Connecting Dots

Bolseiro:
Carla Abreu

Orientadores:
Prof. Eugénio Oliveira
Jorge Teixeira
Data do projeto:
15 de Outubro 2014, 14 Abril de 2015

Projeto desenvolvido no âmbito do SAPO Labs
na Universidade do Porto

14 de Abril de 2015

Resumo

Neste documento é apresentado o relatório final relativo à terceira bolsa do projeto "Connecting the Dots"desenvolvido no âmbito do Laboratório SAPO/ U.Porto.

O projeto "Connecting the Dots"tem como objetivo estabelecer automaticamente a ligação de notícias portuguesas que fazem parte da mesma história e que aparecem em diferentes fontes noticiosas e/ou em diferentes momentos no tempo. Um exemplo concreto do objetivo deste trabalho é o caso do desaparecimento do avião da Malaysia Airlines sobre o qual surgiram várias notícias em diferentes momentos no tempo. Uma parte da história sobre este acontecimento pode ser apresentada, cronologicamente, pelos seguintes títulos: "Avião da Malaysia Airlines desaparece com 239 pessoas a bordo"; "Vietname diz que avião desaparecido da Malaysia Airlines caiu ao mar"; "Buscas pelo voo MH370 podem custar US$ 250 milhões, dizem autoridades"e "Profundidade dificulta missão de minissubmarino de localizar avião".

Este relatório visa documentar o trabalho desenvolvido. É de notar que este relatório surge em sequência de dois relatórios anteriores onde se definiu a arquitetura e implementação do sistema.

Conteúdo

Capítulo 1

Trabalho Realizado

Diariamente, são publicadas milhares de notícias em formato digital. Em consequência, o número de notícias é cada vez maior, sendo a tarefa de acesso aos dados uma tarefa cada vez mais árdua. Devido a este problema é essencial a construção de ferramentas que permitam ao utilizador aceder aos dados de uma forma útil e eficiente. A abordagem utilizada para solucionar este problema é a categorização. De uma forma genérica, a categorização consiste na atribuição de categorias aos documentos, dentro de um conjunto de categorias conhecidas. Esta tarefa é frequentemente solucionada como um problema de aprendizagem supervisionada.

Existem vários tipos de classificação como: a linear e a hierárquica. Existem também várias tipologias: *single* - atribuição de apenas uma categoria; e, *multi label* associação de múltiplas categorias.

Apesar da classificação linear ser a abordagem mais utilizada, a mesma apresenta inconvenientes, como, por exemplo o de não considerar a possibilidade das categorias terem alguma ligação entre si. Por sua vez a classificação hierárquica, tem como objetivo organizar as categorias numa estrutura de especificidade crescente. Esta forma de representação tem várias vantagens, como: a melhoria da usabilidade do sistema, a melhoria da taxa de sucesso das pesquisas e ainda a melhoria da satisfação do utilizador.

Para que seja possível a classificação hierárquica é necessário a definição de hierarquia. Várias abordagens têm sido utilizadas nesta área como a construção manual e construção automática. Estudos prévios relacionados com a classificação hierárquica mostram que pouco tem sido feito para a geração hierárquica de tópicos, sendo que maioritariamente os trabalhos assentam numa estrutura realizada manualmente. A estrutura manual porém apresenta algumas desvantagens: tem que ser construída por um especialista que perceba a importância dos tópicos nas notícias, é uma tarefa custosa e estática. Por sua vez a geração automática de hierarquias, visa obter os tópicos existentes e por correlações de dados verificar qual a sua posição num sistema hierárquico, podendo considerar-se esta abordagem menos custosa e mais dinâmica que a anterior.

A ordem do plano de trabalhos sofreu algumas alterações (Anexo A): Mês 1 - foi efetuada a leitura do trabalho relacionado com a geração de hierarquias e métodos de classificação; Mês 2 e 3 - foi elaborado um primeiro protótipo para a geração de hierarquias, este primeiro protótipo resultou da análise de trabalhos na área, o sistema implementado sofreu as alterações necessárias para se adaptar

ao domínio pretendido; Mês 4 - destinou-se à escrita e submissão de um artigo científico e por fim o Mês 5 e 6 - foram efetuadas alterações à interface do sistema de modo a melhorar a usabilidade do utilizador na pesquisa de cadeias noticiosas (não foi possível, no entanto, fazer a integração da hierarquia nesta fase pois apesar de se ter gerado a hierarquia não foi efetuada a tarefa de classificação).

1.1 Mês 1: Leitura de trabalho relacionado com a hierarquização

No primeiro mês de bolsa foi efectuada a leitura de artigos relacionados com os tópicos: classificação hierárquica de documentos e geração automática de hierarquias. Também sobre estes tópicos foram estudadas as abordagens propostas e os métodos de avaliação utilizados (Anexo B).

1.2 Mês 2/3: Hierarquização das categorias das notícias

1.2.1 Construção da Baseline

De acordo com o plano de trabalhos, após o estudo das abordagens propostas para a geração automática de hierarquias seria necessário a implementação de um primeiro modelo. A *baseline* desenvolvida para a geração automática de hierarquias foi baseada na abordagem proposta pelos autores "Sanderson and Croft"designada de *subsumption*. Esta abordagem probabilística tem em consideração a co-ocorrência de termos. De acordo com esta abordagem, existe uma relação hierárquica do termo x para o termo y, se P(x/y) >= 0.8 e P(y/x) < 1.

1.2.2 Adaptação da Baseline

Termos Utilizados

O conjunto de dados é composto por notícias portuguesas disponibilizadas *online*. Os tópicos utilizados para a geração automática da hierarquia foram obtidos de elementos das notícias, mas concretamente: das *tags* atribuídas pelos jornalistas e pela informação proveniente do URL.

Processo Normalização

Os termos utilizados foram sujeitos a um processo de normalização. Neste processo, foram removidas todas as expressões que não representavam níveis hierárquicos, como por exemplo: "notícias"; "destaques"; "detalhe"encontradas essencialmente associadas às *tags* e "www.", ."pt", "destaques", "paginainicial"encontradas essencialmente no URL da notícia. Após descartadas as expressões sem relevância todos os termos encontrados foram convertidos para *lower-case*, foram também substituídos todos os carateres acentuados e cedilhados.

