc	*Array*	Geolocalização do vídeo com latitude e longitude.
height	*Number*	Altura em pixeis do vídeo.
id	*String*	Identificador único do vídeo na plataforma.
language	*String*	Idioma do vídeo.
media_type	*String*	Tipo de mídia desse conteúdo.

`private`	*Bool*	Valor que indica se o vídeo é privado.
`private_id`	*String*	Identificador de vídeo privado.
`status`	*String*	Estado do vídeo na plataforma.
`tags`	*Array*	Lista de tags associadas ao vídeo.
`thumbnail_60_url`	*String*	URL da imagem de 60 px de resolução.
`thumbnail_120_url`	*String*	URL da imagem de 120 px de resolução.
`thumbnail_180_url`	*String*	URL da imagem de 180 px de resolução.
`thumbnail_240_url`	*String*	URL da imagem de 240 px de resolução.
`thumbnail_360_url`	*String*	URL da imagem de 360 px de resolução.
`thumbnail_480_url`	*String*	URL da imagem de 480 px de resolução.
`thumbnail_720_url`	*String*	URL da imagem de 720 px de resolução.
`thumbnail_1080_url`	*String*	URL da imagem de 1080 px de resolução.
`thumbnail_url`	*String*	URL da imagem do vídeo.
`tiny_url`	*String*	URL reduzida do vídeo.
`title`	*String*	Título do vídeo.
`updated_time`	*Date*	Data em que o vídeo foi modificado.
`url`	*String*	URL desse vídeo no Dailymotion.
`verified`	*Bool*	Indica se o criador de conteúdo verificou sua conta.
`width`	*Number*	Largura do vídeo em pixeis.

Tabela B.2 – Atributos de `video` na Plataforma Dailymotion

APÊNDICE C – Modelo de Dados do Vimeo

C.1 Usuários

Implementado na API como recurso `users` (VIMEO, 2019b).

C.1.1 Requisição

```
search_response = v.get('/users', params={
    'fields': FIELDS,
    'limit': options.max_results,
    'page': nextPage,
    'query': options.q,
    'sort': options.order
})
```

C.1.2 Resposta

Atributo	Tipo	Descrição
`uri`	*String*	Identificador único do registro.
`name`	*String*	Nome utilizado pelo usuário na plataforma.
`link`	*String*	Endereço da página pessoal na plataforma.
`location`	*String*	Localização.
`bio`	*String*	Descrição ou biografia do usuário.
`created_time`	*Date*	Data de criação de usuário.
`Pictures`	*Object*	Coleção de imagens vinculadas a conta.
`resource_key`	*String*	Chave de recursos do usuário.
`websites`	*Array*	Informações sobre sites na web e redes sociais.
`metadata`	*Object*	Metadados do usuário.
`account`	*String*	Tipo de conta que o usuário possui, isso indica se ele paga para utilizar algum recurso da plataforma.

Tabela C.1 – Atributos de `user` na Plataforma Vimeo

C.2 Conteúdo

Implementado na API como recurso `videos` (VIMEO, 2019b).

C.2.1 Requisição

```
search_response = v.get(user_id + '/videos', params={
    'fields': FIELDS,
    'page': nextPage,
    'per_page': n_results
})
```

C.2.2 Resposta

Atributo	Tipo	Descrição
`uri`	*String*	Identificador único do video.
`name`	*String*	Nome do video.
`description`	*String*	Descrição do video.
`type`	*String*	Tipo de recurso da API.
`link`	*String*	Endereço do vídeo na plataforma.
`duration`	*Number*	Duração do vídeo em segundos.
`width`	*Number*	Largura em pixeis do vídeo.
`language`	*Idioma*	O idioma principal do vídeo.
`height`	*Number*	Altura em pixeis do vídeo.
`created_time`	*Date*	Data e horário de envio do vídeo.
`content_rating`	*Array*	Classificação de conteúdo.
`pictures`	*Object*	A imagem utilizada pelo vídeo.
`tags`	*Array*	Um vetor com todas as tags atribuídas ao video
`stats`	*Object*	Fornece informação sobre o estado do video na plataforma, se está sendo enviado, está publicado, etc.
`categories`	*Array*	
`user_uri`	*String*	Identificador único do usuário que publicou o video.

