

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



Olhó-passarinho: uma extensão do TweeProfiles para fotografias

Ivo Filipe Valente Mota

PARA APRECIAÇÃO POR JÚRI

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Orientador: Luís Filipe Pinto de Almeida Teixeira (PhD)

Co-orientador: Carlos Manuel Milheiro de Oliveira Pinto Soares (PhD)

21 de Julho de 2014

© Ivo Mota, 2014

Resumo

O Twitter é uma das redes sociais atuais que mais informação gera todos os dias. Face à sua dimensão, foi desenvolvido o TweeProfiles, uma ferramenta que analisa as mensagens partilhadas neste serviço. Esta ferramenta utiliza técnicas de *Data Mining* para identificar padrões, apresentados através de *clusters* de Tweets, em que são analisados, o conteúdo na forma de texto, as ligações sociais, e as dimensões espaço-temporais das mensagens.

Face ao aumento do número de utilizadores que recorrem a smartphones para acederem ao Twitter, o número de fotografias partilhadas neste serviço tem crescido significativamente nos últimos anos. Esta dissertação teve como objetivo principal o desenvolvimento de uma extensão da ferramenta TweeProfiles, através de técnicas de processamento de imagem e *data mining*, que permita a identificação de padrões espaço-temporais através da informação das imagens partilhadas no serviço de *microblogging* Twitter. Para a sua concretização foi desenvolvido um módulo que utiliza o conceito de vocabulário visual para a representação das imagens de uma forma mais compacta e eficiente.

Os resultados obtidos podem ser visualizados através de uma aplicação web que permite a navegação e visualização pelas imagens e dimensões espaço-temporais dos *clusters*.

Abstract

Twitter is one of social networks that generates more information on a continuous basic. Due to its dimension, a tool called TweeProfiles was created, which uses the messanges posted in this social network. This tool uses data mining techniques to identify patterns presented as clusters of Tweets, according to four dimensions: textual, content, social connections, spatial and temporal characteristics.

Given the increasing number of users who use smartphones to access Twitter, the number of shared photos in this service has grown significantly in recent years. The main goal of this thesis is developing an extension of the TweeProfiles tool that also looks for patterns in those images. Through techniques of computer vision and data mining, this tool enables the identification of spatio-temporal patterns using all information in the shared images. The implementation is based on the concept of visual vocabulary for representing images in a more compact and efficient way.

The results can be visualized through a web application that allows browsing and viewing the images and spatial and temporal dimensions of clusters.

Agradecimentos

Em primeiro lugar quero deixar os meus agradecimentos aos meus orientadores, Professor Luís Filipe Teixeira e Professor Carlos Pinto Soares, pela excelente colaboração e disponibilidade sempre demonstrada no desenvolvimento deste projeto de dissertação. As suas orientações foram um fator determinante para o sucesso do mesmo.

Também queria deixar o meu agradecimento ao Tiago Cunha por ter-se demonstrado sempre disponível para ajudar e esclarecer dúvidas fundamentalmente relativas ao TweeProfiles.

Não menos importante, queria aqui apresentar os meus agradecimentos pelo companheirismo e força dada por todos os meus amigos, salientando os nomes de Hugo Marques, Nuno Duarte e Pedro Ribeiro que estiveram sempre presentes durante o desenvolvimento deste projeto.

Por fim, um enorme agradecimento aos meus pais pela paciência e por me terem ajudado em todos os momentos que necessitei.

À Joana Pinto, um agradecimento especial pela sua presença, companheirismo e amizade em todos os bons e maus momentos desta longa caminhada. Por nunca me ter deixado desistir. Por ter sido o meu maior pilar.

Ivo Mota

Este trabalho é parcialmente financiado por fundos nacionais, através da FCT – Fundação para a Ciência e Tecnologia no âmbito do projeto "REACTION (UTAustin/EST-MAI/0006/2009)" bem como do projeto "NORTE-07-0124-FEDER-000059", que é financiado pelo Programa Operacional Regional do Norte de Portugal (ON.2 – O Novo Norte), sobre o Quadro de Referência Estratégico Nacional (QREN), através do Fundo de Desenvolvimento Regional Europeu (FDRE), e da agência de financiamento Portuguesa, Fundação para a Ciência e a Tecnologia (FCT).

*"Logic will get you from A to Z.
Imagination will get you everywhere."*

Albert Einstein

Conteúdo

Agradecimentos	v
1 Introdução	1
1.1 Contexto	1
1.2 Motivação	2
1.3 Objetivos	2
1.4 Estrutura do documento	2
2 Conceitos e Trabalhos Relacionados	5
2.1 Clustering	5
2.1.1 Clustering por Partição	6
2.1.2 Clustering Hierárquico	7
2.1.3 Clustering Baseado em Densidade	9
2.1.4 Clustering Baseado em Grelhas	9
2.1.5 Funções de Distância	10
2.2 TweeProfiles	13
2.2.1 Descrição e objetivos	13
2.2.2 Resultados ilustrativos	14
2.2.3 Prós e contras	16
2.3 Representação de Informação Visual	17
2.3.1 Representação Matricial	18
2.3.2 Histogramas	18
2.3.3 Descritores de Cor	19
2.3.4 Descritores de Textura	21
2.3.5 Descritores de Forma	22
2.3.6 Descritores Locais	23
2.3.7 Descritores Locais Binários	27
2.3.8 Descritores Baseado em Vocabulário Visual	28
3 Olhó-passarinho: Módulo do Conteúdo Visual	31
3.1 Recolha dos Dados	31
3.1.1 Descrição dos Dados	31
3.1.2 Filtragem dos Dados	32
3.1.3 Conjunto de Dados Final	33
3.2 Extração, Processamento e Armazenamento da Informação Visual	34
3.2.1 Extração dos Pontos de Interesse e Descritores Locais	35
3.2.2 Criação do Vocabulário Visual	36
3.2.3 Armazenamento da Informação Visual	37

3.3	Matriz de Distâncias entre Imagens	38
4	Olhó-passarinho: Aplicação Web	39
4.1	Arquitetura do Sistema	39
4.2	Informação Espaço-Temporal	40
4.3	Clustering da Informação Visual, Espacial e Temporal	41
4.4	Visualização	41
4.5	Resultados Ilustrativos	43
4.6	Sumário	46
5	Conclusões e Trabalho Futuro	49
5.1	Resumo	49
5.2	Discussão e trabalho futuro	50
A	Tabela da Base de Dados para Seleção de Tweets	53
A	Exemplo objeto JSON de um tweet	55
	Referências	61

Listas de Figuras

2.1	Dendrograma ilustrativo da divisão entre nós do conjunto completo, no topo, ao objeto individual, no fundo [1]	8
2.2	Matriz de confusão de dois objetos com atributos binários	12
2.3	Exemplo ilustrativo da distribuição espacial dos <i>clusters</i> calculada apenas com a consideração da dimensão espacial. <i>Retirada de</i> [2]	14
2.4	Exemplos de resultados de <i>clusters</i> com pesos diferentes para as diferentes dimensões. <i>Retiradas de</i> [2]	15
2.5	<i>Clusters</i> em Portugal: Conteúdo 25% + Espacial 25% + Temporal 25% + Social 25%. <i>Retirada de</i> [2]	16
2.6	Visualização de informação mais detalhada de um <i>cluster</i> incluindo o seu grafo social, e da informação relativa a um determinado tweet. <i>Retirada de</i> [2]	17
2.7	Tweets pertencentes a um <i>cluster</i> para a seguinte distribuição de pesos: Conteúdo 25% + Espacial 25% + Temporal 25% + Social 25%. <i>Retirada de</i> [2]	17
2.8	Imagen em tons cinza e respetivo histograma	19
2.9	Sub-imagem e bloco de imagem. <i>Retirada de</i> [3]	22
2.10	Exemplo de formas de objetos que podem ser descritas eficazmente pelo descriptor baseado em região. <i>Retirada de</i> [4]	23
2.11	Esquematização da obtenção da diferenças de Gaussianos e respetiva. <i>Retirada de</i> [5]	25
2.12	Ilustração do processo de deteção de máximos e mínimos das imagens de Diferença Gaussiana. O píxel candidato está marcado com X e os vizinhos com um circulo. <i>Retirada de</i> [5]	25
2.13	Imagen	26
2.14	(a) Filtros Gaussianos de segunda ordem nas direções yy e xy; (b) aproximação por filtros de caixa; (c) filtros de Haar; As regiões a cinzento têm valor igual a zero. <i>Retirada de</i> [6]	27
2.15	Representação de um <i>cluster</i> de pontos de interesse semelhantes de imagens distintas. <i>Retirada de</i> [7]	29
3.1	Arquitetura do módulo de extração, processamento e armazenamento da informação visual. <i>Adaptada de</i> [8]	34
3.2	Exemplo de projeção de centroides após tarefa de <i>clustering</i> num espaço a duas dimensões.	36
3.3	Estrutura de uma matriz de distâncias. <i>Retirada de</i> [9]	38
4.1	Arquitetura do sistema completo	40
4.2	Distribuição de tweets na dimensão espacial	42
4.3	Exemplo ilustrativo da ferramenta Timeline. <i>Retirada de</i> [10]	42

4.4	Exemplo ilustrativo da visualização da matriz para visualização de nove imagens.	43
4.5	Distribuição dos <i>clusters</i> no mapa calculado exclusivamente através da dimensão espacial	44
4.6	Projeção do <i>Clusters</i> no mapa calculado exclusivamente através da dimensão temporal	44
4.7	Projeção do <i>Clusters</i> no tempo calculado exclusivamente através da dimensão temporal	44
4.8	Projeção do <i>Clusters</i> no mapa calculado exclusivamente através do conteúdo visual	45
4.9	Exemplo da visualização do conteúdo visual de um <i>cluster</i> calculado com 100% do peso para a dimensão das imagens	45
4.10	Projeção dos <i>Clusters</i> no mapa com peso atribuído a cada dimensão de 33.33% .	46
4.11	Projeção dos <i>Clusters</i> no tempo com peso atribuído a cada dimensão de 33.33% .	46
4.12	Exemplo da visualização do conteúdo visual de dois <i>clusters</i> calculados com 33.33% do peso para cada uma das dimensões	47

Lista de Tabelas

3.1	Descrição em números do total de tweets com indicação, nos que contém URL para imagem, do número de tweets por serviço de partilha de imagem	32
3.2	Atributos da tabela da base de dados para filtragem dos tweets.	32
3.3	Elementos contidos no objeto com a informação de um tweet.	33
3.4	Esquema da base de dados SQLite	37
4.1	Distribuição temporal entre os diferentes subconjuntos de dados	41

Abreviaturas e Símbolos

BoW	<i>Bag Of Words</i>
CSD	<i>Contour-based Shape Descriptor</i>
DBSCAN	<i>Density-Based Spatial Clustering of Applications with Noise</i>
DoG	<i>Diference of Gaussian</i>
GLOH	<i>Gradient Location and Orientation Histogram</i>
HMM	<i>Hue-Max-Min-Diff</i>
HOG	<i>Histogram of Oriented Gradients</i>
HSV	<i>Hue Saturation Value</i>
HTD	<i>Homogeneous Texture Descriptor</i>
JSON	<i>JavaScript Object Notation</i>
MS	<i>Maximally Stable</i>
RGB	<i>Red Green Blue</i>
RSD	<i>Region-based Shape Descriptor</i>
SA	<i>Shape Adapted</i>
SIFT	<i>Scale-Invariant Feature Transform</i>
SURF	<i>Speeded Up Robust Features</i>
URL	<i>Uniform Resource Locator</i>

Capítulo 1

Introdução

Neste capítulo é feita uma introdução ao projeto desenvolvido no âmbito da dissertação com a apresentação do seu contexto, a motivação para o seu desenvolvimento, os objetivos a alcançar e a descrição da estrutura deste documento.

1.1 Contexto

As redes sociais são uma excelente fonte de informação sempre em atualização, que fornecem aos investigadores uma vasta quantidade e variedade de dados. Este dados apresentam-se de diferentes formas como texto, imagem ou mesmo vídeos. Esta informação está acessível através de API's disponibilizadas pelos próprios serviços, e pode ser assim utilizada para, por exemplo, realizar análise de sentimentos ou opiniões partilhadas através de texto por utilizadores da rede social Twitter [11, 12], ou mesmo para a descoberta de novas técnicas mais eficazes na pesquisa de imagens no serviço Flickr [13] utilizando anotações inseridas por utilizadores [14].

O Twitter faz parte do grupo de redes sociais existentes que mais informação produz todos os dias, sendo caracterizado como um serviço de *microblogging*, que permite aos utilizadores partilharem mensagens, designadas por tweets, até um máximo de 140 caracteres. Essas mensagens podem conter, para além de texto, imagens ou links para imagens de outros serviços, como por exemplo, o Instagram [15] ou o Twitpic [16]. Ao contrário do que acontece com outras redes sociais como o Facebook [17] e Linkedin [18] que utilizam uma rede de comunicação bi-direcional, o Twitter utiliza uma infraestrutura assimétrica onde existem "*friends*" e "*followers*". Os "*friends*" correspondem às contas das pessoas que o utilizador segue e os "*followers*" às contas das pessoas que o seguem [19].

O TweeProfiles [2], é uma ferramenta que tem como principal objetivo identificar padrões em mensagens escritas, partilhadas na rede social Twitter. Esta ferramenta utiliza técnicas de *data mining*, mais precisamente de *text mining*. A principal característica do TweeProfiles é o facto de utilizar a tarefa de *clustering* para identificar padrões em mensagens partilhadas no Twitter, através do conteúdo das mensagens (o texto) e das dimensões espaço-temporais das mesmas.

1.2 Motivação

Devido ao grande número de utilizadores e de informação partilhada a todo o instante no Twitter, este torna-se um excelente serviço de recolha de dados, proporcionando aos investigadores e empresas uma quantidade e variedade de dados necessários para o desenvolvimento de ferramentas de análise de dados e extração de conhecimento.

As mensagens partilhadas no Twitter sobre a forma de texto têm sido uma das grandes fontes de dados utilizadas por muitas ferramentas como o TweeProfiles [2]. No entanto, apesar de se tratar de uma rede social em que a maioria da informação disponível se encontra em forma de texto, o Twitter também permite a partilha de imagens a partir do seu próprio serviço, ou através de outros serviços como Twitpic ou Instagram. Estas imagens também podem ser utilizadas para a análise e extração de conhecimento, pois o seu conteúdo pode mesmo em muitos casos complementar o texto ou até mesmo, o substituir.

A análise de informação visual é assim um acréscimo importante para o desenvolvimento de ferramentas de extração de conhecimento das redes sociais.

1.3 Objetivos

Esta dissertação tem como principal objetivo a criação de uma extensão para o TweeProfiles através de técnicas de processamento de imagem e *data mining*, que permita a identificação de padrões em imagens partilhadas no serviço de microblogging Twitter, através da identificação de *clusters*.

Será assim necessário realizar a recolha dos dados alojados numa base de dados MongoDB [20] criada através da plataforma Socialbus (anteriormente designada por TwitterEcho [21]). Esta plataforma consiste num projeto open source de desenvolvimento de uma ferramenta para extrair e armazenar tweets de uma determinada comunidade de utilizadores. Foi desenvolvido com o intuito de ajudar os investigadores a terem facilidade de acesso a uma base de dados de redes sociais, na sua maioria. Após recolhidos os dados será necessário o desenvolvimento de um módulo responsável pela recolha das imagens através do *URL* existente nos tweets, do processamento da informação visual de modo a torná-la mais compacta e eficiente, e do armazenamento dessa informação. Por fim, a informação visual deverá ser integrada na ferramenta TweeProfiles, com objetivo de realizar o processo de *Data Mining*, mais especificamente a tarefa de *clustering* e desenvolver a aplicação para visualizar os *clusters* nas diferentes dimensões.

1.4 Estrutura do documento

Este documento está organizado da seguinte forma: o Capítulo 2 descreve conceitos e trabalhos relacionados e apresentada uma pesquisa sobre os vários domínios científicos necessários para o desenvolvimento deste projeto de dissertação. No Capítulo 3 é apresentado o modelo desenvolvido para a extração, processamento e armazenamento da informação visual. Já no Capítulo 4 é descrita

a ferramenta Olhó-passarinho e a integração do módulo desenvolvido para a informação visual com a ferramenta TweeProfiles. Para finalizar é apresentado o Capítulo 5 com um resumo do trabalho desenvolvido e discute o desenvolvimento deste projeto de dissertação com sugestões de trabalho futuro a realizar.

Capítulo 2

Conceitos e Trabalhos Relacionados

Neste capítulo é apresentado o estudo realizado, tendo em vista a aquisição de competências e conhecimentos necessários para o desenvolvimento do projeto, que se focam essencialmente em análise de métodos de *data mining* e em técnicas aplicadas em visão por computador. Em primeiro lugar será exposto conteúdo relativamente a métodos de *clustering* como uma tarefa de *data mining*. Em seguida serão apresentadas formas de representação de imagens. Por fim, são referenciados alguns trabalhos relacionados com o projeto a desenvolver nesta dissertação.

2.1 Clustering

Extração de conhecimento em base de dados ou *data mining* é um processo de exploração de grandes quantidades de dados que procura encontrar padrões "interessantes" [9]. Trata-te assim de uma fusão de estatística aplicada, sistemas de lógica, inteligência artificial, *machine learning* e gestão de base de dados [22]. Este processo é caracterizado por várias tarefas possíveis de ser aplicadas, dependendo do problema abordado, tais como [23]:

- Deteção de anomalias (outliers/ alterações/ desvios) - Identifica registo de dados incomuns, podendo ser erros nos dados ou objetos interessantes que apresentam comportamento diferente dos restantes;
- Regras de associação - Procura relações entre variáveis cujos valores ocorram frequentemente em conjunto;
- Classificação - É a tarefa de identificar padrões nas características de objetos que distingam entre os que pertencem a grupos pré-definidos diferentes, sendo essencialmente utilizada em tarefas de previsão;
- Regressão - Tenta encontrar padrões nas características de objetos que as relacionam com uma variável numérica;
- Resumo - Tenta procurar uma representação mais compacta do conjunto de dados, que pode incluir visualização e descrição através de um relatório,

- *Clustering* - Tarefa de descobrir grupos em que os dados apresentam de alguma forma semelhanças, sem o uso de estruturas previamente conhecidas

Este projeto terá como uma das principais tarefas a realização *clustering* sobre dados recolhidos e tratados de fotografias partilhadas no Twitter. Pode-se definir *clustering* como "um processo de agrupamento de um conjunto de objetos de dados em vários grupos ou *clusters*, de modo que os objetos dentro de um *cluster* apresentem alta similaridade, mas que sejam muito diferentes de objetos de outros *clusters*. Diferenças e semelhanças são avaliados com base nos valores de atributos que descrevem os objetos e muitas vezes envolvem medidas de distância" [9]. A realização de *clustering* é assim uma escolha lógica para a extração de padrões em dados e para o agrupamento de *tweets* pela sua semelhança em conteúdo, neste caso as imagens, e com a integração de outras dimensões, como o tempo e espaço.

O *clustering* faz parte de um conjunto de técnicas aplicadas na aprendizagem não supervisionada. Enquanto que na aprendizagem supervisionada existe um conjunto de dados previamente analisados e rotulados que são usados para treinar um modelo capaz de encontrar relações entre os atributos desses dados com os rótulos respetivos, na aprendizagem não supervisionada, não são utilizados conjuntos de dados previamente rotulados. Assim, o processo de descoberta de padrões nos dados apenas tenta organizar as instâncias em grupos semelhantes [24].