A estrutura de agrupamento definida na primeira bolsa, foi útil para a deteção e eliminação de termos duplicados. O fluxo de processamento utilizado na deteção e eliminação de termos duplicados, consistiu na:

- extração da informação proveniente do *URL* e das *tags* das notícias;
- normalização dos termos extraídos;
- associação dos termos extraídos às notícias;
- associação dos termos extraídos aos agrupamentos;
- para cada agrupamento:
  - comparação dos termos utilizando o algoritmo de edição de distância Levensthein;
  - verificação do grau de semelhança;
  - eliminação do termo cujo o grau de semelhança fosse superior ao estabelecido.

Foi efetuada uma análise manual dos resultados obtidos pela comparação dos termos dentro de um agrupamento, dessa análise foi possível concluir que quando dois termos tem uma distância inferior a 30% então estes referem-se a um mesmo elemento. Um exemplo de um par de termos duplicados é dado por "Formula 1"e "formula1".

De foma a não se perder a informação, os elementos duplicados foram armazenados numa estrutura de dicionário.

Elaboração da Hierarquia

Após a obtenção de todos os termos relevantes existente no conjunto de dados é possível passar-se à geração automática da hierarquia. Para a realização desta tarefa, os dados foram armazendados numa estrutura de matriz, em que as linhas representavam os agrupamentos e as colunas os diferentes termos existentes. É de notar que partimos do pressuposto que um termo só entra na composição da matriz se tiver ocorrido mais do que um determinado número de vezes, neste caso em particular, pela observação do conjunto de dados este número assumiu um valor de 15.

Como já foi referido anteriormente, a geração automática da hierarquia teve por base uma teoria probablística. Nesta abordagem é calculada a probabilidade de dois termos coocorrerem. Após uma análise aos resultados obtidos pelo cálculo da co-ocorrência, foi possível observar que quando: P(x/y) >= 0.9 e P(y/x) >= 0.9, os termos são equivalentes (ou seja, x é igual a y). Um exemplo disso, situado num mesmo contexto, é dado pelas expressões: Futebol > [inglaterra, mundial'2014 inglaterra]; Opinião > [leitores, escrevem os leitores]. Para além do dicionário de duplicados referido foi também elaborada uma lista de termos equivalentes.

Dentro do domínio deste trabalho, verificou-se que quando a probabilidade de um termo coocorrer é superior a 35% significa que estamos na presença de uma hierarquia, e que o termo com o maior nível de probabilidade corresponde a um nó folha. Exemplos:

- (u'mario figueiredo', u'liga clubes') [0.6666666666666666, 0.2857142857142857]
- (u'futebol', u'sporting covilha') [0.0013463480309660047, 0.5]

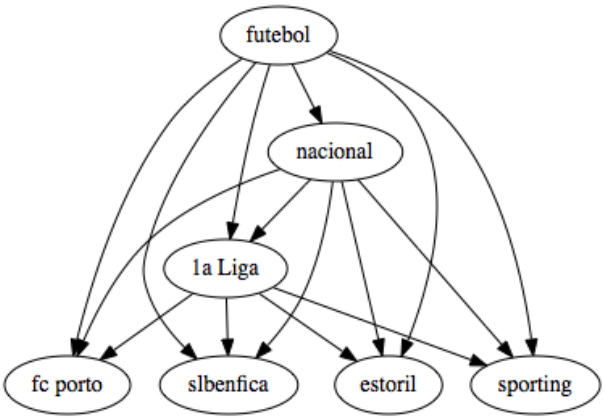


Figura 1.1: Geração automática de hierarquia

Simplificação da Hierarquia

Um exemplo do resultado da geração automática da hierarquia pode ser visualizado na Figura 1.1. Após analisarmos a imagem é percetível a existência de ligações redundantes. Por exemplo, com os termos *futebol*, *nacional* e *1a liga*, são formadas as seguintes ligações:

- *futebol* $->$ *nacional*
- *futebol* $->$ *1a liga*
- *nacional* $->$ *1a liga*

A nível hierárquico a ligação *futebol* $->$ *1a liga* é redundante pois não adiciona informação à hierarquia.

O processo de simplificação da estrutura hierárquica compreendeu a eliminação de ligações e nós redundantes. A nível dos nós, a eliminação deu-se em dois níveis:

- Profundidade: quando o nó filho está incluído ou é similar ao nó pai;
- Largura: quando dois nós ao mesmo nível têm os mesmos nós filhos.

Para o exemplo apresentado anteriormente, o resultado final seria o apresentado pela Figura 1.2.

Enriquecimento da estrutura Hierárquica

De forma a enriquecer a estrutura hierárquica obtida foi estudada a possibilidade de adição de nós "esquecidos". Os nós esquecidos, são todos aqueles, que ocorram com uma frequência reduzida, e tem ligações aos nós que estão presentes na hierarquia. Os nós "esquecidos"podem estar em quatro níveis diferentes da hierarquia, considerando a e b como nós presentes na hierarquia, com a seguinte ligação hierárquica a $->$ b (Figura 1.3), temos:

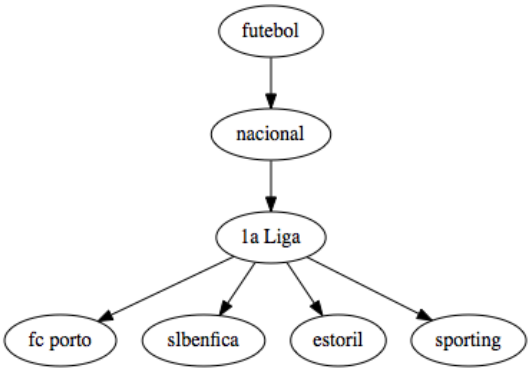


Figura 1.2: Geração automática de hierarquia - após a simplificação

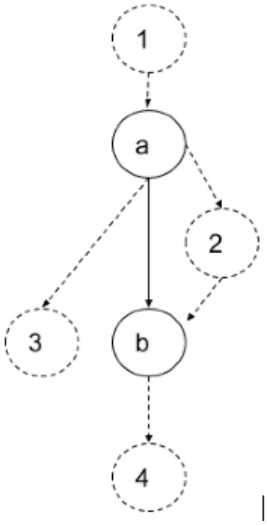


Figura 1.3: Enriquecimento da estrutura hierárquica

1. O nó é hierarquicamente superior a a e a b;
2. O nó situa-se entre a ligação a $->$ b;
3. O nó é uma derivação do nó a e encontra-se ao mesmo nível hierarquico que b;
4. O nó descende do nó b.