Tabela C.2 – Atributos de `video` na Plataforma Vimeo

APÊNDICE D – Modelo de Dados do YouTube

D.1 Usuários

Implementado na API como recurso `channel` (YOUTUBE, 2015).

D.1.1 Requisição

```
search_response = youtube.search().list(
  part = 'snippet',
  type = 'channel',
  order = options.order,
  safeSearch = 'none',
  q = options.q,
  regionCode = options.region_code,
  maxResults = options.max_results,
  pageToken = nextPageToken
).execute()
```

D.1.2 Resposta

Atributo	Tipo	Descrição
`description`	*String*	Descrição do canal.
`title`	*String*	Título do canal.
`publishedAt`	*Date*	Data de criação do canal.
`liveBroadcastContent`	*String*	Indicador de conteúdo ao vivo.
`channelTitle`	*String*	Similar ao `title`.
`thumbnails`	*Object*	Objeto com todos os tipos de miniaturas de imagem de usuário.
`channelId`	*String*	Identificador único do canal na plataforma.

Tabela D.1 – Atributos de `channel` na Plataforma YouTube

D.2 Conteúdo

Implementado na API como recurso `playlistItems` (YOUTUBE, 2015).

D.2.1 Requisição

```
search_request = youtube.playlistItems().list(
    part = 'snippet,contentDetails',
    playlistId = channel['contentDetails']['relatedPlaylists']['uploads'],
    maxResults = 50,
    pageToken = nextPageToken
)
```

D.2.2 Resposta

Atributo	Tipo	Descrição
`playlistId`	*String*	Indicador da lista de reprodução de envios do canal.
`description`	*String*	Descrição do vídeo.
`title`	*String*	Título do vídeo.
`channelId`	*String*	Identificador do canal que enviou o vídeo.
`videoId`	*String*	Identificador único do vídeo na plataforma.
`publishedAt`	*Date*	Data de envio do vídeo.
`id`	*String*	Identificador único do vídeo na lista de reprodução de envios do canal.
`thumbnails`	*Object*	Objeto com todos os tipos de miniaturas imagens do conteúdo.

Tabela D.2 – Atributos de `playlistItems` na Plataforma YouTube

Glossário

API *Application Programming Interface* ou Interface de Programação de Aplicações é uma padronização elaborada para o reaproveitamento de métodos presentes em um software. 4, 9, 10, 18, 24, 29, 30, 32–35

Bitrate Quantidade de bits processados em um intervalo de tempo. 7

BSON Similar ao JSON, é uma representação de objetos adotada pelo banco de dados não relacional MongoDB. 28

CDN *Content Delivery Network* ou Redes de fornecimento de conteúdo são uma rede servidores que oferecem cobertura geográfica na distribuição de conteúdo. 5

Framerate Quantidade de quadros ou imagens exibidas na tela em um intervalo de tempo. 7

GET Trata-se de um método do protocolo HTTP que retorna apenas dados sobre um recurso. . 8–10

HTTP *Hypertext Transfer Protocol* ou Protocolo de Transferência de Hipertexto é um protocolo de comunicação baseado no modelo cliente-servidor. O protocolo possui métodos de requisição que atuam sobre um recurso específico. 8, 16, 36

ISO 3166-1 Alpha 2 Trata-se de um padrão de códigos de países ou territórios. Cada código possui dois caracteres e são amplamente utlizados na internet para a definição de domínios. 10, 30

JSON *JavaScript Object Notation* é uma sintaxe para a representação de objetos baseada na sintaxe da línguagem JavaScript. . 28, 36

MongoDB MongoDB é um banco de dados não relacional. 9, 11, 13, 36

Python Python é uma línguagem de programação. 9

Resolução de Vídeo Corresponde ao nível de detalhamento que cada quadro do vídeo comporta. 7

TF-IDF Medida estatística que determina o grau de importância de um termo presente em um documento em uma coleção de documentos. 4, 20, 24

URL *Uniform Resource Locator* é um endereço eletrônico que se refere a algum recurso disponibilizado pela rede. 4

Web 2.0 Trata-se de um conceito que busca criar um ecosistema de comunidades virtuais na web, baseado em redes sociais, wikis, blogs e plataformas de video. 5