Nesta secção serão apresentadas as principais características e técnicas para aplicação da tarefa de *clustering*, tais como, *clustering* por partição, *clustering* hierárquico, *clustering* baseado em densidade, *clustering* baseado em grelhas, funções de distância para o cálculo da distância entre objetos.

2.1.1 Clustering por Partição

A utilização de métodos baseados em partições é a forma mais simples e elementar de realizar análise por *clustering*. Nestes métodos em que um conjunto de objetos é distribuído em vários grupos ou *clusters* mais pequenos. É assumido que o número de *clusters* é conhecido antes da realização da tarefa, sendo esse valor tomado como o ponto de partida para aplicação de métodos baseados em partição [9].

O algoritmo *k-means* é o algoritmo de *clustering* por partição mais utilizado devido à sua simplicidade e eficiência [24]. Inicialmente é necessário que o utilizador indique o valor de *k* e o algoritmo irá iterativamente dividir o conjunto de objetos em *k clusters* diferentes, baseado em funções de distância [24] que são apresentadas na secção 2.1.5.

Cada *cluster* apresenta um objeto que representa um grupo, sendo o valor médio de todos os objetos (instâncias) pertencentes ao *cluster* e é normalmente designado por centroide. Este centroide é recalculado de forma iterativa até que seja atingido o critério de paragem. A convergência ou critério de paragem pode ser um dos seguintes enumerados:

1. Não ocorre (ou ocorre um valor mínimo) de alterações dos objetos para diferentes *clusters*.
2. Não ocorre (ou ocorre um valor mínimo) de alterações dos centroides.

3. Diminuição mínima da **soma do erro quadrático** (SEQ),

$$SEQ = \sum_{j=1}^k \sum_{x \in C_j} dist(x, m_j)^2, \quad (2.1)$$

em que k é o número de *clusters* pretendidos, C_j é i-ésimo *cluster*, m_j é o centroide do *cluster* C_j e $dist(x, m_j)$ é a distância entre uma instância x e o centroide m_j .

Assim, "o algoritmo *k-means* pode ser usado em qualquer aplicação com um conjunto de dados onde a média pode ser definida e calculada" [24]. No espaço, o centroide de um *cluster* é calculado da seguinte forma:

$$m_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i, \quad (2.2)$$

onde $|C_j|$ é o número de pontos (instâncias) no *cluster* C_j . A distância de um ponto x_i a um centroide m_j através de uma função de distância. A mais utilizada para este fim é a função distância Euclidiana, apresentada mais à frente na Secção 2.1.5. A equação 2.3 representada da seguinte forma:

$$dist(x_i, m_j) = \|x_i - m_j\| = \sqrt{(x_{i1} - m_{j1})^2 + (x_{i2} - m_{j2})^2 + \dots + (x_{ir} - m_{jr})^2} \quad (2.3)$$

O pseudo-código deste algoritmo é apresentado no algoritmo 1.

Algoritmo 1 K-Means

```

1: procedure K-MEANS( $k$ : clusters,  $D$ : conjunto de dados)
2:   escolher  $k$  objetos de  $D$  como centroides dos clusters iniciais;
3:   repeat
4:     (re) atribuir cada objeto ao cluster ao qual o objeto tem menor distância;
5:     atualizar o centroide do cluster;
6:   until clusters sem alterações;
7:   return conjunto de  $k$  clusters;
8: end procedure
```

2.1.2 Clustering Hierárquico

O *clustering* hierárquico é outra abordagem importante na tarefa de *clustering*. Os *clusters* são criados sobre a forma de uma sequência em árvore (dendrograma). Os objetos (instâncias) encontram-se no fundo do diagrama, enquanto que o conjunto de todos os objetos encontra-se no topo do diagrama. Cada nó que se encontra no interior do diagrama possui nós filhos, sendo que cada nó representa um *cluster*. Assim designam-se por *clusters* irmãos, aqueles que derivam de um mesmo *cluster*, isto é, do nó pai [24]. A Figura 2.1 exemplifica a representação de um dendrograma onde o topo é representado o conjunto de todas as letras e o fim de cada letra individual.

Existem dois tipos principais de métodos de *clustering* hierárquico, sendo eles [24] :

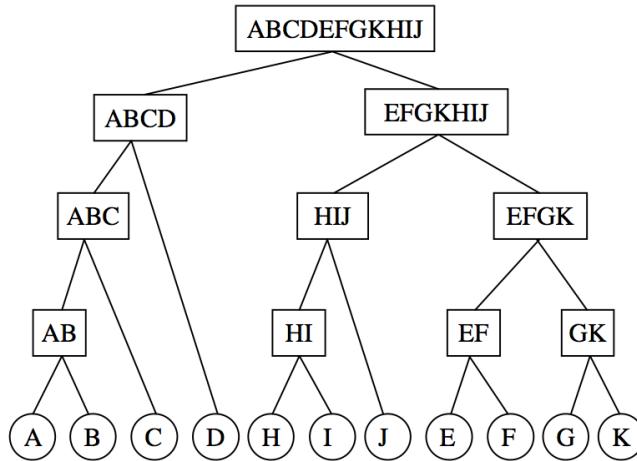


Figura 2.1: Dendrograma ilustrativo da divisão entre nós do conjunto completo, no topo, ao objeto individual, no fundo [1]

Clustering por aglomeração: O dendrograma é construído do nível mais baixo até ao mais alto, juntando sucessivamente e iterativamente os *clusters* com maiores semelhanças até existir um único *cluster* com todo o conjunto dos dados.

Clustering por divisão: O dendrograma é construído do nível mais alto até ao nível mais baixo, onde o processo tem início com um único *cluster* que possui todos os objetos, sendo dividido sucessivamente em *clusters* mais pequenos, até que estes sejam constituídos apenas por um único objeto.

Ao contrário do algoritmo *k-means*, que apenas calcula a distância entre os centroides de cada grupo ou *cluster*, no clustering hierárquico podem ser usados os vários métodos apresentados em seguida para determinar a distância entre dois *clusters* [24]:

Método Single-Link: Neste método, a distância entre dois *clusters* é determinada pela distância entre os dois objetos mais próximos (vizinhos mais próximo) pertencentes a *clusters* diferentes.

Método Complete-Link: Neste método, a distância entre dois *clusters* é determinada pela maior distância entre dois objetos (vizinhos mais distante).

Método Average-Link: Este método tenta manter um compromisso entre a sensibilidade a *outliers* do método *Complete-Link* e a sensibilidade do método *Single-Link* ao ruído existente nos dados. Para isso, é determinada a distância entre dois *clusters* através da distância média entre todos os pares de objetos nos dois *clusters*.

Método Ward: Este método, tenta minimizar a variância dentro de um *clusters*. Cada *cluster* começa com um único ponto e recursivamente os *cluster* vão sendo unidos. Este método procura que os *clusters* pares a serem unidos para formarem um novo, sendo que estes

devem-se encontrar a uma distância mínima e garantir um aumento mínimo da variância total do *cluster* criado.

Assim conclui-se que o *clustering* hierárquico apresenta-se para determinados domínios, como bastante intuitivo, mas a interpretação dos resultados pode ser por vezes subjetiva. Outra característica interessante é o facto de, ao contrário do *clustering* por partição, no *clustering* hierárquico não ser necessário especificar logo à partida o número de *clusters*.

2.1.3 Clustering Baseado em Densidade

Os métodos de *clustering* por partição ou hierárquico estão preparados para encontrar *clusters* que apresentam formas geométricas circulares, sendo ineficiente quando as formas destes grupos são, por exemplo, elípticas. Assim, para descobrir *clusters* com formas arbitrárias, podem ser usados métodos baseados na densidade dos objetos [9].

Um dos algoritmos mais conhecidos que utiliza este tipo de método é o DBSCAN (*Density-Based Spatial Clustering of Applications with Noise*) que é capaz de encontrar *clusters* através da análise da densidade e proximidade dos objetos pertencentes a um conjunto de dados. Para esta análise é necessário previamente definir o raio da vizinhança considerada para cada objeto. Esse parâmetro é designado por ϵ e terá de ser necessariamente maior que 0. Assim, a ϵ -vizinhança de um objeto x é o espaço dentro de um raio com valor ϵ , centrado em x [9]. Já para determinar a densidade de uma vizinhança, é utilizado o parâmetro *MinPts* também previamente definido. Este parâmetro especifica o número mínimo de objetos vizinhos que um objeto necessita ter em seu redor, para ser considerado como centroide. Os passos necessários para a implementação deste método são apresentados no Algoritmo 2.

Apesar de no *clustering* baseado em densidade ser necessário definir um raio e o número mínimo de objetos para a formação de um *cluster*, este apresenta como principal vantagem o fato de não é necessário uma prévia definição do número de *clusters*, sendo apresentados os que forem encontrados consoante os dados que possui e os parâmetros definidos.

2.1.4 Clustering Baseado em Grelhas

Os métodos de *clustering* discutidos até agora, apresentam algoritmos que se adaptam a distribuição dos dados no espaço. Em alternativa, o *clustering* baseado em grelhas é orientado ao espaço, na medida em que divide o espaço em células independentemente da distribuição dos objetos de entrada. Este quantifica o espaço num número finito de células, que formam uma estrutura de grelhas sobre a qual são executadas as operações de *clustering*. Este método apresenta como principal vantagem o tempo baixo de processamento, que normalmente é independente da quantidade de dados. No entanto, depende do número de células em cada uma das dimensões no espaço [9].

O algoritmo STING (*Statistical Information Grid*) [25] é um dos algoritmos utilizados para *clustering* baseado em grelhas. Este divide o espaço em células retangulares, correspondente a diferentes resoluções que formam uma estrutura hierárquica. Cada célula pertencente a um nível

Algoritmo 2 DBSCAN

```

1: procedure DBSCAN(MinPts: limiar da vizinhança, D: conjunto de dados,  $\epsilon$  : raio )
2:   Marcar todos os objetos como não selecionados;
3:   repeat
4:     escolher um objeto  $x$  não selecionado;
5:     atualizar objeto  $x$  como selecionado;
6:     if a  $\epsilon$ -vizinhança de  $x$  tem pelo menos MinPts objetos then
7:       criar um novo cluster  $C$  e adicionar objeto  $x$  ao cluster  $C$ ;
8:       Seja  $N$  o conjunto de objetos na  $\epsilon$ -vizinhança de  $x$ ;
9:       for cada ponto  $x'$  em  $N$  do
10:        if  $x'$  não selecionado then
11:          Marcar  $x'$  como selecionado
12:          if a  $\epsilon$ -vizinhança de  $x'$  tem pelo menos MinPts then
13:            adicionar ponto a  $N$ 
14:          end if
15:        end if
16:        if  $x'$  não pertence a nenhum cluster then
17:          adicionar  $x'$  a  $C$ 
18:        end if
19:      end for
20:      output  $C$ ;
21:    else marcar  $x$  como outlier
22:    end if
23:  until todos os objetos selecionados;
24: end procedure

```

mais perto do topo, é dividida para formar células de menor dimensão no nível seguinte. Assim, sabe-se que o nível mais baixo apresenta uma maior resolução. Isto permite que os *clusters* sejam encontrados recorrendo a uma pesquisa de cima para baixo (*clustering* por divisão, como explicado na secção 2.1.2), passando por cada nível até atingir o mais baixo, retornando no fim as células mais relevantes para a consulta especificada. A informação estatística de cada célula é calculada e armazenada para o processamento de consultas futuras. Também é necessário ter em atenção que apenas é considerado para este algoritmo um espaço bidimensional.

Um outro algoritmo com características semelhantes é o CLIQUE [26], que "identifica *clusters* densos em sub-espacos de máxima dimensão". Isto é, são detetados todos os *clusters* em todos sub-espacos existentes e em que um ponto pode pertencer a vários *clusters* em sub-espacos diferentes.

2.1.5 Funções de Distância

As funções de distância têm um papel fulcral em todos os algoritmos de *Clustering*. Existem inúmeras funções de distância usadas para diferentes tipos de atributos (ou variáveis) [24]. Em seguida serão apresentadas diferentes funções de distância para diferentes atributos: numéricos, binários e nominal. Também serão apresentadas funções de distância utilizadas para as dimensões temporal e de conteúdo.

2.1.5.1 Atributos Numéricos

As funções de distância mais utilizadas para variáveis numéricas são a **Distância Euclidiana** e **Distância Manhattan**. É utilizado $dist(x_i, x_j)$ para representar a distância entre duas instâncias. Ambas as funções referidas anteriormente são casos especiais da função mais geral chamada **Distância Minkowski** [24]:

$$dist(x_i, x_j) = (|x_{i1} - x_{j1}|^h + |x_{i2} - x_{j2}|^h + \dots + |x_{ir} - x_{jr}|^h)^{\frac{1}{h}}, \quad (2.4)$$

onde h é um inteiro positivo. Se $h=2$, temos a **Distância Euclidiana**,

$$dist(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{ir} - x_{jr})^2}, \quad (2.5)$$

Se $h=1$, temos a **Distância City-block (Manhattan)**,

$$dist(x_i, x_j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ir} - x_{jr}|. \quad (2.6)$$

Também, não menos importantes, são outras funções distância apresentadas em seguida:

Distância Euclidiana Ponderada : A ponderação é atribuída através de pesos dados pela importância que cada atributo representa relativamente a outros atributos, representada por w_i :

$$dist(x_i, x_j) = \sqrt{w_1(x_{i1} - x_{j1})^2 + w_2(x_{i2} - x_{j2})^2 + \dots + w_r(x_{ir} - x_{jr})^2}. \quad (2.7)$$

Distância Euclidiana Quadrática : Trata-se de uma alteração da função **Distância Euclidiana**, elevando a mesma ao quadrado, o que faz com que seja progressivamente atribuído peso maior a pontos dos dados que estejam mais afastados:

$$dist(x_i, x_j) = (x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{ir} - x_{jr})^2. \quad (2.8)$$

Distância Chebyshev : Utilizada para casos em que há necessidade de definir dois pontos dos dados como diferentes, caso sejam diferentes em qualquer dimensão:

$$dist(x_i, x_j) = \max(|x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \dots + |x_{ir} - x_{jr}|). \quad (2.9)$$

2.1.5.2 Atributos Binários e Nominais

As funções apresentadas anteriormente apenas podem ser utilizadas com atributos do tipo numérico. Assim serão necessárias funções de distância específicas para atributos do tipo binário e nominal.

Uma variável binária é aquela que apenas pode assumir dois estados ou valores, sendo normalmente representado pelo valor 0 e 1. Mas estes estados não apresentam um ordem definida. Por

exemplo, no caso de uma lâmpada, esta pode assumir apenas dois estados, ligado ou desligado, ou o género de uma pessoa, masculino ou feminino. Estes exemplos apresentam dois valores diferentes mas que não possuem qualquer ordem.

Os atributos binários podem ser divididos em dois tipos diferentes, os simétricos e os assimétricos, sendo em seguida apresentadas as funções de distância para ambos os casos [24].

Na Figura 2.2 é representada uma matriz de confusão para uma melhor compreensão do cálculo da distância entre objetos com atributos binários.

		Objeto j		
		x = 1	x = 0	
Objeto i	x = 1	a	b	$a+b+c+d = \text{número de variáveis}$
	x = 0	c	d	

Figura 2.2: Matriz de confusão de dois objetos com atributos binários

Atributos simétricos: Um atributo é simétrico quando ambos os estados (0 ou 1) têm a mesma importância e o mesmo peso, tal como ocorre no exemplo dado anteriormente com o atributo género (masculino e feminino). Para este caso, a função distância mais utilizada é designada por *simple matching distance*, que corresponde à proporção de incompatibilidade ou desacordo:

$$dist(x_i, x_j) = \frac{b + c}{a + b + c + d} \quad (2.10)$$

Atributos assimétricos: Um atributo é assimétrico se um dos estados apresenta maior importância do que o outro. Normalmente o estado mais valioso é o que ocorre com menor frequência. No nosso caso iremos considerar o estado 1 como o mais valioso. Assim, a função distância mais frequentemente utilizada para atributos assimétricos é a *Jaccard distance*:

$$dist(x_i, x_j) = \frac{b + c}{a + b + c} \quad (2.11)$$

No caso de atributos nominais com mais de dois estados ou valores, a função de distância mais utilizada, é baseada na *simple matching distance*. Dados dois objetos i e j , r corresponde ao número total de atributos e q ao número de valores que são mutuamente correspondidos entre os objetos x_i e x_j :

$$dist(x_i, x_j) = \frac{r + q}{r} \quad (2.12)$$

2.1.5.3 Dimensão Temporal

O tempo é representado apenas por uma dimensão, sendo que para calcular a distância, por exemplo, entre dois tweets t_i e t_j , apenas é necessário calcular a diferença dos tempos entre os mesmos. Supondo que os valores dos tempos são respetivamente Δ_i e Δ_j , o intervalo de tempo pode ser definido pela seguinte equação:

$$dist^T(t_i, t_j) = |\Delta_i - \Delta_j| \quad (2.13)$$

2.1.5.4 Dimensão Espacial

Ao contrário da dimensão temporal, a dimensão espacial apresenta mais do que uma dimensão, latitude e longitude. Estas apresentam-se sobre a forma numérica, sendo possível o cálculo da distância entre dois objetos através de funções de distância para atributos numéricos como referido anteriormente (2.1.5.1). Assim, para o cálculo entre pontos distribuídos num espaço poder-se-á recorrer à função Minkowski (equação 2.4), à função Euclidiana (equação 2.5), à função Manhattan (equação 2.6) ou mesmo à função de Chebychev (equação 2.9), sendo que no caso mais específico de uma distribuição espacial geográfica, em que os pontos possuem latitude e longitude, é considerada mais apropriada a utilização da função distância Haversine pois esta toma em consideração a forma esférica da Terra [27]. Assim, obtendo um par de objetos x_i e x_j distanciados geograficamente, são consideradas a latitude ϕ_{x_i} e ϕ_{x_j} e a longitude λ_{x_i} e λ_{x_j} para determinar a distância entre os objetos através da equação 2.14

$$dist^{Sp}(x_i, x_j) = 2R \sin^{-1} \left(\left[\sin^2\left(\frac{\phi_{x_i} - \phi_{x_j}}{2}\right) + \cos \phi_{x_i} \cos \phi_{x_j} \sin^2\left(\frac{\lambda_{x_i} - \lambda_{x_j}}{2}\right) \right]^{0.5} \right) \quad (2.14)$$

onde R representa o raio da Terra e que determina as unidades do resultado retornado pela função, sendo comum a utilização das unidades no sistema internacional (SI), o metro, podendo também ser representado em quilómetros devido ao fator de escala.

2.2 TweeProfiles

Esta dissertação pretende dar continuidade e um trabalho designado por TweeProfiles [2]. O TweeProfiles é uma ferramenta que identifica perfis de tweets envolvendo múltiplos tipos de informação: espacial, temporal, textual e social. Esta secção faz uma breve introdução e descrição sobre esta ferramenta.

2.2.1 Descrição e objetivos

O TweeProfiles aborda o problema de identificar perfis de tweets envolvendo múltiplos tipos de informação: espacial, temporal, social e de conteúdo. A informação espacial, trata-se da informação de localização das mensagens de texto partilhadas no Twitter; a temporal é relativa à data

de publicação do tweet; a social às ligações entre os utilizadores, fim o conteúdo que no trabalho original é relativo apenas ao texto contido em cada tweet.

Os objetivos do TweeProfiles foram o desenvolvimento de uma metodologia de *data mining* que identificasse perfis de tweets que combinem de forma flexível as várias dimensões consideradas, a criação de uma ferramenta de visualização para representar os resultados obtidos e a sua aplicação a um caso de estudo na *twitosfera* portuguesa.