A pesquisa destes nós foi feita na estrutura de matriz definida anteriormente com a ocorrência de elementos por documento. Aos nós extraídos foram aplicados algoritmos de classificação de forma a perceber se estes deveriam ou não pertencer à hierarquia. Os nós do tipo 1 e 2 têm uma fraca frequência pelo que não foi possível encontrar uma amostra considerável destes nós. Em relação aos nós do tipo 3, os resultados não foram os melhores, os resultados rondaram os 0%. Os nós do tipo 4 apresentaram a seguinte precisão, através da utilização da validação *k-cross-fold validation*: 76.1% pelo método *decision tree*, 75.4% *random forest* e 71.8% *svc*.

1.2.3 Considerações

A hierarquia gerada automaticamente não foi exposta a nenhum método de avaliação. No entanto, foram feitos vários refinamentos como: a simplificação da hierarquia e o enriquecimento da estrutura hierárquica com base em análises efetuadas manualmente. A nível do resultado global da hierarquia:

- O resultado da hierarquia depende sempre do contexto das notícias: Mundo $->$ Europa $->$ Alemanha e Desporto $->$ Futebol $->$ Alemanha. Contextualizando, no primeiro caso, o resultado significa que a Alemanha faz parte da Europa que por sua vez faz parte do mundo enquanto que no segundo significa que o futebol é um desporto e que por sua vez a Alemanha tem uma equipa de futebol.
- Nós iguais que aparecem em categorias distintas podem ter significados diferentes.
- O aparecimento dos nós depende da importância de um tópico nas notícias;
- A representação hierárquica elaborada resultante do domínio das notícias é contextualizada pelas mesmas. Por exemplo, considerando a seguinte hierarquia: Mundo $->$ Israel $->$ Palestina, pelo senso comum seria considerada como errada, pois apesar de Israel e a Palestina fazerem parte do Mundo, a Palestina não se encontra em Israel. No entanto, contextualizando-nos pelos acontecimentos, podemos perceber que esta ligação existe, referindo-se ao confito israelo-palestino. Esta hierarquia surge, pois, a maior parte das notícias relacionadas com a Palestina referem-se os confrontos com Israel.

1.2.4 Repositório e Estrutura do Código

O código associado a esta parte do projeto encontra-se no gitLab no projeto ConnectingTheDots na pasta ConnectingDotsHierarchy. O modo de execução

deste componente encontra-se descrito no Anexo H (este módulo requer a utilização de uma base de dados mySQL).

Os *scripts* principais do sistema são:

- createHierarchy.py: Geração automática da hierarquia;
- statistics.py: Cálculo da probabilidade de coocorrência de tópicos;
- graphpath.py: Estabelecimento de ligações entre elementos;
- duplicates.py: Eliminação de dados duplicados.

1.3 Mês 4: Escrita e submissão de um artigo científico

Durante o quarto mês de bolsa foi escrito o artigo "ENCADEAr: ENCADEmaneto Automático de Notícias"e submetido para a revista "Linguistica, Informatica e Traducao: Mundos que se Cruzam"da Osla (Anexo C).

1.4 Mês 5-6: Melhoramento da interface web

Durante o quinto e sexto mês de bolsa foi realizado uma reestruturação aos dados e à interface anteriormente implementada.

1.4.1 Interface

De modo a tornar percetível as histórias noticiosas criadas pelo sistema e de forma a melhorar a usabilidade do utilizador foi elaborada uma nova interface. A proposta desta nova interface encontra-se no Anexo D e a versão final implementada encontra-se no Anexo E.

1.4.2 Estrutura do Código

O módulo da interface é um módulo independente. Este módulo requer apenas a conexão com a base de dados Neo4j.

ConnectingDotsInterface/InterfaceResources - Pasta com os recursos associados à interface.

- website.py: *script python* para execução da *interface*;
- graphrepresentation.json: estrutura *json* com o gráfico de cadeias a serem visualizadas na *interface*;
- templates: pasta com os *templates* necessários.
  - index.html: página principal - visualização das cadeias noticiosas;
  - about.html: página com a descrição do projeto.
- static: pasta com as imagens necessárias para a *interface*.

ConnectingDotsInterface/FlaskResources - Pasta com os recursos associados ao Flask.

- flaskIni.py: *script* de iniciação do *flask*, responsável também pela receção de pedidos.
- bdConnectNEOymw.py: *script* de acesso à base de dados neo4j.
- novaInterface.py: *script* responsável pela adaptação dos dados recebidos à interface.

1.4.3 Repositório

O código da interface encontra-se no gitLab na pasta designada Connecting-Dots Interface. De forma a executar a interface é necessário executar os passos mencionados no Anexo F.

1.5 Overview

1.5.1 Connecting Dots

Ao longo do tempo houve a necessidade de se fazer algumas correções/ melhorias ao sistema desenvolvido. Apesar das alterações necessárias a estrutura do código manteve-se inalterada (esta encontra-se descrita no primeiro relatório da bolsa). A documentação deste componente encontra-se disponível no Anexo I.

O código deste projeto encontra-se no gitLab no diretório ConnectingDots Encadear. Os requisitos necessários para o pleno funcionamento do sistema são: base de dados com a informação das notícias; uma base de dados para o armazenamento e acesso aos dados relativos ao encadeamento de notícias (cuja estrutura se encontra descrita em 1.5.3); a instalação do tree-tagger e indicação nos *scripts* do diretório onde este se encontra (mudar valor da variável TAGDIR nos *scripts* similarprocess.py e similarprocess2.py).

Adicionalmente, foi desenvolvido um *script* que permite a execução diária do sistema (scriptdiario.sh), dependendo da localização do ficheiro poderá ser necessário a alteração da primeira linha do código.

1.5.2 APIS

Durante a segunda bolsa foram desenvolvidas API's para os módulos principais do sistema. Estas API's encontram-se no gitLab no projeto ConnectingTheDots na pasta API. A introdução às mesmas encontra-se no 2º Relatório, e o modo de execução esta descrito no Anexo G.

1.5.3 Base de Dados

A base de dados requerida para a hierarquização e para a construção de cadeias noticiosas é composta pelas seguintes tabelas:

CDHierarchyCategories : Tópicos por categoria.

ID - int(11) - Identificador do par tópico por categoria;

category - varchar(512) - Categoria atribuídas pelos jornalistas às notícias;

categories - varchar(512) - Tópico a aparecer na hierarquia;

Status - int(11) - estado do par.

CDarchCategoryFinal : Categoria de uma ligação.

IDCluster1 - int(11) - identificador do agrupamento temporalmente mais antigo;

IDCluster2 - int(11) - identificador do agrupamento temporalmente mais recente;

category - varchar(512) - categoria das notícias associadas à ligação.

CDarchCategoryValuesFinal : Categoria e valores obtidos pela ligação de dois elementos.

ID - int(11) - Identificador da ligação.