A ferramenta de visualização está desenhada para uma utilização dinâmica e intuitiva, direcionada para a representação dos perfis de uma forma comprehensível e interativa. Esta apresenta vários widgets capazes de representar os padrões obtidos. O caso de estudo que o TweeProfiles usa dados georeferenciados do Socialbus (antes designado como TwitterEcho [21]). No entanto, esta ferramenta é adequada para tratar quaisquer mensagens georeferenciadas provenientes do Twitter.

2.2.2 Resultados ilustrativos

Através do TweeProfiles podemos visualizar *clusters* nas suas diversas dimensões. Esta ferramenta apresenta três secções distintas para visualização e inclui ainda controlos para navegação e seleção de determinados parâmetros. Uma das secções é constituída por um mapa onde é possível visualizar geograficamente os *clusters*, como podemos ver na Figura 2.3, em que são representados por círculo. Outra das secções é um gráfico que apresenta a distribuição temporal dos *clusters*, onde é possível visualizar a data de início e a data de fim de um determinado *cluster*, isto é, visualizar o intervalo de tempo entre a data do primeiro tweet partilhado e o último, contidos no mesmo *cluster*. Por fim, é-nos apresentada uma secção onde é possível visualizar informação mais espe-



Figura 2.3: Exemplo ilustrativo da distribuição espacial dos *clusters* calculada apenas com a consideração da dimensão espacial. *Retirada de* [2]

cífica sobre os *clusters*, incluindo a informação relativa às dimensões de conteúdo e social com a representação das palavras mais utilizadas e as ligações entre utilizadores respetivamente nesse *clusters*. Os controlos permitem selecionar um dos três intervalos de tempo existentes e os pesos

para cada uma das dimensões, que podem assumir valores entre 0% e 100% com incrementos de 25%. Cada dimensão pode assumir como peso assim um dos seguintes valores percentuais: {0, 25, 50, 75, 100}, sendo que a soma dos pesos das diferentes dimensões deve ser igual a 1.

Em seguida são apresentados alguns resultados ilustrativos obtidos através da ferramenta TweeProfiles, e que demonstram o seu funcionamento global. Este resultados serão baseados apenas nos apresentados por Tiago Cunha [2], sendo ilustrados resultados de *clusters* em Portugal para diferentes combinações de importância das dimensões possíveis.

Na Figura 2.4 são apresentadas quatro combinações entre a dimensão do conteúdo e as restantes, exceto no primeiro caso (Figura 2.4a) onde podemos ver um *cluster* em Portugal apenas tendo em consideração a dimensão do conteúdo. Já no caso da Figura 2.4b o peso é distribuído da mesma forma entre o conteúdo e a dimensão espacial, resultando em dois *clusters*. Na Figura 2.4c, tal como no caso anterior, é distribuído o peso de igual forma entre o conteúdo, e neste caso a dimensão temporal. O resultado obtidos no caso de Portugal, assemelha-se ao do primeiro caso da Figura 2.4a. Por fim, temos o caso da Figura 2.4d onde o peso é de 50% para o conteúdo e 50% para a dimensão social, tendo resultado para o caso de Portugal, dois *clusters*.

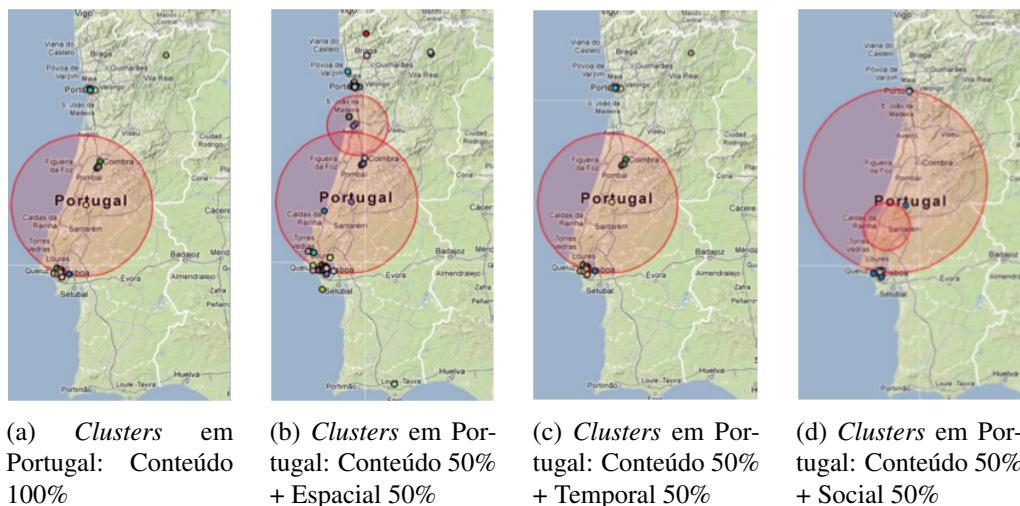


Figura 2.4: Exemplos de resultados de *clusters* com pesos diferentes para as diferentes dimensões. Retiradas de [2]

Outra das combinações apresentadas, foi a distribuição igual por todas as dimensões. Este caso é apresentado na Figura 2.5 onde todas as dimensões possuem o peso de 25% cada. Neste caso foram obtidos três *clusters* em Portugal.

Por fim, na Figura 2.6 é possível visualizar a secção que apresenta ao utilizador informação sobre um *cluster* selecionado, o conteúdo típico desse *cluster*, ou seja, as palavras mais frequentes, e o grafo com as ligações sociais. É possível ainda visualizar um determinado tweet com mais detalhe, como o nome do utilizador, informação temporal e espacial.



Figura 2.5: *Clusters* em Portugal: Conteúdo 25% + Espacial 25% + Temporal 25% + Social 25%. Retirada de [2]

2.2.3 Prós e contras

Como vimos na Secção 2.2.2, a ferramenta TweeProfiles permite uma grande variedade de combinações que produzem resultados que podem variar no número e posição geográfica dos *clusters* consoante a influência e peso das dimensões em consideração.

Este projeto de dissertação não tem como objetivo melhorar o TweeProfiles, mas sim apresentar uma abordagem alternativa de visualização de análise de dados em redes sociais utilizando conteúdo diferente. Por este motivo, nesta secção somente abordamos os prós e contras relativamente à abordagem das dimensões utilizadas.

Uma das vantagens da utilização do texto como dimensão de conteúdo, é o facto de no Twitter este se apresentar como a maior fonte de informação, pois os tweets na sua maioria apresentam texto. Mesmo adicionando a dimensão espacial, que implica a existência de uma referência geográfica contida na informação dos tweets, o número de tweets disponíveis continua a ser suficiente para uma análise dos dados [2]. Apesar disso, é possível verificar que nos resultados apresentados existe muito conteúdo em forma de texto que apresenta informação muitas vezes com pouca relevância. Um exemplo é o caso da partilha da localização através de outros serviços onde é utilizado um texto pré-definido pelo serviço utilizado, indicando apenas o local onde se encontra o utilizador, como podemos ver pela Figura 2.7.

O TweeProfiles revela ainda que a dimensão espacial, quando tida em consideração, tem um influência interessante nos resultados apresentados no mapa. Já a dimensão temporal não revelou apresentar uma grande influência (Figura 2.4c), mas permite um controlo e visualização de informação através do gráfico temporal. No caso da dimensão social, verifica-se que esta apresenta

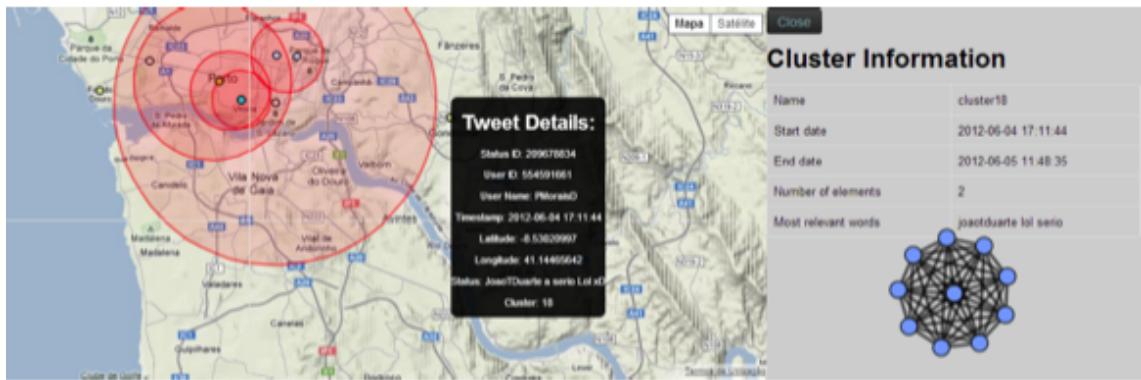


Figura 2.6: Visualização de informação mais detalhada de um *cluster* incluindo o seu grafo social, e da informação relativa a um determinado tweet. *Retirada de [2]*

Cluster	Tweet ID	User Name	Timestamp	Latitude	Longitude	Status
3	208529373	Madptron	01/06/2012 13:04	-9.20568466	38.66111817	I'm at Faculdade de Ciências e Tecnologia - Universidade Nova de Lisboa (Almada, Portugal) http://t.co/mDq2VH8y
3	209562126	Madptron	04/06/2012 09:27	-9.20568466	38.66111817	I'm at Faculdade de Ciências e Tecnologia - Universidade Nova de Lisboa (Almada, Portugal) http://t.co/noHVgTY
3	209982566	pedro_jose	05/06/2012 13:18	-9.20568466	38.66111817	I'm at Faculdade de Ciências e Tecnologia - Universidade Nova de Lisboa (Almada, Portugal) http://t.co/y8T8q0VI
3	210022881	diogoreis32	05/06/2012 15:58	-9.20568466	38.66111817	I'm at Faculdade de Ciências e Tecnologia - Universidade Nova de Lisboa (Almada, Portugal) http://t.co/SZQVE4fw

Figura 2.7: Tweets pertencentes a um *cluster* para a seguinte distribuição de pesos: Conteúdo 25% + Espacial 25% + Temporal 25% + Social 25%. *Retirada de [2]*

alguma influência na divisão dos *clusters* mas a visualização da sua informação através do grafo, apresenta-se pobre e de difícil compreensão para o utilizador.

Conforme dito anteriormente (Capítulo 1), o objetivo deste projeto é estender o TweeProfiles para identificar padrões também nas imagens publicadas nos tweets, para além das dimensões já analisadas. Assim, na próxima secção, são apresentadas abordagens para a representação de informação visual, que servirá de base a este projeto.

2.3 Representação de Informação Visual

Na secção 2.1 foi apresentado o conceito e características da tarefa de *clustering* de uma forma geral. Nesta secção serão expostas formas de representar imagens como dados.

Como o objetivo desta dissertação passa por a realização da tarefa de *clustering*, utilizando como dados as imagens partilhadas no serviço Twitter, é necessário utilizar formas eficientes para descrever cada imagem de uma forma compacta e de forma a que seja descrito o conteúdo geral das imagens. É importante salientar que a tarefa de *clustering* em imagens é muitas vezes associada à técnica de segmentação em imagem [28], em que se pretende distinguir objetos individuais, não sendo este o objetivo deste projeto de dissertação. Pretende-se sim que para a tarefa de *clustering* o objeto representativo de uma imagem descreva esta pelo seu conjunto, tal como referido anteriormente.

2.3.1 Representação Matricial

Uma imagem pode ser vista como um objeto (ou instância), sendo computacionalmente representada como uma matriz (um vetor bi-dimensional) de píxeis. A matriz de píxeis descreve assim a imagem como $N \times M$ m -bit píxeis, onde N corresponde ao número de pontos ao longo do eixo horizontal, M o número de pontos ao longo do eixo vertical e m o número de bits por píxel que controla os níveis de brilho. Com m bits temos uma gama de valores para o brilho de 2^m , que varia entre 0 e $2^m - 1$. Assim se o valor de m for 8, os valores de brilho de cada píxel de uma imagem podem variar entre 0 e 255, que normalmente correspondem ao preto e branco respetivamente, sendo que os valores intermédios correspondem aos tons de cinza [29].

No caso de imagens a cores, o princípio é idêntico, no entanto ao invés de se usar apenas um plano, as imagens a cores são representadas por 3 componentes de intensidade, designado por modelo RGB (*Red Green Blue*), a que corresponde respetivamente as cores vermelho (Red), verde (Green) e azul (Blue). Para além deste esquema de cores, também existem outros como o CMYK composto pelas componentes de cor, azul turquesa, magenta, amarelo e preto. Para qualquer esquema de cores, existem 2 métodos principais para representar a cor do píxel. No primeiro método é utilizado um valor inteiro para cada píxel, sendo esse valor como um índice para uma tabela, também conhecida como paleta da imagem, com a correspondência à intensidade de cada componente de cor. Este método tem como vantagem o facto de ser eficiente na utilização da memória, pois apenas é guardado um plano da imagem (os índices) e a paleta (tabela). Por outro lado, tem como desvantagem o facto de normalmente ser usado um conjunto reduzido de cores o que provoca uma redução da qualidade da imagem. Já o segundo método consiste na utilização de vários planos da imagem para armazenar a componente de cor de cada píxel. Este representa a imagem com mais precisão pois considera muito mais cores. O formato mais usual é 8 bits para cada uma das 3 componentes, no caso do RGB. Assim, são utilizados 24 bits para representar a cor de cada píxel, o que permite que uma imagem possa conter mais de 16 milhões de cores simultaneamente. Como era de esperar, isto envolve um custo grande na utilização de memória [29].

Em suma, a representação matricial é uma forma fácil de representar uma imagem, mas ao contrário dos objetivos desta dissertação, não consegue fazê-lo de uma forma compacta, sendo necessária a existência de grandes quantidades de memória devido à sua dimensionalidade. Para além disso, este apresenta uma grande sensibilidade ao ruído por dar a mesma relevância a todos os píxeis da imagem. Assim, esta não se apresenta como uma boa solução para a extração da informação das imagens.

2.3.2 Histogramas

Outras das formas de representar a informação de uma imagem é através de um histograma. Um histograma de uma imagem apresenta a frequência de ocorrência de níveis individuais de brilho, representado através um gráfico que mostra o número de píxeis da imagens com um determinado nível de brilho. No caso de píxeis representados por 8-bits, o brilho vai variar de 0 (preto)

até 255 (branco) [29]. Também pode ser apresentada informação de cor sobre uma imagem através de um histograma, sendo para isso necessário apresentar 3 histogramas diferenciados, um para cada componente de cor, no caso do esquema RGB. A figura 2.8 apresenta um exemplo de um histograma de uma imagem com tons cinza, onde são representados o número de pixels para cada nível diferente de cinzento.

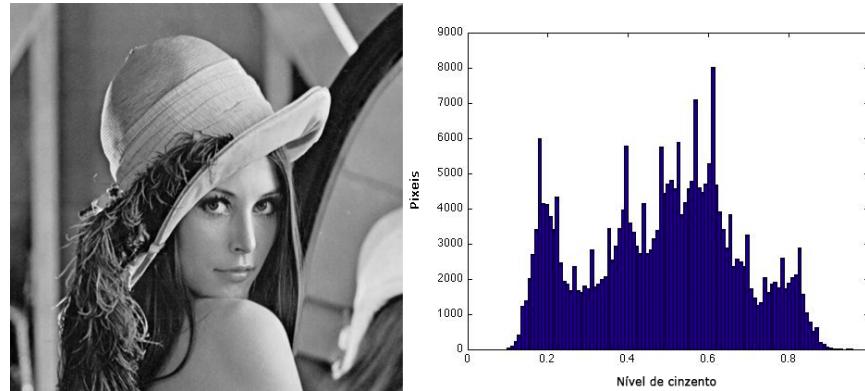


Figura 2.8: Imagem em tons cinza e respetivo histograma

2.3.3 Descritores de Cor

A cor apresenta-se como um importante atributo da imagem para o olho humano e processamento por computador. Nesta secção são apresentados vários descritores de cor considerados pelo MPEG-7 [30, 31, 32, 33], e utilizados para extração de informação e reconhecimento de similaridade em imagens. cor

2.3.3.1 Espaços de Cor

Existe uma vasta seleção de espaços de cores, tais como, RGB, YCbCr, HSV, HMMS, Monocromático e Matriz linear de transformação com referência a RGB.

O espaço de cor RGB é um dos modelos referidos mais utilizados, que apresenta três componentes distintas, vermelho, verde e azul, tal como foi referido na Secção 2.3.1. Neste modelo é utilizada a combinação das 3 cores primárias para representar as diferentes cores. O modelo YCbCr provém do padrão MPEG-1/2/4 [33] e é definido pela transformação linear do espaço de cor RGB como demonstrado na equação 2.15:

$$\begin{aligned} Y &= 0.299 \times R + 0.587 \times G + 0.114 \times B \\ Cb &= -0.169 \times R - 0.331 \times G + 0.500 \times B \\ Cr &= 0.500 \times R - 0.419 \times G - 0.081 \times B \end{aligned} \quad (2.15)$$

No caso do espaço de cor Monocromático, é usado apenas a componente Y do modelo YCbCr.

O espaço de cor HSV (*Hue Saturation Value*) apresenta uma especificação mais complexa, tendo sido desenvolvido para fornecer uma representação mais intuitiva e para se aproximar mais do sistema visual humano. A transformação do modelo RGB para o HSV não é linear, mas é reversível [30]. Uma das componentes é a matiz (H - *Hue*), que representa a componente de cor espectral dominante na sua forma mais pura, como o verde, amarelo, azul e vermelho. Ao ser adicionado branco à cor, esta sofre uma alteração e torna-se menos saturada. A saturação (S - *Saturation*) é precisamente outra das componentes deste modelo. Por fim, o valor (V - *Value*) corresponde ao brilho de cor.

O espaço de cor HMMD (*Hue-Max-Min-Diff*) [30, 33] é mais recente. É caracterizado pela componente matiz, tal como o modelo HSV, pelo *max* e *min*, que são respetivamente o máximo e mínimo entre os valores R, G e B. Para descrever este modelo, também é utilizada a componente *Diff*, que corresponde à diferença entre o *max* e *min*. Para representar este espaço de cor, apenas são necessárias três das quatro componentes referidas anteriormente, como por exemplo, *Hue*, *Max*, *Min* ou *Hue*, *Diff*, *Sum*, onde *Sum* pode ser definida pela equação 2.16.

$$Sum = \frac{Max + Min}{2} \quad (2.16)$$

2.3.3.2 Cor Dominante

O descritor de cor dominante fornece uma representação compacta das cores de uma imagem ou da região da imagem. Este apresenta a distribuição das cores mais representativas na imagem. Ao contrário do descritor de cor por histograma, na especificação do descritor de cor dominante, as cores mais representativas são calculadas a partir de cada imagem, em vez de ser fixado no espaço de cor, permitindo assim, uma representação das cores presentes numa região de interesse mais exata e compacta.

O descritor de cor dominante pode ser definido como:

$$F = c_i, p_i, v_i, s, (i = 1, 2, \dots, N)$$

onde N é o número de cores dominantes. Cada valor c_i da cor dominante é um vetor de valores das componentes do espaço de cor correspondente (por exemplo, um vetor de 3 dimensões no espaço de cor RGB). O valor p_i é a fração de pixels na imagem ou região da imagem (normalizado para um valor entre 0 e 1) que corresponde à cor c_i , sendo $\sum_i p_i = 1$. Já v_i descreve a variação dos valores de cor dos pixels contidos num grupo, em torno da cor representativa correspondente. Por fim, a coerência espacial s é um único número que representa a homogeneidade espacial global das cores predominantes na imagem [33].