IDCluster1 - int(11) - identificador do agrupamento temporalmente mais antigo;

IDCluster2 - int(11) - identificador do agrupamento temporalmente mais recente;

category - varchar(512)- categoria das notícias associadas à ligação;

S1 - float - Valor da comparação dos termos chave simples;

S2 - float - Valor da comparação dos termos chave compostos;

S3 - float - Valor da comparação das entidades;

S4 - float - Valor da comparação das personalidades;

VAL - int(11) - Indica se a ligação foi ou não processada;

Status - int(11) - Estado da ligação.

CDcluster Definição dos agrupamentos.

ID - int(11) - Identificador do agrupamento;

IdNews - varchar(512) - Lista com os identificadores das notícias;

Status - int(11) - Estado do agrupamento;

NumberElements - int(11) - Número de notícias associadas ao agrupamento;

PrincipalNews - int(11) - Identificador da notícia principal;

Date - datetime - Data inicial do agrupamento;

OldID - int(11) - Identificador do agrupamento noutra base de dados;

idmdt - int(11) - Identificador da notícia principal (pela máquina do tempo);

idnewsmdt - varchar(512) - Lista com os identificadores de notícias (pela máquina do tempo);

oMDT - int(11) - Se os campos já estão de acordo com a máquina do tempo;

CDkeyword Palavras-chave associadas às notícias.

ID - int(11) - Identificador da notícia;

expSimple - varchar(512) - Lista com os termos chave simples;

expComp - varchar(512) - Lista com os termos chave compostos;

entity - varchar(512) - Lista com as entidades;

bMDT - int(11) - Se o identificador da notícia já está de acordo com a máquina do tempo.

CDkeywordCluster Palavras-chave associadas aos agrupamentos

ID - int(11) - Identificador do agrupamento;

expSimple - varchar(512) - Lista de termos chave simples;

expComp - varchar(512) - Lista de termos chave compostos;

entity - varchar(512) - Lista de entidades;

size - int(11) - Número de notícias associadas ao agrupamento;

dateBegin - datetime - Data da ocorrência da primeira notícia;

dateEnd - datetime - Data da ocorrência da última notícia.

CDsimilar Similaridade entre pares de notícias.

ID - int(11) - Identificador da comparação;

EXP - varchar(255) - Identificador da experiência;

Idnews1 - int(11) - Identificador da notícia 1;

Idnews2 - int(11) - Identificador da notícia 2;

Path - varchar(255) - Caminho utilizado na determinação da similaridade;

State - varchar(255) - Estado da comparação;

ST - float - Semelhança do título;

SB - float - Semelhança do primeiro parágrafo;

SC - float - Semelhança do conteúdo;

CDtagCluster Associação de *tags*, imagens e entidades ao agrupamento.

ID - int(11) - Identificador do agrupamento;

tag - varchar(512) - lista de *tags* Associadas ao agrupamento;

category - varchar(512) - Categoria do agrupamento;

Status - int(11) - Estado do agrupamento;

StatusArch - int(11) - Estado do agrupamento;

StatusArch1 - int(11) - Estado do agrupamento;

StatusArch2 - int(11) - Estado do agrupamento;

StatusArch3 - int(11) - Estado do agrupamento;

bImage - int(11) - Estado da imagem;

urlImage - varchar(512) - *Url* da imagem;

bEntity - int(11) - Estado da entidade;

listEntity - varchar(512) - Lista de entidades.

Capítulo 2

Divulgação

No âmbito deste projeto foi submetido e publicado um artigo na revista "Linguistica, Informatica e Traducao: Mundos que se Cruzam"da Osla.

Apêndice A

Plano de trabalhos

"Connecting the Dots Between News"

Plano de actividades

Este projeto visa dar continuidade ao trabalho na bolsa de investigação "Connecting the Dots Between News", desenvolvido nos últimos 12 meses. Em particular, explorar e expandir a hierarquização automática das categorias das notícias.

Plano de trabalhos

[Mês 1-2] – Hierarquização das categorias das notícias (baseline)

- Estudo de trabalho relacionado
- Aplicação de abordagens state-of-the-art (baseline)
- Avaliação das abordagens implementadas

Milestone 1: Implementação do método baseline

[Mês 3-4] – Hierarquização das categorias das notícias

- Experimentação de novas abordagens de hierarquização de categorias
- Avaliação das abordagens implementadas

Milestone 2: Implementação do novo método

[Mês 5] – Integração e melhoramento do interface web

- Integração dos métodos desenvolvidos no interface web

Milestone 3: Melhoramento de um interface web

[Mês 6] – Escrita e submissão de um artigo científico

Milestone 3: Publicação de um artigo científico

_____

Bolseira (Carla Abreu)

_____

Orientador (Prof. Eugénio Oliveira)

Porto, 19 de Setembro de 2014

Apêndice B

Classificação Hierárquica

Classificação Hierárquica

Identificador artigo: Nome artigo; Autores

1. A tutorial on hierarchical classification with applications in bioinformatics; Alex Freitas e André Carvalho
2.Hierarchical Text Classification and Evaluation; Aixin Sun and Ee-Peng Lim
3. Learning Classifiers Using Hierarchically Structured Class Taxonomies: Feihong Wu, Jun Zhang and Vasant Honavar
4.Hierarchical Document Classification Using Automatically Generated Hierarchy - Tao Li; Shenghuo Zhu: li2007hierarchical
5.Deriving concept hierarchies from text Mark - Mark Sanderson, Bruce Croft; sanderson1999deriving
6.Automatic Creation of Hierarchical Faceted Metadata Structures: Emilia Stoica, Marti Hearst and Megan Richards
7. Machine Learning in Automated Text Categorization, Fabrizio Sebastiani
8. Hierachical topic detection in large digital news archives
9.Dynamic Hierarchical Compact Clustering Algorithm - Gil-Garcia, Badia-Contelles, Pons-Porrata
10. Learning Hierarchical Classifiers with class taxonomies - Feihong Wu, Jun Zhang
12. Hierarchical multi-classification - Blockeel, Bruynooghe, Dzeroski, Ramon, Struyf