2.3.3.3 Cor Escalável

O descritor de cor escalável, pode ser interpretado como um esquema de codificação base, que recorre à transformada de Haar e aplica à aos valores do histograma de cor no espaço de cor HSV (referido na secção 2.3.3.1). De uma forma mais específica, o descritor de cor escalável extrai,

normaliza e mapeia de forma não linear os valores do histograma, numa representação inteira a 4-bit, dando assim mais relevância a valores mais pequenos. A transformada de Haar, é assim aplicada aos valores inteiros a 4-bit através das barras do histograma.

A extração do descritor é realizada com computação de um histograma de cor com 256 níveis no espaço de cor de HSV com a componente matiz (H) quantificada a 16 níveis, e a saturação (S) e o valor (V) quantificados para 4 níveis [31].

Este descritor é tipicamente utilizado na busca de similaridade numa base de dados com conteúdo multimédia e pesquisa em enormes base de dados.

2.3.3.4 Estrutura de Cor

Este descritor é uma generalização do histograma de cores, que apresenta algumas características espaciais da distribuição de cores numa imagem. Tem a particularidade de, para além de apresentar o conteúdo da cor de forma semelhante a um histograma de cor, também apresentar informações sobre a estrutura de uma imagem, sendo esta a característica diferenciadora deste descritor de cor. Em vez de considerar cada píxel separadamente, o descritor recorre a uma estrutura de 8x8 píxeis que desliza sobre a imagem. Ao contrário do histograma de cor, este descritor consegue distinguir duas imagens em que uma determinada cor está presente em quantidades iguais, mas que apresenta uma estrutura num dos grupos de 8x8 com uma cor diferente nas duas imagens. Os valores de cores são representadas no espaço de cor HMMD, sendo o espaço quantificado de maneira não uniforme em 32, 64, 128 ou 256 níveis. Cada valor de amplitude de um nível é representado por um código de 8 bits. Este descritor apresenta um bom desempenho na tarefa de recuperação de imagens baseado na similaridade [34].

2.3.4 Descritores de Textura

A textura das imagens é uma característica visual importante, que tem muitas aplicações na recuperação, navegação e indexação de imagens. Existem três descritores de textura, referenciados na norma MPEG-7 [3]. Em seguida é apresentada uma pequena descrição dos mesmos.

2.3.4.1 Descritor de Textura Homogénea

O descritor de textura homogénea (HTD - *Homogeneous Texture Descriptor*) descreve a distribuição estatística da textura de uma imagem. Uma imagem pode ser considerada como um mosaico de texturas homogéneas, de forma que estas características de textura associadas com as diferentes regiões da imagem possam ser utilizadas para indexar os dados da imagem.

Este descritor fornece uma representação quantitativa utilizando 62 valores, sendo 2 no domínio espacial e 60 no domínio das frequências. Este tenta aproximar-se ao modo de funcionamento do cérebro humano, mais precisamente ao córtex visual [3, 35]. O descritor de textura homogénea é essencialmente utilizado em aplicações de recuperação de imagens por similaridade.

2.3.4.2 Descritor de Histograma de Borda

O descritor de histograma de borda EHD (*Edge Histogram Descriptor*) apresenta-se sobre a forma de um histograma de 80 níveis, que representa a distribuição de borda local de uma imagem. Este descreve as bordas em cada sub-imagem. Estas sub-imagens são obtidas através da divisão da imagem numa grelha 4×4 como pode ser visto na Figura 2.9. Existem 5 tipos de classificação diferentes das bordas de cada sub-imagem, sendo elas: vertical, horizontal, 45-graus, 135-graus e não direcional [3].

Este descritor é utilizado na recuperação de imagens, como por exemplo, imagens naturais ou de esboço, devido à sua textura homogénea.

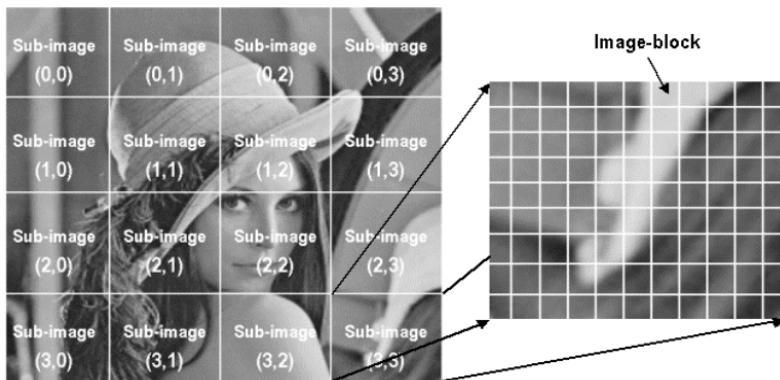


Figura 2.9: Sub-imagem e bloco de imagem. *Retirada de [3]*

2.3.5 Descritores de Forma

Os descritores de forma são dos descritores mais poderosos no reconhecimento de objetos. Isto deve-se ao facto de os seres humanos serem exímios no reconhecimento de objetos característicos exclusivamente através da suas formas, provando que a forma muitas vezes possui informação semântica [4].

2.3.5.1 Descritor de Forma Baseado em Região

O descritor de forma baseado em região (RSD - *Region-based Shape Descriptor*) permite a descrição de objetos simples através da sua forma, com ou sem buracos (figura 2.10) ou de objetos mais complexos que contém múltiplas regiões sem ligação [4]. Assim, este descritor oferece uma forma compacta e eficiente de descrever imagens com várias regiões disjuntas. Este é essencialmente utilizado na deteção de objetos.

2.3.5.2 Descritor de Forma Baseado no Contorno

O descritor de forma baseado no contorno (CSD - *Contour-based Shape Descriptor*) utiliza esta propriedade para descrever uma imagem. O contorno é uma propriedade importante para



Figura 2.10: Exemplo de formas de objetos que podem ser descritas eficazmente pelo descriptor baseado em região. *Retirada de [4]*.

identificação de objetos semanticamente semelhantes. Este consegue distinguir objetos que apresentem formas semelhantes mas que a forma do seu contorno apresenta propriedades bem diferenciadoras como por exemplo, deformações de perspetiva. Esta apresenta um boa eficiência mesmo perante a existência de ruído nos contornos [4].

2.3.6 Descritores Locais

Um descriptor local permite a localização das estruturas locais de uma imagem de forma repetitiva. Estas são codificadas de modo a que sejam invariantes a transformações das imagens, tais como a translação, rotação, mudanças de escala ou deformações. Assim, estes descritores podem ser utilizados para representar uma imagem e podem ser utilizados para diversos fins, tais como, reconhecimento de objetos, reconhecimento de cenas, perseguição de movimento, correspondência entre imagens ou mesmo obtenção de estruturas 3D de múltiplas imagens. Para a extração das características deste descriptor é necessário utilizar um processo com as seguintes etapas [36]:

- Encontrar um conjunto de pontos chave;
- Definir uma região em torno de cada ponto-chave numa escala invariante;
- Extrair e caracterizar o conteúdo da região;
- Calcular o descriptor da região normalizada;
- Combinar os descritores locais.

Em seguida são apresentados duas das técnicas mais representativos dos descritores locais, o SIFT e o SURF.

2.3.6.1 SIFT

O descriptor SIFT (*Scale-Invariant Feature Transform*) [37, 5] é um descriptor local que transforma uma imagem numa grande coleção de vetores de características locais invariantes a translação, rotação, mudanças de escala, e parcialmente invariante a mudanças de iluminação. Pode assim ser utilizado para detetar correspondência entre imagens com diferentes visões de objetos ou

cenas. Este descritor apresenta como característica interessante o facto de compartirilhar uma série de propriedades em comum com as respostas dos neurónios do lobo temporal na visão dos primatas. Os pontos-chave SIFT derivados de uma imagem são usados na indexação numa abordagem de vizinho mais próximo, para encontrar objetos candidatos.

Como foi indicado anteriormente, é necessário levar a cargo uma lista de etapas bem definidas para obtenção de descritores locais. As etapas para o descritor SIFT são as seguintes:

- 1. Deteção de extremos:** A deteção de extremos (máximos e mínimos) é conseguida através da procura realizada em várias escalas e localizações, de modo a serem extraídos pontos de interesse invariáveis à escala e rotação, através da diferença de filtros Gaussianos. Estes extremos detetados quando são máximos, dão origem ao que se designa por pontos chave ou pontos de interesse da imagem.

A equação 2.17 corresponde convolução entre a imagem I e o filtro Gaussiano passa baixo representado pela função G que dá origem a uma imagem filtrada L .

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (2.17)$$

A função G é descrita pela equação 2.18:

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \quad (2.18)$$

em que o filtro varia à escala através do parâmetro σ .

Para a deteção de extremos é necessário a aplicação numa imagem de um filtro de diferença de Gaussianos *DoG* ("Difference of Gaussians") em escalas adjacentes. Isto corresponde à diferença entre uma imagem filtrada por dois filtros Gaussianos iguais mas com escalas diferentes σ e $k\sigma$, como descrito na equação 2.19.

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (2.19)$$

Este filtro *DoG* aplicado a uma imagem provoca uma perda de nitidez (efeito desfoque) e torna-se assim capaz de detetar variações de intensidade nas imagens, como por exemplo, nos contornos. A Figura 2.11 ilustra este processo de obtenção da diferença de Gaussianos.

Segundo Lowe [5] é necessário atingir a escala 2σ para ser possível a construção de um descritor local invariável à escala, logo

$$k = 2^{(1/s)}$$

onde s é o número de intervalos entre imagens obtidas por *DoG* e $D(x, y, \sigma)$ corresponde à primeira imagem e $D(x, y, 2\sigma)$ à última de todo o conjunto de imagens geradas. Também deverão ser assim obtidas $s + 3$ imagens na pilha de imagens filtradas para cada oitava. Cada oitava contém as imagens de Diferença Gaussiana, sendo as que ficam entre as escalas superiores e inferiores designadas de intervalo.

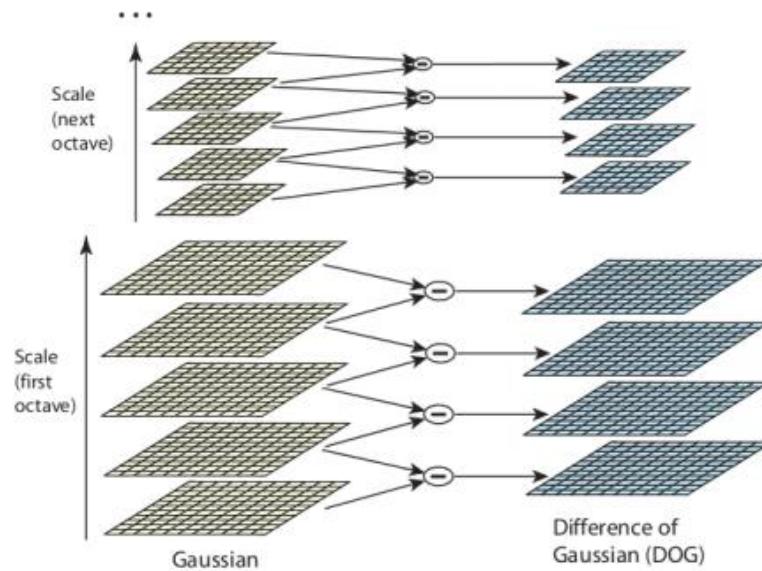


Figura 2.11: Esquematização da obtenção da diferenças de Gaussianos e respetiva. *Retirada de [5]*

Por fim é realizado o processo de deteção de extremos, onde um píxel é comparado com os seus oito vizinhos na imagem atual e com os nove vizinhos das imagens de escalas adjacentes, numa região de 3x3. A figura 2.12 ilustra este processo.

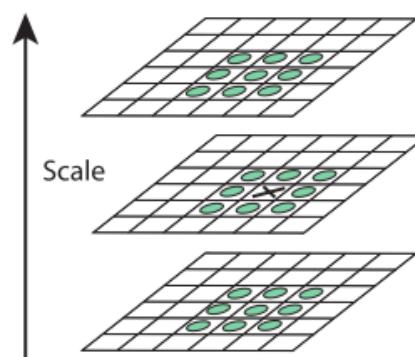


Figura 2.12: Ilustração do processo de deteção de máximos e mínimos das imagens de Diferença Gaussiana. O píxel candidato está marcado com X e os vizinhos com um círculo. *Retirada de [5]*

O próximo passo passa pela localização dos ponto chave.

2. **Localização de pontos chave:** quando detetado um extremo, esse ponto é considerado um candidato a ponto chave ou ponto de interesse. Este ponto foi encontrado através da comparação de um píxel com os seus vizinhos como referido anteriormente. É utilizada então a série de Taylor para obter uma localização mais exata dos extremos, sendo rejeitado caso a intensidade de um extremo seja inferior a um limiar previamente definido.

Figura 2.13: Imagem

Como DoG tem uma boa resposta a arestas e como estas fazem com que os pontos sejam instáveis com ruído, estes necessitam de ser removidos. É assim através da utilização de uma matriz Hessiana 2x2, é possível calcular as curvaturas principais. Caso o rácio seja superior a um limiar previamente definido, o ponto é considerado uma aresta e assim o ponto chave é descartado.

Após a remoção de todos os pontos considerados como não sendo de interesse, fica-se com todos os pontos chave, sendo necessário a atribuição das suas orientações.

3. **Atribuição da orientação dos descritores:** este processo tem como principal finalidade possibilitar a representação de um descritor em relação a sua orientação, permitindo assim que este seja invariante a rotações. Para realizar esta tarefa é utilizada a escala Gaussiana σ para a escolha da imagem filtrada L com a escala mais próxima e com a oitava referente ao ponto avaliado, tornando assim invariante também à escala.

Assim são calculados os gradientes para cada imagem $L(x, y, \sigma)$ de intervalo, referentes às escalas e oitavas utilizadas.

A magnitude e orientação são calculados da seguinte forma:

$$m(x, y) = \sqrt{\left((L(x+1, y) - L(x-1, y))^2 + (L(x, y+1) - L(x, y-1))^2 \right)} \quad (2.20)$$

$$\theta(x, y) = \tan^{-1} \left(\frac{(L(x, y+1) - L(x, y-1))}{(L(x+1, y) - L(x-1, y))} \right) \quad (2.21)$$

Assim, é criado um histograma das orientações para uma região em redor do ponto chave, em que, de todas as orientações obtidas para um ponto, apenas o maior pico e aquelas acima de 80% do valor desse pico são utilizadas para definir a orientação de cada ponto chave.

Por fim, é possível a construção dos descritores para os pontos chave definidos como o ponto a seguir apresenta.

4. **Construção do descritor local:** este é o último passo em que é criado o descritor local para cada ponto de interesse. Para esse processo, é considerado um bloco 16x16 em redor do ponto chave e posteriormente dividido em 16 sub-blocos de 4x4. Por cada sub-bloco é criado um histograma com 8 picos relativos à orientação. Isto faz que no fim seja extraído um vetor de 128 posições para cada ponto chave. Para além deste retorno, são também tomadas várias medidas de modo a que exista robustez suficiente para que esse ponto de interesse seja invariante a mudanças de iluminação e rotação.

O algoritmo SIFT é regularmente utilizado em reconhecimento de objetos ou cenas, tendo sido utilizado em deteção de objetos em *frames* de vídeo [7, 38] com utilização de técnicas mais avançadas de deteção de objetos e cenas utilizando vocabulários visuais como é apresentado na secção 2.3.8.

2.3.6.2 SURF

Outro algoritmo importante de extração de descritores locais é o SURF (*Speeded Up Robust Features*). Este é baseado no SIFT apresentado na secção 2.3.6.1 e também é utilizado, por exemplo, em reconhecimento de objetos ou mesmo na reconstrução 3D. Segundo os autores [6], o descritor SURF é mais rápido (cerca de dez vezes) e robusto do que o SIFT, sendo que, segundo a comparação realizada em [39] comprovou-se que o SIFT é mais lento e não muito bom a mudanças de iluminação, mas apresenta melhores resultados a variações de rotação, mudanças de escala e transformações na imagem.

Este usa a técnica da imagem integral, onde cada píxel de uma imagem recebe um valor igual à soma dos píxeis da sua esquerda e acima, incluindo o próprio. Este também utiliza um filtro Haar em formato de caixa numa sub-região 4x4 em redor de um ponto de interesse, como se pode ver na Figura 2.14, tornando o processo computacionalmente eficiente. Isto é realizado calculando a soma das respostas dos filtros e a soma do módulo das respostas dos filtros nas direções horizontal e vertical, gerando assim 4 valores por cada sub-região. Logo, o SURF retorna um vetor de 64 dimensões, metade do retornado pelo algoritmo SIFT.

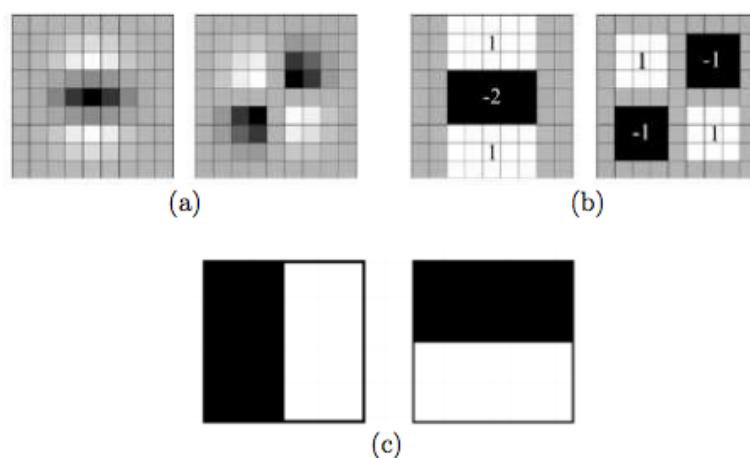


Figura 2.14: (a) Filtros Gaussianos de segunda ordem nas direções yy e xy; (b) aproximação por filtros de caixa; (c) filtros de Haar; As regiões a cinzento têm valor igual a zero. *Retirada de [6]*

2.3.7 Descritores Locais Binários

Os descritores SIFT e SURF são uns dos mais utilizados e conhecidos, mas existem um conjunto de descritores locais mais recentes que se caracterizam por serem descritores locais binários. Em seguida são apresentados alguns exemplos deste tipo:

BRIEF: É um descritor local binário que não possui um mecanismo elaborado de extração de características nem de compensação de orientação como o SIFT ou o SURF, sendo um descritor de elevado desempenho. Este utiliza a informação de um único píxel de uma região, sendo necessário a utilização de um filtro Gaussiano para reduzir a sensibilidade ao ruído. Uma característica dos descritores binários é a seleção pares de pontos (píxels) de uma região para a criação do descritor. Este seleção feita de uma forma aleatória no caso deste descritor. Quando comparados dois pontos, é atribuído o valor 1 quando a intensidade do primeiro ponto é superior ao do segundo, caso contrário, é atribuído o valor 0. O descritor é assim um vetor com informação binária. [40].

ORB: Este é também um descritor binário muito semelhante ao BRIEF, tendo como principais características diferenciadoras o fato de possuir um mecanismo de compensação de orientação que o torna invariante à rotação e de aprendizagem na seleção dos pares de pontos de interesse. [41].

FREAK Este é o mais complexo relativamente aos dois anteriores. Este utiliza da mesma forma que ORB um mecanismo de aprendizagem para seleção dos pares de pontos. A sua principal característica é o fato de utilizar uma distribuição de pontos numa região de uma forma semelhante à retina ocular, em que existe uma maior densidade de pontos na zona central de uma região [42].

2.3.8 Descritores Baseado em Vocabulário Visual

No reconhecimento de documentos de texto é utilizado o conceito de vocabulário. Esta representação é muitas vezes designada por *BoW (Bag Of Words)* que em português significa, saco de palavras, onde existe um conjunto pré-definido de palavras armazenadas.