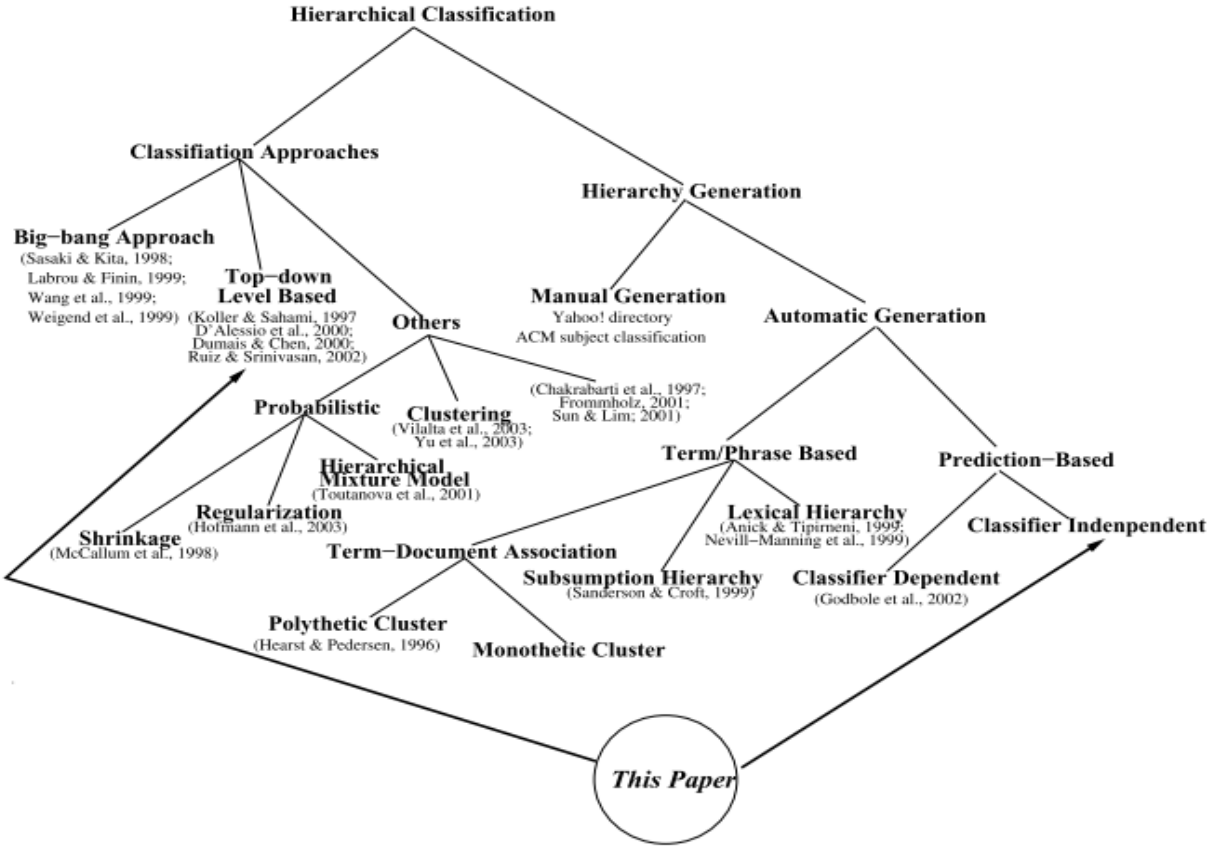


Figure 6: Summary of Related Work on Hierarchical Classification

Classificação Hierárquica
Métodos de avaliação mencionados e/ou utilizados nos artigos em questão

Artigos	Métodos
1: 10	***Uniform misclassification costs:*** calcula o erro cometido em cada uma das diferentes categorias; não é o método ideal para avaliação de problemas hierárquicos porque ignora o facto de algumas classes estarem próximas na hierarquia; Métrica de avaliação desaconselhada em sistemas multi-label (artigo 10).
1, 2, 12	***Distance-based misclassification costs:*** este cálculo é proporcional à distância existente entre a classe atribuída à notícia e a classe a que esta de facto deveria pertencer. No artigo 12, são enunciadas várias formas de distância: menor caminho; menor caminho pesado; peso dos empates.
1	***Semantic-based misclassification costs:*** é independente da distância entre classes na hierarquia. A medida de similaridade é baseada na noção 'semântica' das classes.
1, 12	***Hierarchical misclassification cost matrix:*** especificação explicita de cada erro cometido (utilizada em problemas de classificação linear)
1	***Leaf-node prediction:*** consiste em fazer a avaliação olhando só para os nós folha da hierarquia.
2; 4; 7	***Precision/Recall:*** *flat classification (*em 2 provam que não é uma métrica adaptável à classificação hierárquica): também no artigo 4 dizem que não é uma métrica apta para problemas hierárquicos.
2	***Measures based on category similarity:*** está relacionada com a forma hierárquica e com a distância para chegar ao nó correto.
3: 10	***Label Mapping:*** também tem em conta a distância à qual o nó deveria ter sido atribuído e ao qual realmente foi (sub árvores)
6	**User Study**
8	***Minimal cost metric proposed by Alla*****n** tem em consideração a distância; mas também os falsos alarmes e os esquecimentos
4, 12	**Accuracy:** não são permitidas substituições o modelo funciona como omissões ou inclusões

Classificação Hierárquica

Métodos

Artigos	Abordagem
2: 4	***ML: Top-down level-based classification method:*** um ou mais classificadores são construídos em cada nível hierárquico. Cada classificador funciona como um classificador linear. O grande problema desta abordagem é que se inicialmente for errada a classificação, nunca mais recuperamos o nó
2	***ML: Big-bag approach:*** temos apenas um classificador que tem logo a função de dizer a quais as categorias a que um documento deve estar associado os testes efetuados a esta abordagem baseia-se em observações empíricas
7	**Rules:** utilização de regras para atribuição das categorias (utilizado nos anos 80)

Tipo

7	***Single-label text categorization***
7	***Multi-label text categorization***

Construção automática de hierarquias

4	***Linear Discriminat Projection Approach:*** 1º passo consiste na medição da distância entre categorias similares. Duas categorias são similares se estiverem próximas no espaço transformado.
4: 8	***Hierarchical Agglomerative Clustering (HAC):*** responsável por gerar automaticamente a hierarquização dos tópicos dado um conjunto de classes linear.
5	***Subsumption:*** está relacionado com a probabilidade condicional entre termos.
6	***Castanet Algorithm***
8	***Divisive clustering works top-down:*** o conjunto de dados é tratado como um agrupamento. A ideia é fazer-se divisões do agrupamento até atingirmos o critério de paragem.

Tipos de hierarquia

1: 2	***Árvore / Virtual Category Tree*** (atribuição feita apenas ás folhas) **/ *Category Tree*** (aqui é permitida a atribuição de categorias a nós intermédios)
1; 8	***Grafo Direccional Acíclico (DAG):*** diferença para a árvore é que este último um nó pode ter mais de que um nó pai. **/ *Virtual Directed Acyclic category graph*** (atribuição às folhas) **/ *Directed Acyclic category graph*** (todos os níveis)

Apêndice C

Artigo

Simões, Barreiro, Santos, Sousa-Silva & Tagnin (eds.) *Linguística, Informática e Tradução: Mundos que se Cruzam*, Oslo Studies in Language 7(1), 2015. 153–181. (ISSN 1890-9639 / ISBN 978-82-91398-12-9)
http://www.journals.uio.no/osla

ENCADEAr
ENCADEAMENTO AUTOMÁTICO DE NOTÍCIAS

CARLA ABREU, JORGE TEIXEIRA E EUGÉNIO OLIVEIRA

ABSTRACT

This work aims at defining and evaluating different techniques to automatically build temporal news sequences. The approach proposed is composed by three steps: (i) near duplicate documents detention; (ii) keywords extraction; (iii) news sequences creation. This approach is based on: Natural Language Processing, Information Extraction, Name Entity Recognition and supervised learning algorithms. The proposed methodology got a precision of 93.1% for news chains sequences creation.

[1] INTRODUÇÃO

Diariamente são publicadas grandes quantidades de notícias *online*, o que pode conduzir a uma sobrecarga de informação para o leitor. Para estar informado e atualizado de um determinado acontecimento, o leitor depara-se com um vasto conjunto de artigos noticiosos, artigos esses que, em muitos casos, descrevem um mesmo evento, podendo apresentar apenas pequenas variações textuais. A situação agrava-se quando o leitor pretende saber mais detalhes sobre uma dada história ou sequência de eventos. Um exemplo concreto é o desaparecimento do avião da *Malaysia Airlines* a 8 de março de 2014. Considerando o dia 6 de outubro de 2014 a pergunta (*query*) *"avião Malaysia"* pesquisada no *Google News (news.google.pt)* retorna uma lista com mais de 50 notícias relacionadas. Dessas notícias retiramos a informação de que as buscas pelo avião foram retomadas. Como é possível observar pelos seguintes títulos: *Retomadas buscas pelo avião da Malaysia Airlines* (Renascença, 06/10/2014) e *Recomeçam as buscas pelo avião desaparecido da Malaysia Airlines* (Jornal de Notícias, 06/10/2014) o evento noticiado é o mesmo, mas pelo facto das notícias serem provenientes de fontes noticiosas diferentes apresentam variações textuais.

Este problema da sobrecarga de informação agrava-se quando o leitor quer perceber a história do desaparecimento do avião como um todo, e informar-se sobre todos os eventos que ocorreram relativamente a este acontecimento. A pergunta (*query*) *"desaparecimento Malaysia Airlines"* sem delimitações temporais ao Google News apresenta mais de 4.500 resultados. Neste conjunto de resultados torna-se complicado ou até mesmo humanamente impossível não só a deteção de todos os eventos como apenas os mais relevantes para a história. Por conseguinte,

o leitor não consegue ter a perceção de toda a história, descrita em mais de 4.500 notícias diferentes.

O objetivo deste trabalho é colmatar este problema: automaticamente detetar e agrupar notícias similares e automaticamente criar histórias a partir de notícias relacionadas temporalmente. Proporciona-se deste modo ao leitor uma nova forma de navegação entre eventos relativos a um mesmo acontecimento.

Pretendemos, numa primeira fase, detetar e agrupar notícias duplicadas (ver Figura 1). Utilizamos métodos de processamento de linguagem natural, algoritmos de medição de distância entre *strings*[1] (para o cálculo da proximidade entre notícias) e algoritmos supervisionados de aprendizagem automática (para a determinação da similaridade entre notícias). Numa segunda fase (ver Figura 2), com vista à formação automática de cadeias noticiosas, extraímos termos relevantes das notícias como por exemplo o tópico principal da notícia, as entidades, os locais e os nomes das personalidades; e ligamos os grupos de notícias pela medição da distância entre os mesmos. Utilizamos algoritmos de aprendizagem supervisionada para ligar notícias de forma sequencial para criar uma história temporalmente lógica e contextualizada.

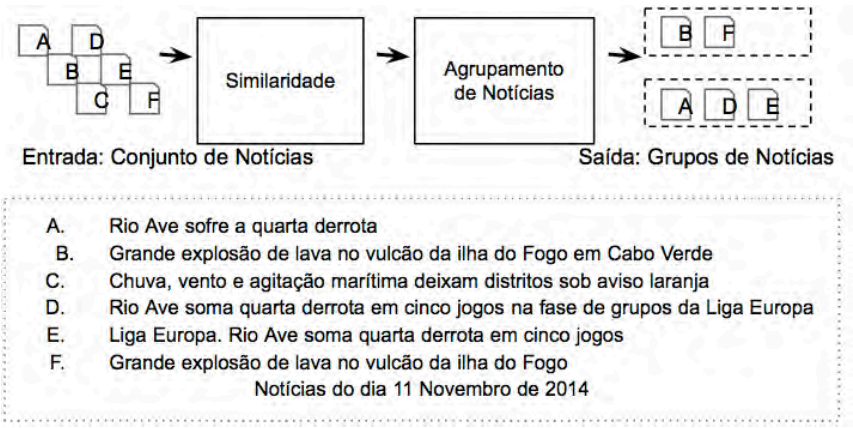


FIGURA 1: Deteção e agrupamento de notícias similares

Este artigo encontra-se organizado da seguinte forma: na secção [2] apresentaremos o essencial sobre trabalhos relacionados. Na secção [3] vamos expor detalhadamente todos os passos da metodologia aplicada. Na secção [4] vamos enunciar os recursos linguísticos utilizados. Seguem-se a descrição das experiências realizadas (secção [5]) e a apresentação e discussão dos resultados na secção [6]. Na secção [7] é apresentada a interface gráfica desenvolvida como prova de conceito. Por fim são apresentadas as conclusões e o trabalho futuro na secção [8].

[1] Sequência de carateres.