Um texto é assim caracterizado, através da análise das palavras que possui. Para isso são contabilizadas as palavras que estejam simultaneamente no texto e no *BoW*. Isto permite o acesso um vetor de tamanho fixo que descreve um texto através da frequência das palavras definidas no *BoW*.

Recentemente esta técnica foi adotada em aplicações de extração de informação visual, como por exemplo, é mostrado em [7, 38]. Os autores recorrem a descritores locais invariantes a escala e rotação, para criar um vocabulário visual. A utilização de descritores locais como o SIFT, referido na secção 2.3.6.1, permite a extração de pontos de interesse nas imagens, invariantes a rotação e mudanças de escalas.

Quando efetuado este processo repetidamente com um grande conjunto de imagens, é possível criar *clusters* de regiões de imagens muito semelhantes entre si, como se pode ver na Figura 2.15. Todas estas regiões são candidatas a palavras visuais, sendo selecionada a que representa melhor esse *cluster*. Isto é, é selecionado o centroide desse conjunto de regiões semelhantes entre si, recorrendo ao algoritmo *k-means* referido na secção 2.1.1. Esse centroide passa então a ser considerado uma palavra visual e é adicionado a um vocabulário com outras palavras visuais.

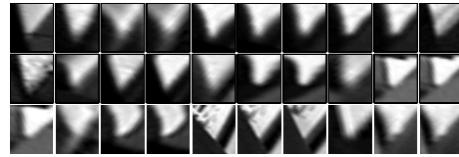


Figura 2.15: Representação de um *cluster* de pontos de interesse semelhantes de imagens distintas.
Retirada de [7]

Por fim é necessária a realização de uma indexação de cada palavra visual às imagens, sendo utilizado um vetor para cada imagem que indica, o número de vezes em que uma determinada palavra visual se repete numa imagem. Neste caso o vetor funciona como em *text mining*, em que existe a contagem da frequência de palavras que ocorrem num determinado documento.

Outro processo possível para indexação do conteúdo visual ou texto, é atribuição de um peso ou ponderação para cada palavra visual numa determinada imagem. Aqui é utilizada a ponderação padrão conhecida como tf-idf ('*term frequency-inverse document frequency*') [7]. Considerando um vetor com k palavras visuais $Vd = (t_1, \dots, t_i, \dots, t_k)$, em que a ponderação de cada palavra é dada por:

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{n_i} \quad (2.22)$$

onde n_{id} é o numero de ocorrências da palavra i num documento d , n_d o número total de palavras no documento d , n_i é o numero de ocorrências da palavra i em todos documentos e N é o número de documentos existentes.

Conclui-se assim que este descritor permite a identificação de imagens semelhante ou reconhecimento de objetos, através da comparação dos vetores de cada imagem, com a informação relativa ao vocabulário posteriormente criado. Este demonstra ser bastante eficiente. Para além disso, segundo Nistér e Stewénius [43], é possível escalar este processo para enormes quantidades de imagens sem perda de performance. Para isso, um vocabulário em forma de árvore, isto é, com a hierarquização das palavras visuais de um vocabulário visual, obtido com *clustering* hierárquico descrito na Secção 2.1.2.

Capítulo 3

Olhó-passarinho: Módulo do Conteúdo Visual

Neste capítulo será introduzido o módulo de informação visual desenvolvido, sendo apresentadas a estrutura e organização deste sistema. Como referido no capítulo 1, este projeto tem como objetivo estender a ferramenta TweeProfiles descrita na secção 2.2, dando-lhe uma dimensão de conteúdo diferente à que possui. Em conjunto com a informação espacial e temporal, o conteúdo das imagens partilhadas em tweets passa também a ser uma fonte de informação. Para que isto seja realizável, foi necessário o desenvolvimento de um módulo que efetue a recolha os dados com a devida filtragem, extraia e processe a informação visual e armazene essa informação de modo a que fosse possível a sua integração com o TweeProfiles.

3.1 Recolha dos Dados

O desenvolvimento deste módulo apenas era realizável com um conjunto de imagens partilhadas no serviço de microblogging Twitter, sendo fundamental a recolha dos dados necessários para esse processo, neste caso, os Tweets. Nesta secção é feita uma descrição do conjunto dos dados disponíveis, das filtragens que foram necessárias realizar e uma apresentação do conjunto de dados finais obtidos e utilizados no desenvolvimento deste projeto de dissertação.

3.1.1 Descrição dos Dados

O primeiro passo para a realização deste projeto de dissertação foi a recolha dos dados necessários. Estes dados foram recolhidos através de uma base de dados MongoDB previamente criada usando a plataforma Socialbus, anteriormente designada por TwitterEcho [21]. Este dados são estruturados sobre a forma de objetos JSON [?] e possuem informação relativa a cada tweet. O JSON é um formato de notação, utilizado para armazenamento e transferência de dados na Internet. Este é o formato utilizado pela base de dados MongoDB.

A Tabela 3.1 apresenta informação relativamente ao número total de tweets que a base de dados contém e o total destes tweets que possuem informação visual. Para além disso, ainda são apresentados o número de tweets com informação visual que provém de alguns serviços externos.

	Total	Com imagem	Twitter	TwitPic	Instagram
Nº Tweets	1704273	86349	202	6100	79210

Tabela 3.1: Descrição em números do total de tweets com indicação, nos que contém URL para imagem, do número de tweets por serviço de partilha de imagem

3.1.2 Filtragem dos Dados

Em primeiro lugar, foi necessário realizar uma filtragem dos dados de modo a apresentarem a informação necessária para a realização deste projeto. O primeiro passo desta filtragem foi recolher todos os tweets que contivessem no seu objeto um URL (*Uniform Resource Locator*) para uma imagem, sendo que esse URL teria de pertencer a um dos seguintes serviços:

- Twitter
- TwitPic
- Instagram

O URL teria que ser válido. Isto é, foi feita uma verificação prévia se a imagem estaria ainda disponível através do endereço existente.

Para ser mais fácil posteriormente uma seleção mais cuidadosa dos tweets foi criada uma base de dados local (SQLite) com uma tabela (Anexo A), contendo os seguintes atributos:

id	Id da linha da tabela.
id_tweet	Id do tweet na base de dados Mongodb.
servico	Nome do serviço de alojamento da imagem.
url	Endereço url para a imagem fonte.
tipo	Identifica se a imagem pertence a um tweet ou retweet.
retweet	Caso a imagem pertença a um retweet e este seja o primeiro da base de dados, assume o valor "primeiro", caso contrário, assume o valor NULL.

Tabela 3.2: Atributos da tabela da base de dados para filtragem dos tweets.

Após a criação desta base de dados, foi decidido selecionar todos os objetos da base de dados do tipo tweet e em que o seu serviço fosse o Instagram. Esta decisão deveu-se ao facto dos retweets se referirem a imagens já existentes e, portanto, a sua recolha iria provocar duplicação de dados. No caso da escolha do Instagram, este deveu-se ao facto de apresentar um número superior de imagens relativamente aos outros serviços.

Para além destes critérios é importante salientar que foi necessário selecionar apenas os dados com georreferenciação.

3.1.3 Conjunto de Dados Final

Por fim armazenou-se um ficheiro JSON com todos os dados que serão utilizados e foi realizado o *download* de todas as imagens relativas a cada tweet e armazenadas localmente em formato JPEG, que era o formato de origem das imagens descarregadas.

Relativamente aos objetos de cada tweet presentes no ficheiro JSON, optou-se por não armazenar tudo para reduzir o tamanho do ficheiro, tendo sido apenas incluídos os elementos de um objeto de um tweet descritos na Tabela 3.3.

_id: \$oid	Identificador do objeto
coordinates: coordinates	Coordenadas da localização da partilha do tweet
created_at	Data da partilha do tweet.
entities: urls: display_url	Endereço url para a imagem fonte
id_str	Identificador do tweet em formato de texto
text	Mensagem do tweet
user: id_str	Identificador do utilizador
user: name	Nome do utilizador

Tabela 3.3: Elementos contidos no objeto com a informação de um tweet.

O formato de um objeto relativo a um tweet é ilustrado em seguida:

```

1 {
2     "_id": {
3         "$oid": "52c6d0f08ef20d397e42b516"
4     },
5     "coordinates": {
6         "coordinates": [
7             -8.61136747,
8             41.14668427
9         ]
10    },
11    "created_at": "Tue Jun 18 17:02:09 +0000 2013",
12    "entities": {
13        "urls": [
14            {
15                "display_url": "instagram.com/p/atTgTbEu0S/"
16            }
17        ]
18    },
19    "id_str": "347036525172772864",
20    "text": "#ogiganteacordou #DilmaNAO #brasilnarua #foradilma #
verasqueumfilhoteunaofogeaaluta #vamospararua\u2026 http://t.co/AF0hcTxUBX"
,
21    "user": {
22        "id_str": "1072354428",
23        "name": "Carlos Roma"
24    }
25 }
```

No caso das imagens armazenadas, foi atribuído o campo *id_str* presente no objeto JSON ao nome do ficheiro de imagem JPEG de modo a que fosse associada cada uma das imagens ao seu respetivo tweet. Assim ao utilizar uma das imagens, apenas é necessário procurar o objeto JSON que possua o atributo *id_str* igual ao nome do ficheiro da imagem para saber a que tweet pertence.

Recolhidos todos os identificadores dos tweets pertencentes ao serviço Instagram que apenas fossem do tipo tweet e que os respetivos tweets apresentassem a informação de geolocalização no campo *coordinates*, avançou-se com o processo de *download* de todas as imagens, tendo sido efetuado com sucesso o *download* de 5964 imagens em 7195 possíveis.

Em suma, o conjunto de dados final utilizado para o desenvolvimento deste projeto foi de 5964 objetos relativos aos tweets contidos num ficheiro JSON e as respetivas 5964 imagens.

3.2 Extração, Processamento e Armazenamento da Informação Visual

O passo seguinte no desenvolvimento do módulo da informação visual foi a implementação de um sistema capaz de extrair, processar e armazenar a informação visual das imagens armazenadas localmente. Para o desenvolvimento deste módulo foi utilizada a linguagem Python, pela sua capacidade de integração de diferentes bibliotecas desde manipulação de estruturas de dados, até mesmo a bibliotecas de manipulação e processamento de imagem. Para a concretização do modelo foram tidos como linhas guia, o processo desenvolvimento de uma ferramenta de pesquisa de imagens disponibilizado por [44]. A arquitetura deste módulo é apresentada na Figura 3.1.

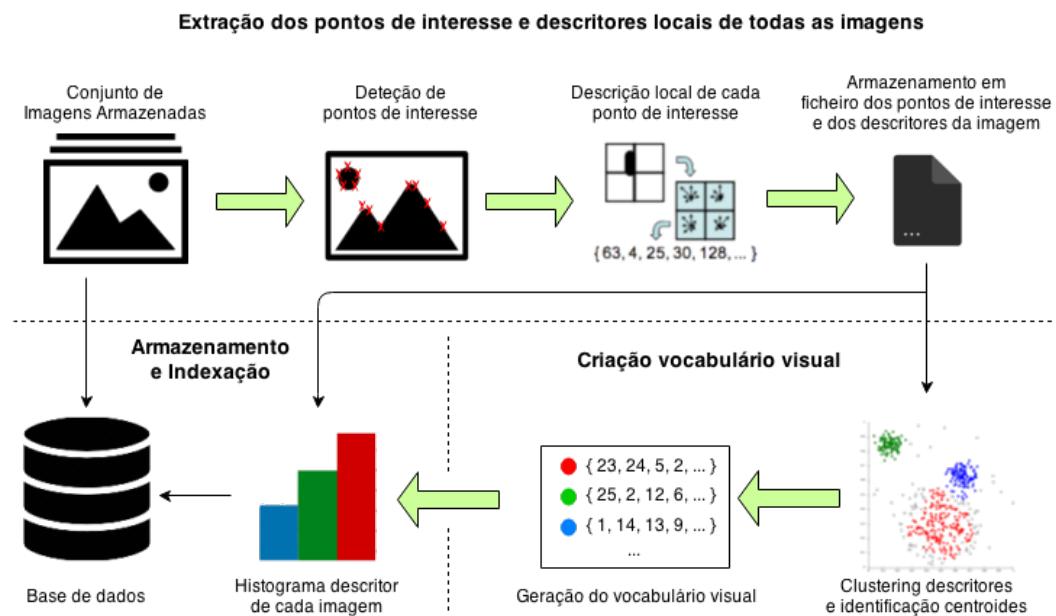


Figura 3.1: Arquitetura do módulo de extração, processamento e armazenamento da informação visual. Adaptada de [8]

Foram desenvolvidos três sub-módulos diferentes: o primeiro é responsável pela extração dos descritores locais, o segundo utiliza o primeiro para gerar um vocabulário visual, e o terceiro utiliza os dois anteriores para armazenar um histograma descritor de cada imagem numa base de dados indexada à sua respetiva imagem. Em seguida é apresentada uma subsecção com a descrição para cada sub-módulo, como representado na Figura 3.1

3.2.1 Extração dos Pontos de Interesse e Descritores Locais

O primeiro passo no desenvolvimento deste módulo passou pela extração da informação visual. Esta informação visual deveria representar uma imagem eficientemente e de forma a possibilitar a criação de um vocabulário visual. Uma das formas possíveis e apresentadas na secção 2.3 é a utilização de um descritor local. Neste caso foi utilizado o descritor SIFT [37, 5], pois como referido na secção 2.3.6, apesar de ser mais lento e menos resistente a mudanças de iluminação, este apresenta melhores resultados a variações de rotação, mudanças de escala e transformações na imagem. Para além disso, foi utilizada a ferramenta e biblioteca *open source* VLFeat [45] que integra alguns dos algoritmos mais utilizados em visão computacional, e que inclui o algoritmo SIFT. Este, apesar de não possuir uma biblioteca para Python, permite a sua utilização através da linha de comandos.

Para o conjunto de imagens existentes foi necessário criar uma cópia de cada imagem em escalas de cinzento e em formato *.pgm* para ser utilizada pelo VLFeat. Isto deve-se ao fato de o descritor SIFT não utilizar a cor das imagens, sendo mais eficiente para a extração dos descritores locais a utilização da imagem com tons cinzas.

Utilizando assim estas imagens, o VLFeat armazena num ficheiro com o formato *.sift* os pontos de interesse e os descritores de uma imagem, sendo necessário criar um ficheiro para cada imagem. Nesses ficheiros os dados são armazenados em formato ASCII, onde cada linha contém as coordenadas, escala e ângulo de rotação para cada ponto de interesse, nos primeiros 4 valores respetivamente, correspondendo os restantes ao vetor descritor de tamanho 128 como referido na Secção 2.3.6. Em seguida é apresentado um exemplo ilustrativo desta informação armazenada:

1	318.861	7.48227	1.12001	1.68523	0	0	0	1	0	0	0	0	0	11	16	0	...
2	318.861	7.48227	1.12001	2.99965	11	2	0	0	1	0	0	0	173	67	0	0	...
3	54.2821	14.8586	0.895827	4.29821	60	46	0	0	0	0	0	0	99	42	0	0	...
4	155.714	23.0575	1.10741	1.54095	6	0	0	0	150	11	0	0	150	18	2	1	...
5	42.9729	24.2012	0.969313	4.68892	90	29	0	0	0	1	2	10	79	45	5	11	...
6	229.037	23.7603	0.921754	1.48754	3	0	0	0	141	31	0	0	141	45	0	0	...
7	232.362	24.0091	1.0578	1.65089	11	1	0	16	134	0	0	0	106	21	16	33	...
8	201.256	25.5857	1.04879	2.01664	10	4	1	8	14	2	1	9	88	13	0	0	...
9

Como o objetivo era utilizar as imagens originais, foram eliminadas as imagens temporárias com o formato *.pgm*. O passo seguinte passou pelo desenvolvimento do submodelo responsável pela criação do vocabulário visual, que é apresentado na subsecção seguinte.

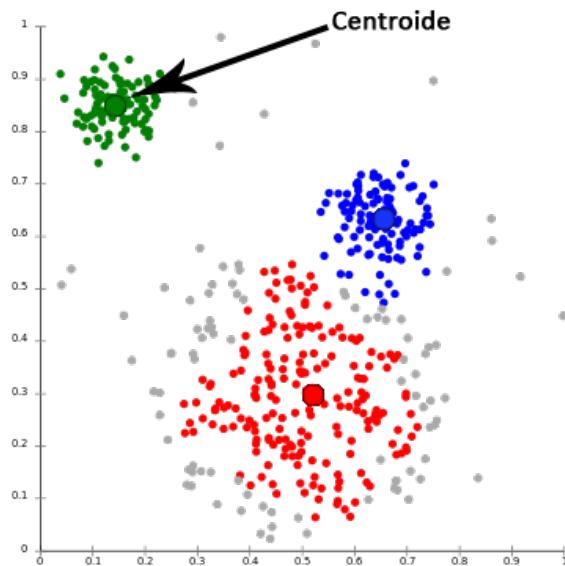


Figura 3.2: Exemplo de projeção de centroides após tarefa de *clustering* num espaço a duas dimensões.

3.2.2 Criação do Vocabulário Visual

Este módulo é o responsável pela criação do vocabulário visual. Para a sua concretização foi necessário utilizar os ficheiros de extensão *.sift* reproduzidos através da ferramenta VLFeat [45] no módulo anterior.

As palavras visuais não são nada mais do que um conjunto de vetores de características de imagens. Assim um vocabulário visual é o conjunto destas palavras visuais. Como cada imagem possui muitos descritores locais, sendo que muitos podem ser semelhantes, é necessário agrupar todos os descritores de um conjunto de imagens e detetar aqueles que possam representar um conjunto de descritores semelhantes. Cada *cluster* corresponde a uma palavra visual, sendo a palavra visual respetiva definida pelo seu centroide.

Para criar um vocabulário visual foi então necessário utilizar um algoritmo de *clustering*. Foi escolhido um algoritmo de *clustering* por partição, em particular o *k-means* por ser um dos mais utilizados e eficiente, como foi referido na secção 2.1.1. O algoritmo foi aplicado aos descritores de um subconjunto de imagens aleatoriamente selecionadas do conjunto de imagens armazenadas localmente. Neste caso foi utilizado aproximadamente 8% das imagens para não comprometer o tempo de processamento. Foi atribuído a *k* o valor 1000, de forma a serem gerados cerca de 1000 *clusters*. Isso significa que o vocabulário visual possuirá cerca de 1000 palavras visuais.

Para utilizar este vocabulário visual é necessário armazená-lo e indexar cada palavra visual a cada imagem. O sub-módulo responsável por este processo é descrito na secção a seguir.

3.2.3 Armazenamento da Informação Visual

Com o vocabulário visual criado foi necessário armazenar esta informação e indexar cada palavra visual às imagens, de modo a que seja possível a comparação entre imagens diferentes. Para armazenar este vocabulário desenvolveu-se o módulo que execute esta tarefa, sendo que, foi necessário seguir alguns passos tais como, a criação de um histograma descritor do vocabulário visual de cada imagem e de uma base de dados com a informação necessária para a utilização do histograma descritor de cada imagem.

A criação do histograma foi o primeiro passo do desenvolvimento deste sub-módulo, em que foi utilizado o vocabulário disponibilizado através do sub-módulo apresentado anteriormente. Como o vocabulário é constituído por vetores descritores que representam cada uma das palavras visuais, para a criação do histograma para cada imagem foi necessário utilizar novamente os ficheiros com os descritores locais SIFT de cada imagem e assim projetar no espaço cada ponto-chave, de modo a ser atribuído a cada um destes a sua palavra visual, sendo esta a que se encontrar à menor distância. Foi então possível após este processo criar um histograma, baseado na contagem das palavras visuais em cada imagem.

Assim o próximo passo passou pelo armazenamento dos histogramas e a indexação a sua respetiva imagem. Como todo este módulo foi executado em modo *offline*, optou-se pela utilização de uma base de dados local SQLite [46]. Esta funciona de modo semelhante a uma base de dados MySQL [47] ou PostgreSQL [48], sendo que pode ser desenvolvida e acedida sem recurso a um servidor. Esta é normalmente utilizada apenas para desenvolvimento, como é o caso deste projeto de dissertação. Para a sua concretização foi criado um esquema simples apenas com três tabelas como ilustrado na Tabela 3.4. A tabela *imlist* contém o nome de todas as imagens através do atributo *filename*, a tabela *imwords* contém índice das palavras visuais através do atributo *wordid*, a identificação do vocabulário com o atributo *vocname* e o índice das imagens em que as palavras visuais aparecem com o atributo *wordid*. Por fim, a tabela *imhistograms* contém o histograma de palavras visuais completo para cada imagem com o atributo *histogram*. Este atributo é associado então a vetor que descreve a respetiva imagem associada, tal como um histograma, em que cada bin possui a frequência de uma determinada palavra visual.

imlist	imwords	imhistograms
rowid	imid	imid
filename	wordid	histogram

imhistograms	imwords	imlist
imid	imid	rowid
histogram	wordid	filename
vocname	vocname	

Tabela 3.4: Esquema da base de dados SQLite

3.3 Matriz de Distâncias entre Imagens

Com o módulo descrito na secção anterior 3.2, a informação que descreve cada uma das imagens armazenadas localmente está alojada e é acessível através da base de dados local criada. Esta informação está no formato de um histograma que descreve a imagem através de um vocabulário visual. Este histograma é um vetor de tamanho único e igual para todas imagens, o que permite uma fácil comparação entre eles. Ou seja, permite calcular facilmente a distância entre eles. Como cada histograma está associado a uma imagem, calcular a distância entre dois histogramas equivale a calcular a distância entre duas imagens.

O TweeProfiles utiliza o algoritmo DBSCAN [9] apresentado na secção 2.1.3 para realizar a tarefa de *clustering* relativamente ao conteúdo e às dimensões espacial e temporal. O *clustering* baseado em densidades, neste caso mais concreto, com o recurso ao algoritmo DBSCAN, pode utilizar uma matriz de distâncias entre objetos para realizar a tarefa de *clustering*, sendo que retorna um número não fixado previamente de *clusters* dependentes dos dados. A matriz de distâncias trata-se apenas de uma matriz $N \times N$ em que N é o número total de objetos. A Figura 3.3 apresenta a estrutura de uma matriz distância, em que 0 é a distância de um objeto a ele mesmo.

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

Figura 3.3: Estrutura de uma matriz de distâncias. *Retirada de [9]*

Tendo existido algumas limitações de *hardware*, mais especificamente da memória do computador, e do tempo de processamento devido ao acesso à base de dados local para obter aos histogramas descritores de cada imagem, optou-se por utilizar o mesmo procedimento utilizado no TweeProfiles [2] e dividir os dados em três partes distintas, resultado em três subconjuntos de 1988 tweets cada. Esta divisão foi efetuada tendo em conta uma divisão temporal, isto é, foi garantido que todos tweets estavam ordenados por ordem cronológica.

Para calcular as matrizes de distâncias para o conteúdo visual utilizou-se o histograma descriptor de cada imagem, calculando a distância entre histogramas através da função de distância Euclidiana apresentada na secção 2.1.5. Assim, resultaram três matrizes distância, cada uma com uma dimensão de 1988×1988 .

Com as matrizes distâncias calculadas, estavam reunidas as condições para a integração do modelo no TweeProfiles. O próximo capítulo irá abordar da integração deste modelo com o TweeProfiles e o desenvolvimento da ferramenta Olhó-passarinho e os seus resultados ilustrativos.

Capítulo 4

Olhó-passarinho: Aplicação Web

Neste capítulo será abordada a ferramenta desenvolvida com a descrição da arquitetura implementada, do processamento da informação de espaço e tempo e da sua integração com a informação visual de modo a aplicar a tarefa de *clustering*. Por fim será apresentada a visualização dos resultados ilustrativos.

4.1 Arquitetura do Sistema

A arquitetura do sistema desenvolvido é apresentada na Figura 4.1. Esta apresenta uma divisão entre os serviços externos e o modelo desenvolvido. Este modelo foi desenhado de modo a que existisse uma separação entre o tratamento de toda a parte de processamento dos dados e a visualização. Assim existiu um *back-end* com todos os ficheiros e módulos desenvolvidos e um *front-end* que representa a aplicação web para visualização dos resultados. No *back-end* existe também uma divisão entre dois módulos fundamentais, o módulo de processamento da informação visual, responsável por tratar a informação das imagens como descrito no Capítulo 3, e o módulo que utiliza essa informação através do processo de *Data Mining* já desenvolvido no TweeProfiles [2].

Os serviços externos correspondem à base de dados MongoDB para a recolha dos tweets e os serviços Twitter e Instagram para a recolha das imagens através do URL. No caso do modelo desenvolvido, o *back-end* guarda os ficheiros JSON com os dados e as imagens necessárias, tanto para o módulo de processamento da informação visual como para a extração e processamento do dados espaço-temporais, explicados na Secção 4.2. Os dados processados no módulo da informação visual e os dados espaço-temporal extraídos dos tweets são assim utilizados no processo de *Data Mining*, onde é aplicada a tarefa de *clustering* como explicado na secção 4.3 apresentada mais adiante. Deste processo resultam os *clusters* calculados através de vários parâmetros, sendo esta informação armazenada em ficheiros. Por fim, foi utilizada a *microframework* Flask [49] para desenvolvimento de aplicações web em Python, que permitiu o desenvolvimento da aplicação Olhó-passarinho para visualização dos resultados.

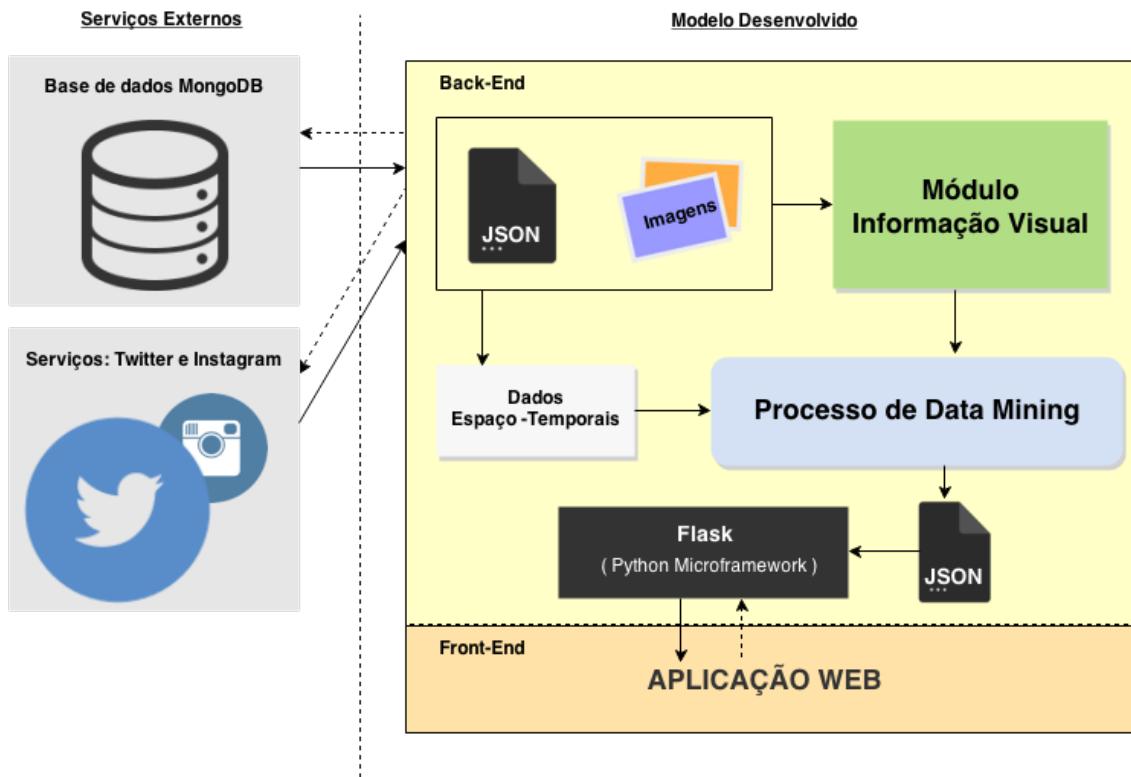


Figura 4.1: Arquitetura do sistema completo

4.2 Informação Espaço-Temporal

Uma das características principais tanto do TweeProfiles como do Olhó-passarinho é a combinação das dimensões espaço-temporais com o conteúdo. Para isto ser possível foi realizada a recolha da informação espacial e temporal dos tweets para o cálculo das respetivas matrizes de distância entre tweets. Esta informação apenas foi recolhida dos tweets associados às imagens armazenadas localmente. É importante referir ainda que, tal como exposto na Secção 3.3, os dados foram divididos em três subconjuntos com 1988 tweets cada.

Em primeiro lugar foi recolhida a informação espacial. Neste caso os dados possuem a informação de latitude e longitude do ponto onde foi enviado o tweet. É importante lembrar que todos os tweets possuem informação de geolocalização, tendo este sido um dos critérios de seleção dos mesmos como explicado na Secção 3.1. Para calcular a distância entre tweets utilizou-se a função distância Haversine abordada na Secção 2.3.3.1. Neste caso foi calculada a distância em quilómetros, tendo sido considerado o valor do raio da Terra igual a 6371 Km.

Posteriormente foi então recolhida a informação temporal dos tweets, sendo que esta informação contém as três primeiras letras do dia e mês, o número do dia, a hora, minutos e segundo, o fuso horário e o ano da partilha do tweet. Assim a informação temporal apresenta-se como o exemplo ilustrativo seguinte: Tue Jun 18 17:02:09 +0000 2013. Para o cálculo da distância entre datas foi utilizada a função distância Euclidiana, que se pode resumir ao módulo da diferença entre

o tempo de dois tweets.

Como resultado, obteve-se para cada dimensão três matrizes de 1988 x 1988. Cada uma dessas três matrizes de cada dimensão representa um subconjunto dos dados e consequentemente um intervalo de tempo. Neste caso cada subconjunto temporalmente é representado na Tabela 4.1

	Hora de início	Data de início	Hora de fim	Data de fim
Subconjunto 1	13:01:46	17 de Junho de 2013	03:14:05	18 de Junho de 2013
Subconjunto 2	03:14:15	18 de Junho de 2013	00:40:08	19 de Junho de 2013
Subconjunto 3	00:40:29	19 de Junho de 2013	23:06:54	19 de Junho de 2013

Tabela 4.1: Distribuição temporal entre os diferentes subconjuntos de dados

4.3 Clustering da Informação Visual, Espacial e Temporal

A tarefa de *clustering* é o último passo para a obtenção dos resultados finais. Este é o processo que engloba os dados resultantes de todo o processamento da informação tratada e discutida anteriormente.

Para a tarefa de *clustering* optou-se pelo mesmo algoritmo utilizado no TweeProfiles [2], o DBSCAN. Este apresenta algumas vantagens na utilização de dados recolhidos de redes sociais, pois não necessita de uma predefinição do número de *clusters* que se pretende obter, sendo este definido através da distribuição em densidade dos objetos, como referido na Secção 2.1.3.

Antes de utilizar os dados para a tarefa de *clustering*, neste caso as matrizes já calculadas com as distâncias entre tweets para as dimensões temporal, espacial e de conteúdo visual, foi necessário realizar a normalização das distâncias através da função normalização mínimo-máximo:

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \quad (4.1)$$

em que x_{max} é o valor máximo presente na matriz e x_{min} o valor mínimo. Neste caso o valor mínimo será sempre 0, que equivale à distância de um objeto a si mesmo. Assim a equação simplificada é apresentada na equação 4.2

$$x' = \frac{x}{x_{max}} \quad (4.2)$$

Após a normalização, realizou-se a combinação entre as matrizes, com atribuição de vários pesos a cada matriz, de forma a que a soma dos diferentes pesos fosse igual a 1. Para isso optou-se por dividir os pesos em fatores de 0.333, em que os valores de cada matriz podem assumir os pesos: 0, 0.333, 0.666, 1.

4.4 Visualização

A aplicação web desenvolvida é responsável pela visualização dos resultados obtidos pela tarefa de *clustering*. Esta apresenta três secções principais de visualização dos *clusters* pelas

diferentes dimensões utilizadas. O primeiro é um mapa, onde é apresentada a distribuição dos tweets, como pode ser visto na Figura 4.2 e a respetiva distribuição dos *clusters* geograficamente. Este foi desenvolvido recorrendo à API Javascript do Google Maps v3 [50] disponibilizada pela Google, sendo toda ela controlada através da linguagem de programação Javascript. É possível utilizar e controlar um mapa de modo a adicionar diferentes componentes visuais. Os tweets foram assim representados por pequenos círculos azuis, e os *clusters* por círculos vermelhos com transparência como será ilustrado mais à frente.



Figura 4.2: Distribuição de tweets na dimensão espacial

A segunda secção é responsável pela representação da distribuição dos *clusters* na dimensão temporal e para isso foi utilizada a ferramenta Google Charts, mais especificamente a API Timeline [10]. Esta, tal como a API do Google Maps, também é desenvolvida em Javascript, e permite criar um gráfico com barras de duração temporal, como podemos ver na figura 4.3.

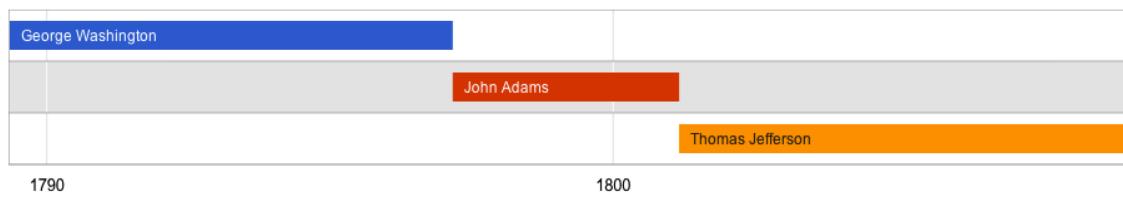


Figura 4.3: Exemplo ilustrativo da ferramenta Timeline. Retirada de [10]

Por último, a secção de visualização de imagens que pertencem a um *cluster*, onde é apresentada uma matriz com nove imagens, como apresentado na Figura 4.4. As imagens presentes nos *clusters* são escolhidas aleatoriamente, havendo ainda a possibilidade de ir modificando as

imagens visíveis. É possível também clicar numa das imagens, visualizar a mesma em tamanho maior, ver a informação relativa ao tweet a que a imagem pertence, o nome do utilizador, o texto e a data de partilha do tweet. Para além disto, é possível aceder ao tweet original através de um botão com essa indicação. Esta secção apresenta ainda o nome e a informação temporal do *cluster*.



Figura 4.4: Exemplo ilustrativo da visualização da matriz para visualização de nove imagens.

A aplicação apresenta também os controlos para escolher o subconjunto que se pretende visualizar, e controlos para definir o peso que se pretende atribuir a cada dimensão.

4.5 Resultados Ilustrativos

Nesta secção serão apresentados alguns resultados ilustrativos do tipo de conhecimento que se pode obter com a ferramenta Olhó-passarinho. Neste relatório não é possível apresentar todos os resultados devido ao número de diferentes combinações possíveis.

A dimensão dos círculos que representam os *clusters* é proporcional ao número de tweets presentes nesse mesmo *clusters*, isto é, quanto maior o número de tweets maior é a respetiva circunferência. Assim a Figura 4.5 um exemplo da visualização da distribuição dos *clusters* no espaço geográfico onde é atribuída 100% do peso a dimensão espacial. Podemos ver que foram gerados sete *clusters* e que existe uma clara divisão espacial entre eles. Podemos localizar um *cluster* na Europa mais sobre a zona do Reino Unido, dois na Ásia em que um situa-se sobre países junto do Golfo do Pérsico e outro sobre o Japão, um na Austrália, dois na América do Norte mais especificamente sobre o Oeste e o Este dos Estados Unidos da América e por último na América do Sul, sendo o *cluster* de com maior dimensão relativamente aos outros, e localiza-se sobre o Brasil.

Já no caso em que é atribuído 100% do peso apenas à dimensão temporal obtemos dois *cluster* como perceptível pela Figura 4.6. No caso da sua visualização temporal, verificamos pela Figura 4.7 que obtém-se um intervalo temporal em que não existe nenhum tweet associado a um *cluster*, podendo-se dever ao facto de aquele período apresentar um volume muito mais reduzido de tweets

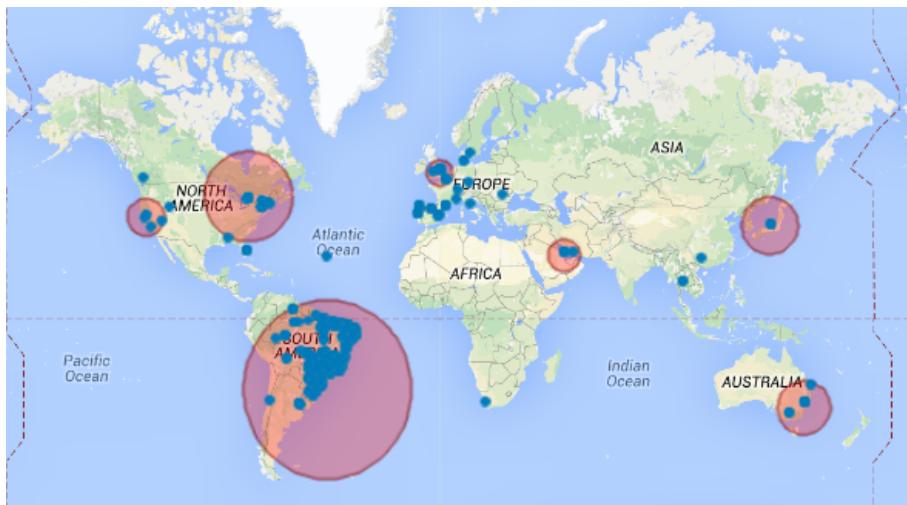


Figura 4.5: Distribuição dos *clusters* no mapa calculado exclusivamente através da dimensão espacial

levando a um dispersão dos mesmos no tempo e consequentemente a uma menor densidade de tweets.