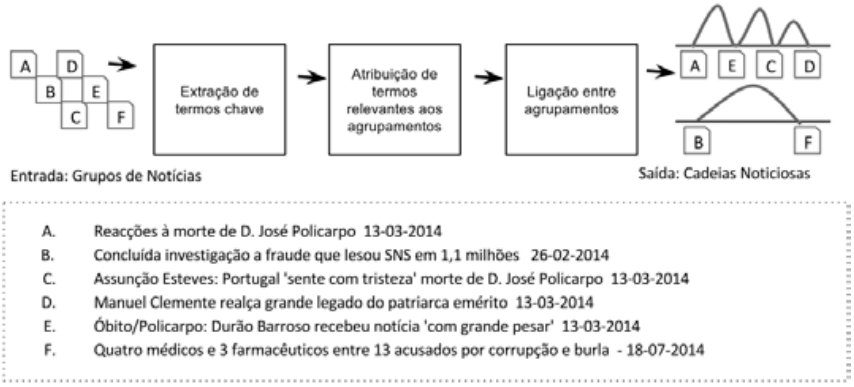


FIGURA 2: Construção de cadeias noticiosas

[2] TRABALHOS RELACIONADOS

[2.1] *Detetar Notícias Duplicadas*

Notícias quase duplicadas são notícias publicadas por fontes distintas mas cujo conteúdo e data de publicação são muito semelhantes. A publicação deste tipo de notícias é bastante comum mas não traz nenhuma mais valia ao leitor. Adicionalmente, o seu armazenamento tem elevados custos computacionais. Devido a estes constrangimentos torna-se necessária a deteção deste tipo de notícias (Kumar & Govindarajulu 2009).

São várias as abordagens propostas para a resolução do problema de deteção de notícias quase duplicadas, entre elas encontram-se: a abordagem baseada no léxico, a abordagem baseada no URL e a abordagem baseada na semântica. A abordagem baseada no léxico não requer nenhum conhecimento linguístico. O objetivo é perceber a existência de termos em comum entre documentos. A abordagem baseada no URL visa detetar notícias duplicadas pela comparação do endereço URL. Porém esta abordagem continua a não ser suficiente. Isto porque, não existe um padrão estabelecido pelas diversas fontes noticiosas de como criar um URL e, portanto, podendo este conter ou não informação útil. A abordagem semântica é uma abordagem mais completa, esta inclui a necessidade de pré-processamento implicando: *tokenization*, *stemming* e remoção das *stop-words*. Após o pré-processamento do texto, as notícias são comparadas através de uma função de similaridade. Esta função tem como objetivo medir o grau de semelhança entre pares de notícias. O valor retornado por esta função varia entre [0,1], e é tanto maior quanto maior for a semelhança existente entre as notícias.

No trabalho intitulado *Duplicate Record Detection: A Survey*, Elmagarmid et al. (2007) explicam todo o fluxo necessário à deteção de documentos duplicados. Este trabalho refere-se à abordagem semântica. As notícias são inicialmente processadas, seguindo-se a determinação dos campos a comparar; é, depois, medido o grau de semelhança entre pares de notícias; e por fim, com base no resultado obtido é determinado se os documentos são ou não similares. Os autores ilustram quatro

métricas: similaridade de *strings* baseada em carateres; similaridade baseada em *tokens*; similaridade fonética e similaridade numérica.

A similaridade baseada em caracteres foi desenvolvida para detetar erros tipográficos. Alguns exemplos dessas métricas são: algoritmos de edição de distância (Hamming (He et al. 2004) e Levenshtein (Levenshtein 1965)) que visam calcular o número de adições, substituições e remoções necessárias para converter uma *string* numa outra, como por exemplo "futebol" e "futbol"; distância Affine Gap (Waterman et al. 1976) que consiste em abrir ou estender um espaço, para transformar uma *string* noutra, como: "C Ronaldo" e "Cristiano Ronaldo"; a métrica de distância Jaro (Bilenko et al. 2003) que mede a semelhança entre duas *strings* tendo em conta o comprimento das mesmas, o número de carateres em comum e o número de transposições necessárias; e a métrica Q-grams (Ullmann 1977) que consiste na divisão das *strings* iniciais em *substrings* de tamanho q, a medição de similaridade entre documentos consiste na medição de *substrings* em comum entre as duas notícias.

Para o cálculo da similaridade entre pares de notícias utilizamos uma abordagem baseada em algoritmos de aprendizagem automática.

Infelizmente, existem poucos estudos desenvolvidos no sentido de verificar a eficiência da utilização de métricas de distância (Elmagarmid et al. 2007). Existem, por exemplo, alguns estudos que mencionam a eficiência da métrica de distância Jaro (Bilenko et al. 2003; Yancey 2005) na comparação de nomes.

Para a deteção e agrupamento de notícias similares é também recorrente a utilização de abordagens de *clustering* (Banerjee et al. 2007; Vadrevu et al. 2011). Nesta abordagem o documento é caracterizado por um conjunto de palavras, usualmente representado por um vetor de frequência da ocorrência dos termos. A determinação da similaridade entre agrupamentos e respetivo agrupamento efetua-se após a aplicação de um algoritmo de *clustering* sobre a coleção. Existem duas abordagens de *clustering* que podem ser aplicadas: a supervisionada, onde os tópicos são conhecidos, e a não supervisionada, onde não existe conhecimento inicial. Existem dois grandes problemas associados à aplicação de técnicas de *clustering* supervisionado, estes são: definição de categorias, tornam o sistema rudimentar, pois ao longo do tempo há uma tendência para o aparecimento de novas categorias; uma categoria abrange não só notícias duplicadas, como abrange também notícias que se referem ao mesmo tema. O problema relacionado com o *clustering* não supervisionado é o de não conhecermos os elementos responsáveis pela elaboração dos agrupamentos.

O nosso contributo, na componente da deteção de notícias quase duplicadas, diz respeito ao estudo da eficiência de alguns algoritmos de edição de distância para textos estruturados de dimensão variável, pela utilização de uma abordagem baseada na semântica. As etapas necessárias para a elaboração deste módulo pode ser observada na Figura 1.

[2.2] *Geração Automática de Histórias*

Diversos trabalhos tem sido conduzidos com o objetivo de criarem histórias a partir de vários documentos como: notícias (Shahaf & Guestrin 2010; Mei & Zhai 2005), blogs (Lin et al. 2012; Qamra et al. 2006) e resultados de pesquisas (Kumar et al. 2004). Em alguns trabalhos, antes da criação da história noticiosa o leitor tem que indicar o tema de pesquisa (Shahaf & Guestrin 2010; Lin et al. 2012). Outros trabalhos porém, visam ser mais abrangentes, e determinar dentro do seu conjunto de dados todas as histórias existentes (Allan et al. 1998b; McKeown et al. 2002). A primeira abordagem é utilizada em estudos relacionados com o tópico "Geração da História" sendo que a segunda abordagem é mais popular em estudos de "Deteção de Tópicos e Monitorização". Em relação a estes dois tópicos, é de notar que existem poucos estudos sobre o primeiro, mas, no entanto, o segundo tópico tem vindo a ser extensivamente estudado (Lin & Liang 2008). Segundo Allan et al. (1998b), o conhecimento inicial dado ao sistema para a criação das histórias pode não ser adequado à monitorização das mesmas uma vez que o tema de discussão associado a um evento muda frequentemente.