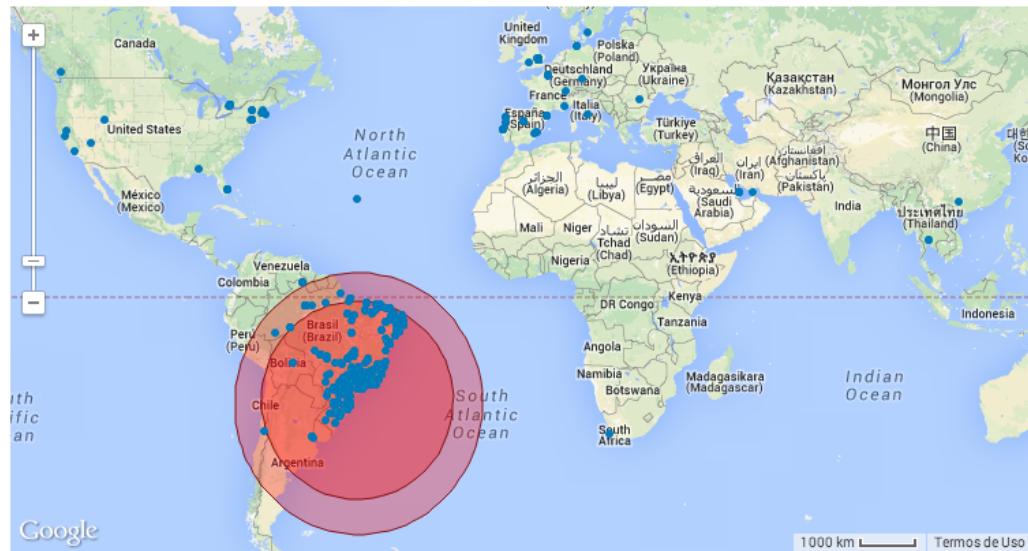


Figura 4.6: Projeção do *Clusters* no mapa calculado exclusivamente através da dimensão temporal

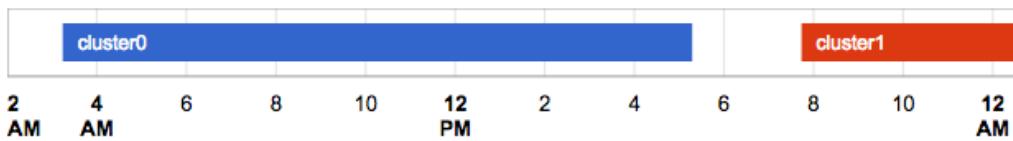


Figura 4.7: Projeção do *Clusters* no tempo calculado exclusivamente através da dimensão temporal

Para o caso da atribuição de 100% do peso apenas ao conteúdo visual, o resultado ilustrativo no mapa é apresentado na Figura 4.8. Para este caso temos a representação de 3 *clusters*. Estes

estão localizados sobre o Brasil e apresentam tamanhos distintos, proporcionais ao número de tweets.

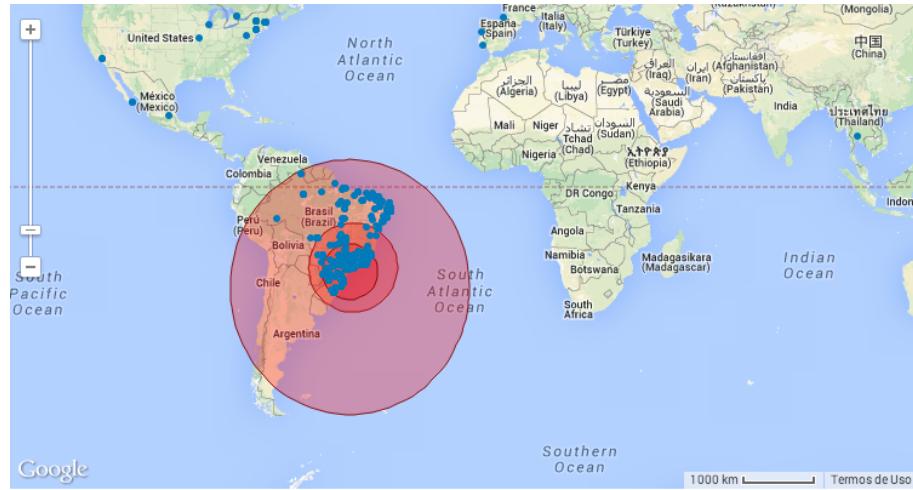


Figura 4.8: Projeção do *Clusters* no mapa calculado exclusivamente através do conteúdo visual

Como referido na Secção 4.4, o Olhó-passarinho permite a visualização do conteúdo visual dos *clusters*. A Figura 4.9a apresenta a visualização de uma amostra de imagens pertencentes a um *cluster*. Neste caso o *cluster* é o mais pequeno representado na Figura 4.8 e contém apenas 9 imagens, o número mínimo definido para a criação de um *cluster*. É visível que estas apresentam o mesmo conteúdo visual, uma bandeira do Brasil e uma citação de uma pessoa com o nome de Federico Devito. Este pormenor pode ser visualizado nas Figuras 4.9b e 4.9b



(a) Visualização de uma amostra de imagens do um *cluster* (b) Visualização de uma das imagens do *cluster* (c) Visualização de outra imagem do *cluster*

Figura 4.9: Exemplo da visualização do conteúdo visual de um *cluster* calculado com 100% do peso para a dimensão das imagens

O exemplo apresentado em seguida mostra-nos o resultado da combinação entre dimensões com atribuição do mesmo peso a cada uma delas. Assim cada dimensão presenta um peso de

33.33%. A Figura 4.10 ilustra a distribuição dos *clusters* no espaço. Por outro lado, a Figura 4.11 mostra a distribuição dos mesmos *clusters*, mas neste caso, na dimensão temporal.

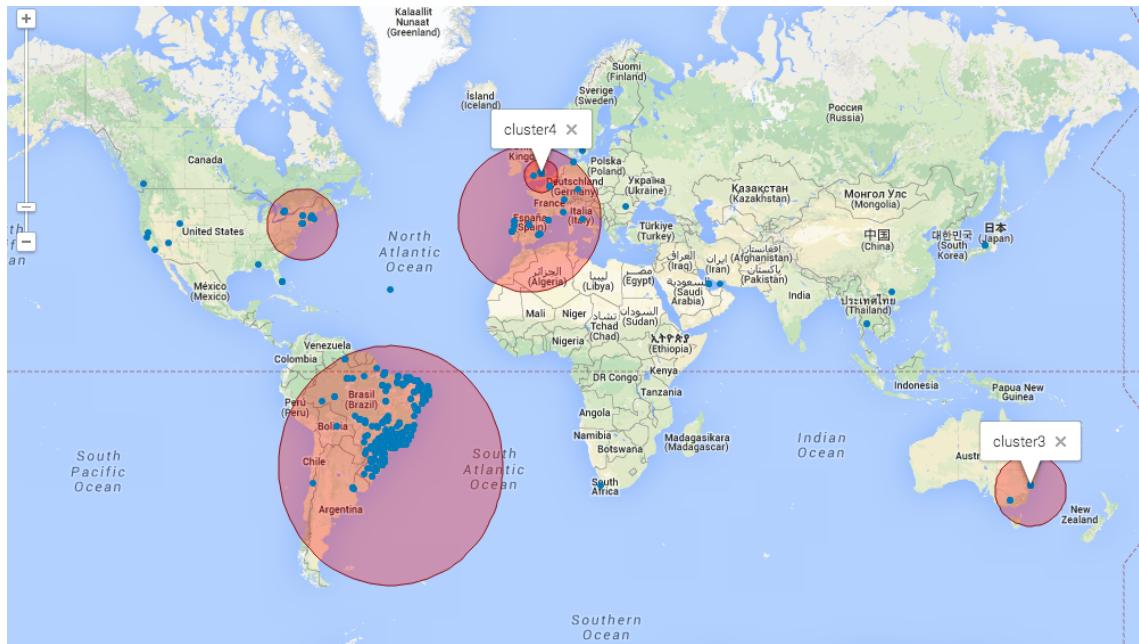


Figura 4.10: Projeção dos *Clusters* no mapa com peso atribuído a cada dimensão de 33.33%

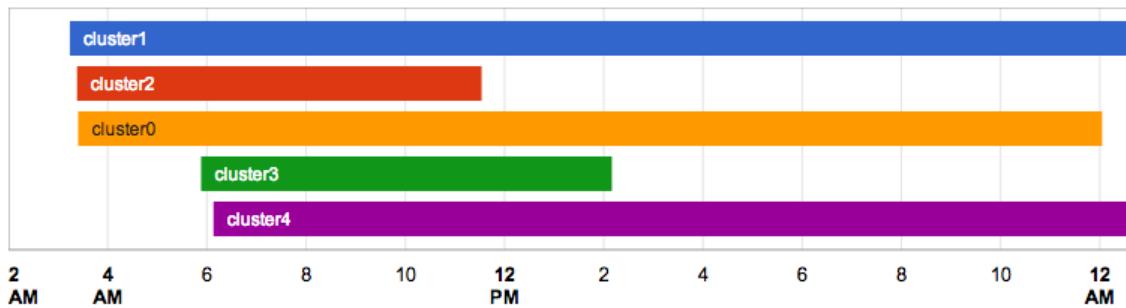


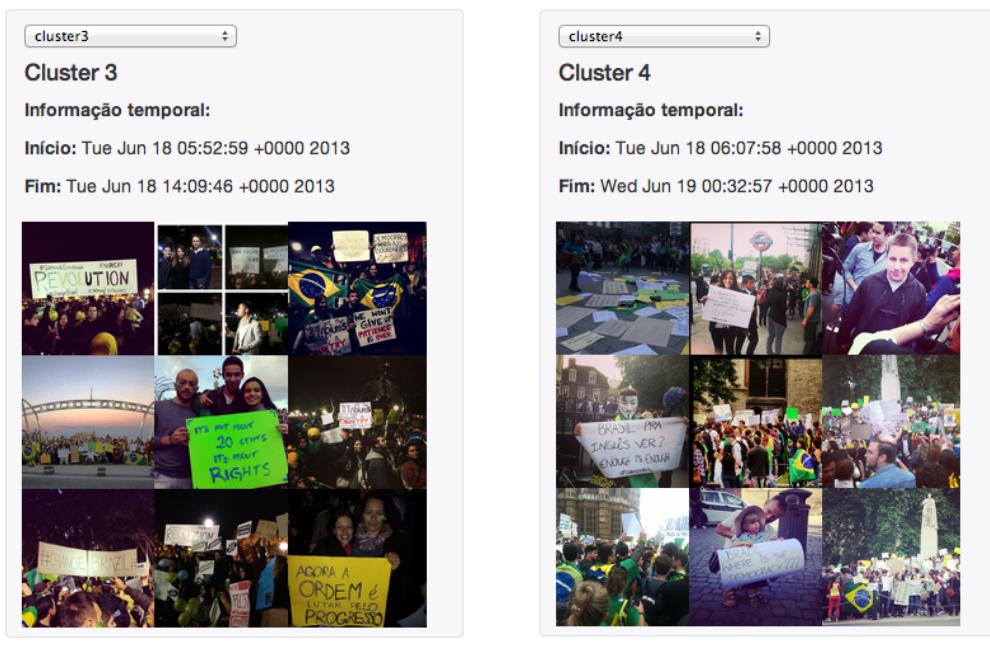
Figura 4.11: Projeção dos *Clusters* no tempo com peso atribuído a cada dimensão de 33.33%

No mapa podemos ver a identificação do *Cluster 3* localizado na Austrália e o *Cluster 4* no Reino Unido. Uma amostra do seu conteúdo pode ser visualizado na Figura 4.12. Aqui podemos notar que o conteúdo visual em ambos os *clusters* são muito semelhantes, sendo assim perceptível neste caso a influência das dimensões espaço-temporal.

4.6 Sumário

Em suma, neste capítulo foi apresentado a concretização da integração do modelo da representação da informação visual desenvolvido, descrito no Capítulo 3, para estender o TweeProfiles.

O primeiro passo passou pelo cálculo as matrizes de distância para a dimensões temporal e espacial. Isto permitiu efetuar combinação entre as várias matrizes das diferentes dimensões. Em



(a) Visualização de uma amostra de imagens do cluster 3 (b) Visualização de uma amostra de imagens do cluster 4

Figura 4.12: Exemplo da visualização do conteúdo visual de dois *clusters* calculados com 33.33% do peso para cada uma das dimensões

seguida foi efetuado o processo de *data mining* que culminou nos diferentes *clusters* para diferentes combinações. Feito isto, foi então desenvolvid a aplicação Web que permitiu a visualização dos resultados e controlo das diferentes combinações. Por fim foram apresentados alguns resultados ilustrativos.

Os exemplos escolhidos para apresentar os resultados ilustrativos pretendem demonstrar as diferentes potencialidades da ferramenta, focando principalmente na atribuição de pesos às diferentes dimensões e na visualização dos resultados dessas mesmas combinações. Para além das combinações demonstradas, é possível realizar mais seis combinações diferentes. Existe também ainda a possibilidade de escolher um dos três diferentes intervalos de tempo que correspondem a cada um dos subconjuntos de dados previamente criados.

Capítulo 5

Conclusões e Trabalho Futuro

Neste capítulo é apresentado um resumo de todo o trabalho realizado, são discutidas algumas conclusões retiradas do desenvolvimento desta dissertação e são apresentadas sugestões para trabalho futuro.

5.1 Resumo

Durante o período dedicado à realização do projeto de dissertação, foi seguida uma sequência de etapas que culminou num sistema capaz de identificar e visualizar *clusters* no espaço, no tempo e através do conteúdo, em particular, das imagens. Este também permitiu visualizar e navegar por fotografias partilhadas no serviço de *microblogging* Twitter contidas num determinado *cluster*.

Inicialmente foi feita uma recolha dos dados necessários para o desenvolvimento deste projeto de dissertação. Este dados foram recolhidos através de base de dados MongoDB e possuíam a informação relativa a tweets partilhados na rede social Twitter. Este tweets eram uma amostra com a duração de três dias que continham, na sua maioria, conteúdo relativo às manifestações que ocorreram em Junho de 2013 no Brasil. Como o objetivo era a descoberta de padrões através de fotografias, foi necessário armazenar localmente todas as imagens partilhadas no Twitter. Devido ao número muito pequeno de imagens partilhadas diretamente pelo Twitter, foi necessário optar por imagens com origem no serviço de partilha de imagens Instagram, por este permitir o acesso a um número bem superior de imagens. Os dados desses tweets também foram armazenados no formato JSON.

O passo seguinte passou pelo desenvolvimento de um módulo responsável pela extração e processamento de informação visual para descrever as imagens. Para isto foi criado um vocabulário visual de tamanho fixo utilizando o descriptor local SIFT. Este módulo foi também responsável pelo armazenamento da informação visual numa base de dados local. O desenvolvimento deste módulo permitiu uma representação das imagens de uma forma mais eficiente e compacta, e tornou possível a comparação entre imagens para a criação de uma matriz de distâncias para ser utilizada na tarefa de *clustering* do processo de *Data Mining*. Para o cálculo das distâncias entre as imagens,

isto é, a distância entre os vetores descritores das imagens, foi utilizada a função de distância Euclidiana.

Proseguiu-se com a produção das matrizes de distância entre tweets, utilizando as funções de distância de intervalo de tempo e Haversine, respetivamente para a dimensão temporal e espacial. Posteriormente foi realizada a normalização das matrizes através da função mínimo-máximo. Com as matrizes criadas e normalizadas foi possível realizar várias combinações entre as três diferentes dimensões atribuindo pesos diferentes a cada. Com esta integração concluída, estavam reunidas as condições para utilizar esta informação no processo de *Data Mining* desenvolvido no Tweeprofiles para a obtenção dos *clusters* com o algoritmo DBSCAN.

Foi desenvolvida a aplicação web em Python, recorrendo a microframework Flask, para visualização dos resultados através do conteúdo dos tweets e das diferentes dimensões já referidas. Uma mapa e um gráfico temporal foram implementados utilizando diferentes bibliotecas Javascript. Também utilizando a linguagem Javascript foi desenvolvido um *widget* que permite navegação pelo conteúdo visual e pela informação dos tweets respetivos, incluindo a possibilidade de acesso ao tweet original. Foram ainda adicionados controlos para a seleção do peso atribuído a cada dimensão e seleção de três diferentes intervalos de tempo.

Após a finalização deste projeto de dissertação foi feita uma análise a todo o processo realizado, tendo sido concluído que os objetivos principais propostos foram atingidos. Apesar disso, alguns objetivos mais ambiciosos não foram atingidos devido a um conjunto diversificado de fatores. Na próxima secção são discutidas algumas decisões tomadas e sugeridas alternativas possíveis a considerar de forma a que possam ser atingidos objetivos mais ambiciosos e dar continuidade a este projeto num trabalho futuro.

5.2 Discussão e trabalho futuro

Nesta secção serão discutidas algumas decisões tomadas e apresentadas sugestões para trabalho futuro.

Base de dados: A base de dados utilizada apresentava um conjunto de dados muito específico como já foi referido anteriormente. A escolha deste conjunto teve como principal objetivo tentar encontrar padrões num evento, como foi o caso das manifestações no Brasil. Para além disto, este evento garantiu-nos um acesso a um número significativo de fotografias. No entanto seria interessante analisar os resultados utilizando uma base de dados mais diversificada. Isto é, base de dados com tweets relativos a eventos mais específicos e outras com dados mais genéricos.

Módulo informação visual: Apesar de não ter sido realizado um estudo empírico sobre o desempenho das várias alternativas de representação da informação visual descritas na Secção 2.3, o método escolhido, SIFT, obteve resultados que levaram consequentemente a resultados finais que, por inspeção visual, parecem frequentemente adequados. No entanto, é importante referir que no trabalho futuro pode fazer sentido realizar uma análise mais aprofundada.

Já na criação do vocabulário visual, apenas foram utilizadas cerca de 8% do total de imagens armazenadas devido a limitações de memória computacional. Assim no trabalho futuro, deverá ser tido em conta a utilização de um número maior de imagens para garantir um vocabulário mais robusto.

Outro aspecto a ser considerado seria, a conjugação do descritor de cor com o vocabulário visual para descrever as imagens. Isto possibilitaria uma melhor descrição das imagens e permitiria distinguir mais eficazmente cenários onde a cor é um fator distintivo, como por exemplo, fotografias de praias, alimentos ou mesmo locais com vegetação, como jardins ou parques naturais onde predomina a cor verde. Esta opção poderia ser disponibilizada ao utilizador, decidindo este se pretende utilizá-la ou não. Esta deve ser optativa pois foi verificado que quando se procura casos de conjuntos de imagens onde o conteúdo é por exemplo textos ou desenhos, a utilização da descritor de cor seria irrelevante podendo-se considerar, neste caso, a inclusão de ruído nos dados.

Matrizes de distância: No caso da matriz relativa ao conteúdo visual, a função de distância Euclidiana foi utilizada para a comparação entre os histogramas com a descrição do vocabulário visual de cada imagem garantindo um cálculo rápido e eficiente, mas a utilização de uma função mais específica para cálculo de distância entre histogramas, deverá ser considerada no trabalho futuro.

Para a combinação das matrizes foram utilizados passos de 33.33% pois este permitiu a existência de uma combinação com pesos iguais para as diferentes dimensões. A utilização de passos mais pequenos foi considerada, mas esta iria fazer crescer muito o número de combinações possíveis.

Para a normalização das matrizes foi utilizada a função de normalização mínimo-máximo para poder utilizar o algoritmo de *clustering* com parâmetros baseados em percentagens. A utilização da normalização *z-score* através da média e desvio padrão é uma alternativa que altera a distribuição de densidade de forma a aproximar-a da distribuição normal, apresentando valores em torno de zero, incluindo valores negativos. O problema é que uma matriz de distância não pode ter valores negativos. Assim a primeira opção tornou-se a mais viável, para uma primeira abordagem, principalmente tendo em conta que este não era o objetivo principal do projeto.

É de salientar ainda que a distância social não foi incluída por ter sido um dos aspetos menos desenvolvidos no TweeProfiles e por isso não foi tratado neste projeto.

Processo de Data Mining: Visto que o objetivo principal desta dissertação não passava por melhorar o processo de análise de dados desenvolvido no projeto TweeProfiles, mas sim estendê-lo para incorporar imagens na dimensão de conteúdo, optou-se por seguir o modelo desenvolvido no TweeProfiles. Assim o algoritmo DBSCAN foi o aplicado neste processo, em que o parâmetro do raio deste algoritmo foi de 10%, o mesmo utilizado no TweeProfiles. Apesar de após a normalização as matrizes das diferentes dimensões apresentarem a mesma

escala, a forma da distribuição das distâncias difere de dimensão para dimensão. Assim o valor do raio definido tem uma influência diferente em cada dimensão. No entanto, não temos conhecimento do impacto desta decisão nos resultados. Assim, seria importante um estudo mais aprofundado do impacto deste parâmetro em cada uma das dimensões e adaptá-lo de forma a que este seja tenha um comportamento similar em todas as dimensões. Aliás, esta questão insere-se num contexto mais geral de avaliação dos resultados do *clustering* no âmbito do projeto TweeProfiles e suas extensões. Até agora, esta questão teve sempre pouca prioridade. No entanto, à medida que a ferramenta se consolida, torna-se cada vez mais importante fazer um estudo empírico da qualidade dos *clusterings*, tanto em termos de medidas estatísticas como em termos de aplicações do conhecimento produzido.

É também importante referir que em paralelo ao desenvolvimento deste projeto de dissertação foi também desenvolvido a segunda versão do TweeProfiles por João Pereira do Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto. Esta nova versão tem o objetivo de descobrir padrões em tempo real, utilizando *stream* de dados. Isto torna ferramenta mais adaptada à realidade das redes sociais como o Twitter, em que os dados nos chegam a todo o instante. Assim, como trabalho futuro, um dos objetivos principais poderá passar pela implementação da mesma metodologia no Olhó-passarinho.

Visualização: O TweeProfiles foi desenvolvido utilizando a linguagem R em todo processo de recolha, tratamento e processamento dos dados e a sua aplicação web baseou-se na tecnologia Java. Visto que no caso do Olhó-passarinho foi necessário utilizar algoritmos de visão por computador, a linguagem Python foi a escolhida por permitir utilizar bibliotecas para este fim, e ao mesmo tempo adicionar bibliotecas para aceder à base de dados MongoDB, manipular os dados e ter também acesso a bibliotecas para utilização da linguagem R. Assim optou-se por também no desenvolvimento da aplicação web para a visualização dos resultados desenvolver uma nova ferramenta utilizando a linguagem Python. Para isso foi utilizado a *microframework* Flask que assenta nesta tecnologia e permite a simulação de um servidor em Python. Já no caso dos *widgets* para visualização foi seguido um padrão semelhante ao TweeProfiles, onde também foi adicionado um mapa e gráfico para respetivamente representar a distribuição espacial e temporal dos *clusters*.

No caso da secção para visualização do conteúdo visual, optou-se por uma matriz de imagens, visto esta ser uma das formas mais utilizadas em redes sociais para a visualização de conjuntos de imagens e permitir uma fácil comparação e navegação por entre as mesmas. Uma das possibilidades num trabalho futuro seria a sintetização de uma imagem capaz de representar um *cluster* de forma eficaz. No caso da visualização de uma imagem específica também seria interessante a visualização das N imagens mais próximas, onde N seria um número pré-definido.

Anexo A

Tabela da Base de Dados para Seleção de Tweets

```
1 create table if not exists IMAGENS (
2     id integer PRIMARY KEY AUTOINCREMENT,
3     id_tweet text ,
4     servico text ,
5     url text ,
6     tipo text ,
7     retweet text
8 );
```


Anexo A

Exemplo objeto JSON de um tweet

```
1 {
2     "_id": {
3         "$oid": "52c6d0f28ef20d397e42c54a"
4     },
5     "contributors": null,
6     "coordinates": null,
7     "created_at": "Tue Jun 18 17:08:15 +0000 2013",
8     "entities": {
9         "hashtags": [],
10        "symbols": [],
11        "urls": [
12            {
13                "display_url": "twitpic.com/cxvh38",
14                "expanded_url": "http://twitpic.com/cxvh38",
15                "indices": [
16                    104,
17                    126
18                ],
19                "url": "http://t.co/P7nwF6GOFc"
20            }
21        ],
22        "user_mentions": [
23            {
24                "id": 1181030630,
25                "id_str": "1181030630",
26                "indices": [
27                    3,
28                    14
29                ],
30                "name": "Bruna",
31                "screen_name": "holdmenian"
32            }
33        ]
34    },
35    "favorite_count": 0,
```

```

36     "favorited": false ,
37     "filter_level": "medium",
38     "geo": null ,
39     "id": 347038057221980162,
40     "id_str": "347038057221980162",
41     "in_reply_to_screen_name": null ,
42     "in_reply_to_status_id": null ,
43     "in_reply_to_status_id_str": null ,
44     "in_reply_to_user_id": null ,
45     "in_reply_to_user_id_str": null ,
46     "lang": "pt",
47     "metadata": {
48         "client": "192.168.102.195",
49         "emoticons": [],
50         "hashtags": [],
51         "language": "pt",
52         "mentions": [
53             "@holdmenian"
54         ],
55         "tokenized": "RT @holdmenian : \" O comercial da fiat ' Vem pra rua ' saiu do ar ap\u00f3f3s virar m\u00f3fasica tema dos protestos \" mas http://t.co/P7nwF6GOFc",
56         "topic": "protestos_brasil",
57         "urls": [
58             "http://t.co/P7nwF6GOFc"
59         ]
60     },
61     "place": null ,
62     "possibly_sensitive": false ,
63     "retweet_count": 0,
64     "retweeted": false ,
65     "retweeted_status": {
66         "contributors": null ,
67         "coordinates": null ,
68         "created_at": "Tue Jun 18 13:09:39 +0000 2013",
69         "entities": {
70             "hashtags": [],
71             "symbols": [],
72             "urls": [
73                 {
74                     "display_url": "twitpic.com/cxvh38",
75                     "expanded_url": "http://twitpic.com/cxvh38",
76                     "indices": [
77                         88,
78                         110
79                     ],
80                     "url": "http://t.co/P7nwF6GOFc"
81                 }
82             ],
83         }
84     },
85 
```

```
83         "user_mentions": []
84     },
85     "favorite_count": 7,
86     "favorited": false,
87     "geo": null,
88     "id": 346978014636146688,
89     "id_str": "346978014636146688",
90     "in_reply_to_screen_name": null,
91     "in_reply_to_status_id": null,
92     "in_reply_to_status_id_str": null,
93     "in_reply_to_user_id": null,
94     "in_reply_to_user_id_str": null,
95     "lang": "pt",
96     "place": null,
97     "possibly_sensitive": false,
98     "retweet_count": 98,
99     "retweeted": false,
100    "source": "<a href=\"http://messaging.nokia.com/\" rel=\"nofollow\">
Social by Nokia</a>",
101   "text": "\"O comercial da fiat 'Vem pra rua' saiu do ar ap\u00f3s virar
m\u00e1fica tema dos protestos\" mas http://t.co/P7nwF6GOFc",
102   "truncated": false,
103   "user": {
104     "contributors_enabled": false,
105     "created_at": "Fri Feb 15 03:50:19 +0000 2013",
106     "default_profile": false,
107     "default_profile_image": false,
108     "description": "make a wish \u221e",
109     "favourites_count": 12,
110     "follow_request_sent": null,
111     "followers_count": 934,
112     "following": null,
113     "friends_count": 718,
114     "geo_enabled": false,
115     "id": 1181030630,
116     "id_str": "1181030630",
117     "is_translator": false,
118     "lang": "en",
119     "listed_count": 0,
120     "location": "Charlie \u2665",
121     "name": "Bruna",
122     "notifications": null,
123     "profile_background_color": "FFFFFF",
124     "profile_background_image_url": "http://a0.twimg.com/
profile_background_images/344918034409728850/6
c945a8006333f3847476148aa68d7a0.png",
125     "profile_background_image_url_https": "https://si0.twimg.com/
profile_background_images/344918034409728850/6
c945a8006333f3847476148aa68d7a0.png",
```

```

126     "profile_background_tile": false ,
127     "profile_banner_url": "https://pbs.twimg.com/profile_banners
128 /1181030630/1371263818",
129     "profile_image_url": "http://a0.twimg.com/profile_images
130 /344513261580011975/f105a548f1b2198d325864dc5313e06b_normal.png",
131     "profile_image_url_https": "https://si0.twimg.com/profile_images
132 /344513261580011975/f105a548f1b2198d325864dc5313e06b_normal.png",
133     "profile_link_color": "B40B43",
134     "profile_sidebar_border_color": "FFFFFF",
135     "profile_sidebar_fill_color": "DDEEF6",
136     "profile_text_color": "333333",
137     "profile_use_background_image": true ,
138     "protected": false ,
139     "screen_name": "holdmenian",
140     "statuses_count": 9291,
141     "time_zone": "Mid-Atlantic",
142     "url": null ,
143     "utc_offset": -7200,
144     "verified": false
145   },
146   "source": "web",
147   "text": "RT @holdmenian: \"O comercial da fiat 'Vem pra rua' saiu do ar ap\u00f3s virar m\u00f3veis\u00fasica tema dos protestos\" mas http://t.co/P7nwF6GOFc",
148   "truncated": false ,
149   "user": {
150     "contributors_enabled": false ,
151     "created_at": "Wed Mar 02 16:18:50 +0000 2011",
152     "default_profile": false ,
153     "default_profile_image": false ,
154     "description": "... o fim virou come\u00e7o. E eu me permiti come\u00e7ar.",
155     "favourites_count": 6,
156     "follow_request_sent": null ,
157     "followers_count": 405,
158     "following": null ,
159     "friends_count": 360,
160     "geo_enabled": true ,
161     "id": 259792830,
162     "id_str": "259792830",
163     "is_translator": false ,
164     "lang": "pt",
165     "listed_count": 4,
166     "location": "Pau dos ferros - RN",
167     "name": "Fernando Cassio",
168     "notifications": null ,
169     "profile_background_color": "759AAD",
170     "profile_background_image_url": "http://a0.twimg.com/
profile_background_images/396888075/bg-meio.jpg",

```

```
169     "profile_background_image_url_https": "https://si0.twimg.com/
170     profile_background_images/396888075/bg-meio.jpg",
171     "profile_background_tile": true,
172     "profile_banner_url": "https://pbs.twimg.com/profile_banners/
173     /259792830/1361109491",
174     "profile_image_url": "http://a0.twimg.com/profile_images/3687937981/
175     bd4525bc45aa159a1c8dabcb5eab3ef4_normal.jpeg",
176     "profile_image_url_https": "https://si0.twimg.com/profile_images/
177     /3687937981/bd4525bc45aa159a1c8dabcb5eab3ef4_normal.jpeg",
178     "profile_link_color": "888E94",
179     "profile_sidebar_border_color": "D1D3DE",
180     "profile_sidebar_fill_color": "E2EEF0",
181     "profile_text_color": "131314",
182     "profile_use_background_image": true,
183     "protected": false,
184     "screen_name": "fernandocassio_",
185     "statuses_count": 5296,
186     "time_zone": "Santiago",
187     "url": "http://www.facebook.com/fernando.cassio.948",
188     "utc_offset": -14400,
189     "verified": false
190   }
191 }
```


Referências

- [1] Max Bramer. *Principles of Data Mining*. 2007.
- [2] TDS Cunha. *TweeProfiles: detection of spatio-temporal patterns on Twitter*. Tese de doutoramento, Faculdade de Engenharia da Universidade do Porto, 2013. URL: http://paginas.fe.up.pt/~ei08142/files/mieic_en.pdf.
- [3] Peng Wu, YM Ro, CS Won, e Yanglim Choi. Texture descriptors in MPEG-7. *Comput. Anal. Images* . . . , páginas 21–28, 2001.
- [4] M. Bober. MPEG-7 visual shape descriptors. *IEEE Trans. Circuits Syst. Video Technol.*, 11(6):716–719, Junho 2001. doi:10.1109/76.927426.
- [5] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *Int. J. Comput. Vis.*, 60(2):91–110, Novembro 2004. URL: <http://link.springer.com/10.1023/B:VISI.0000029664.99615.94>, doi:10.1023/B:VISI.0000029664.99615.94.
- [6] Herbert Bay, Tinne Tuytelaars, e Luc Van Gool. Surf: Speeded up robust features. *Comput. Vision–ECCV 2006*, 2006. URL: http://link.springer.com/chapter/10.1007/11744023_32.
- [7] Josef Sivic e Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. *Comput. Vision, 2003. Proceedings* . . . , (Iccv):2–9, 2003.
- [8] LM Bueno. Análise de descritores locais de imagens no contexto de detecção de semi-rélicas. 2011. URL: <http://www.dca.fee.unicamp.br/~dovalle/recod/works/lucasBueno2001mscDissertation.pdf>.
- [9] Jiawei Han, Micheline Kamber, e Jian Pei. *Data Mining, Second Edition: Concepts and Techniques*. 2006.
- [10] Google. Google chart - timeline. <https://developers.google.com/chart/interactive/docs/gallery/timeline>, 2014. Acedido a 30-06-2014.
- [11] Alexander Pak e Patrick Paroubek. Twitter as a Corpus for Sentiment Analysis and Opinion Mining. *LREC*, páginas 1320–1326, 2010. URL: [http://incctps.googlecode.com/svn/trunk/TPFinal/bibliografia/PakandParoubek\(2010\).TwitterasaCorpusforSentimentAnalysisandOpinionMining.pdf](http://incctps.googlecode.com/svn/trunk/TPFinal/bibliografia/PakandParoubek(2010).TwitterasaCorpusforSentimentAnalysisandOpinionMining.pdf).
- [12] Inc Twitter. Twitter. <https://twitter.com/>. Acedido a 16-07-2014.
- [13] Inc Yahoo. Flickr. <https://www.flickr.com/>. Acedido a 16-07-2014.

- [14] Xirong Li, Cees G.M. Snoek, e Marcel Worring. Learning tag relevance by neighbor voting for social image retrieval. Em *Proceeding 1st ACM Int. Conf. Multimed. Inf. Retr. - MIR '08*, página 180, New York, New York, USA, Outubro 2008. ACM Press. URL: <http://dl.acm.org/citation.cfm?id=1460096.1460126>, doi:10.1145/1460096.1460126.
- [15] Inc Facebook. Instagram. <http://instagram.com/>. Acedido a 16-07-2014.
- [16] Twitpic. <http://twitpic.com/>. Acedido a 16-07-2014.
- [17] Inc Facebook. Facebook. <https://www.facebook.com/>. Acedido a 16-07-2014.
- [18] LinkedIn Corporation. LinkedIn.
- [19] Matthew A Russell. *Mining the Social Web*, volume 54. 2011. doi:10.1081/E-ELIS3-120043522.
- [20] Inc MongoDB. MongoDB. <http://www.mongodb.org/>. Acedido a 16-07-2014.
- [21] M Boanjak e Eduardo Oliveira. TwitterEcho: a distributed focused crawler to support open research with twitter data. *Proc. 21st ...*, 2012.
- [22] Dr. Matthew A North. *Data Mining for the Masses*. Global Text Project, 2012.
- [23] Usama Fayyad, Gregory Piatetsky-shapiro, e Padhraic Smyth. From Data Mining to Knowledge Discovery in Databases. páginas 37–54, 1996.
- [24] Bing Liu. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data*. Springer, 2011.
- [25] Wei Wang, Jiong Yang, e Richard R. Muntz. STING: A Statistical Information Grid Approach to Spatial Data Mining. páginas 186–195, Agosto 1997.
- [26] Rakesh Agrawal, Johannes Gehrke, Dimitrios Gunopoulos, e Prabhakar Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. *ACM SIGMOD Rec.*, 27(2):94–105, Junho 1998. doi:10.1145/276305.276314.
- [27] J. Montavont e T. Noel. IEEE 802.11 Handovers Assisted by GPS Information. Em *IEEE Int. Conf. Wirel. Mob. Comput. Netw. Commun. 2006.*, páginas 166–172. IEEE, 2006. URL: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1696358>, doi:10.1109/WIMOB.2006.1696358.
- [28] David A. Forsyth e Jean Ponce. *Computer Vision: A Modern Approach*. Pearson Education, Limited, 2011.
- [29] Mark S. Nixon e Alberto S. Aguado. *Feature Extraction and Image Processing*. 2002.
- [30] BS Manjunath e JR Ohm. Color and texture descriptors. *Circuits Syst. ...*, 11(6):703–715, 2001.
- [31] Charilaos Christopoulos, Daniel Berg, e Athanassios Skodras. The colour in the upcoming MPEG-7 standard. *Invit. Pap. Eur. ...*, páginas 1–4, 2000.
- [32] Leszek Cieplinski. MPEG-7 Color Descriptors and Their Applications. 7:11–20, 2001.
- [33] Leszek Cieplinski (mitsubishi Electric Ite-vil. The MPEG-7 Color Descriptors Jens-Rainer Ohm (RWTH Aachen, Institute of Communications Engineering).

- [34] Vinay Modi. Color descriptors from compressed images. 2008.
- [35] H Shao, J Ji, Y Kang, e H Zhao. Application Research of Homogeneous Texture Descriptor in Content-Based Image Retrieval. *Inf. Eng.* ..., (2008515):2–5, 2009.
- [36] Kristen Gauman e Bastian Leibe. *Visual Object Recognition*. Morgan & Claypool Publishers, 2010. URL: <http://books.google.com/books?id=fYZgAQAAQBAJ&pgis=1>.
- [37] D.G. Lowe. Object recognition from local scale-invariant features. *Proc. Seventh IEEE Int. Conf. Comput. Vis.*, páginas 1150–1157 vol.2, 1999. URL: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=790410>, doi:10.1109/ICCV.1999.790410.
- [38] Josef Sivic e Andrew Zisserman. Video Google: Efficient visual search of videos. *Towar. Categ. Object Recognit.*, 4170:127–144, 2006. doi:10.1007/11957959_7.
- [39] Luo Juan e O Gwun. A comparison of sift, pca-sift and surf. *Int. J. Image Process.*, (4):143–152, 2009. URL: <http://www.cscjournals.org/csc/manuscript/Journals/IJIP/volume3/Issue4/IJIP-51.pdf>.
- [40] Michael Calonder, Vincent Lepetit, Christoph Strecha, e Pascal Fua. Brief: Binary robust independent elementary features. *Comput. Vision–ECCV 2010*, 2010. URL: http://link.springer.com/chapter/10.1007/978-3-642-15561-1_56.
- [41] Ethan Rublee e Vincent Rabaud. ORB: an efficient alternative to SIFT or SURF. *Comput. Vis. (ICCV ...*, páginas 2564–2571, Novembro 2011. URL: <http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6126544> http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6126544, doi:10.1109/ICCV.2011.6126544.
- [42] Alexandre Alahi, Raphael Ortiz, e Pierre Vandergheynst. Freak: Fast retina keypoint. *Comput. Vis. ...*, 2012. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6247715.
- [43] D. Nister e H. Stewenius. Scalable recognition with a vocabulary tree. ... *Vis. Pattern Recognition, 2006 ...*, 2, 2006. URL: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1641018.
- [44] Jan Erik Solem. Programming Computer Vision with Python. 2012.
- [45] A. Vedaldi e B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms. <http://www.vlfeat.org/>, 2008. Acedido a 30-06-2014.
- [46] SQLite Copyright. Sqlite. <http://www.sqlite.org>. Acedido a 07-07-2014.
- [47] Oracle Copyright. Mysql. <http://www.mysql.com>. Acedido a 07-07-2014.
- [48] The PostgreSQL Global Development Group. Postgresql. <http://www.postgresql.org>, 1996. Acedido a 07-07-2014.
- [49] Armin Ronacher. Flask, web development, one drop at a time. <http://flask.pocoo.org/>. Acedido a 08-07-2014.
- [50] Google. Api javascript do google maps v3. <https://developers.google.com/maps/documentation/javascript/>, 2013. Acedido a 30-06-2014.