Outra área que visa organizar e estruturar informação é a classificação hierárquica (Sun & Lim 2001; Lawrie & Croft 2000; Li et al. 2007). A estrutura hierarquia impõe uma estrutura a um conjunto de dados. Porém, não identificamos nenhum estudo realizado de forma a perceber se essa estrutura reflete as relações existentes entre os diversos documentos (Nallapati et al. 2004).

A nossa abordagem para a geração automática de histórias a partir das notícias baseia-se nas etapas utilizadas nos diferentes trabalhos com o mesmo propósito. As diferentes etapas consideradas, bem com o seu fluxo, podem ser observadas na Figura 2.

Geração da História

O trabalho intitulado *Connecting the Dots Between News* (Shahaf & Guestrin 2010) visa encontrar uma história coerente num conjunto de artigos noticiosos a partir de um ou mais tópicos indicados pelo utilizador. O método utilizado neste trabalho é aplicável a outros domínios como: *emails*, artigos científicos e inteligência militar. Neste trabalho os autores introduziram a noção de coerência, e *feedback* do utilizador. A abordagem proposta pelos autores consistiu na identificação de ligações entre notícias, tendo em conta: palavras omissas, palavras que estão relacionadas com as palavras do texto embora não apareçam no mesmo, e a importância das palavras. O problema da formação das cadeias de notícias foi solucionado recorrendo a uma abordagem de programação linear.

Outro trabalho desenvolvido com o propósito de gerar uma linha temporal de uma história é o *A Graph Teoretic Approach to Extract Storylines from Serach Results* (Kumar et al. 2004). Neste trabalho os resultados de pesquisa são representados numa estrutura de grafos, onde, os nós representam a informação associada ao

documento, e as ligações entre os nós, representam o peso de ligação. Para a elaboração das cadeias, os autores recorrem à utilização de um algoritmo de pesquisa local sobre a estrutura definida.

Deteção de Tópicos e Monitorização

Existem três tarefas associadas a deteção de tópicos e monitorização, são elas: monitorização de eventos conhecidos (eventos já detetados pelo sistema), deteção de novos eventos, e segmentação das notícias em histórias. O grande objetivo dos estudos de deteção de tópicos e monitorização é o de identificar todas e quaisquer notícias relacionadas com um dado evento (Allan et al. 1998a).

Para o nosso trabalho, a componente mais interessante deste estudo é a forma como é executado o monitoramento de uma história nas notícias. A abordagem de monitoramento utilizada em "On-line News event detection and tracking" (Allan et al. 1998b) começa por reduzir o conteúdo noticioso a um conjunto de entre 10 a 20 *features*. Os autores acreditam que poucas *features* são necessárias para o monitoramento de notícias uma vez que o essencial de uma história tende a ser descrito por um conjunto pequeno de palavras ou frases. Neste trabalho, as cadeias são obtidas pelo cálculo de semelhança entre as *queries* que caracterizam cada notícia.

[3] METODOLOGIA

[3.1] *Similaridade*

Abordamos a similaridade entre artigos noticiosos em quatro passos distintos: (i) normalização do conteúdo noticioso;(ii) identificação dos elementos a comparar; (iii) comparação entre pares de notícias; (iv) tomada de decisão.

Normalização

A normalização de textos é uma etapa tradicional em NLP para simplificar a análise posterior dos mesmos. Realizamos as seguintes tarefas de normalização:

1) Remoção de símbolos de pontuação, como: <,>, /, "," , (,), -;
2) Remoção de padrões redundantes e que no âmbito deste trabalho, não são informativos, como: *"Lusa - Esta notícia foi escrita nos termos do Acordo Ortográfico"*;
3) Remoção de *stop-words*, através da utilização de uma lista disponibilizada pelo *snowball* [2] (para a língua portuguesa);
4) Redução das palavras à sua raiz através da utilização do *Porter Stemmer* para língua portuguesa, disponibilizado pelo *PTStemmer* (Oliveira 2008).

Na Tabela 1 apresentamos um exemplo da normalização, desde a notícia original até à sua versão normalizada.

[2] https://snowball.tartarus.org

Operação	Exemplo
Notícia original	Nova Deli, 02 jan (Lusa) - A Índia anunciou que vai permitir a cidadãos estrangeiros investirem no seu mercado de ações.
1- **Pontuação**	Nova Deli 02 jan Lusa A Índia anunciou que vai permitir a cidadãos estrangeiros investirem no seu mercado de ações.
2- **Padrões**	A Índia anunciou que vai permitir a cidadãos estrangeiros investirem no seu mercado de ações.
3- **Stop-words**	Índia anunciou vai permitir cidadãos estrangeiros investirem mercado ações.
4- **Stemm**	Índi anunc va permit cidadã estrangeir invest merc açõ.

TABELA 1: Exemplo do fluxo da normalização.

Identificação dos elementos a comparar

Identificamos cinco conteúdos essenciais nos artigos noticiosos publicados em formato digital: título, corpo da notícia, data de publicação, URL e metadados (*tags*).

URL: provenientes de diferentes domínios têm uma composição distinta. A Tabela 2 apresenta três pares <título, URL>. O primeiro URL é composto pelo título da notícia; já o segundo dá-nos a indicação das áreas a que a notícia está associada, não explicitando em concreto o acontecimento presente; o terceiro exemplo não nos consegue transmitir nenhuma informação concreta para além do domínio.

Al Qaeda revindica atentados em quartel militar do Iêmen
http://visao.sapo.pt/al-qaeda-revindica-atentados-em-quartel-militar-do-iemen=f803958
Plantel empenhado na vitória em Barcelos
http://www.record.xl.pt/Futebol/Nacional/1a_liga/academica/interior.aspx?content_id=919169
Cidade chinesa gera energia com queima de notas de banco
http://diariodigital.sapo.pt/news.asp?id_news=750321

TABELA 2: Exemplos de URL

Corpo da notícia: o título ou corpo da notícia, como componentes isolados, podem não ser suficientes para a determinação da similaridade. Identificamos o cabeçalho da notícia, tipicamente o primeiro parágrafo, como sendo um elemento adicional a considerar para o cálculo da similaridade entre notícias (ver Figura 3). Este cabeçalho corresponde muitas vezes ao resumo da notícia e como tal é muito informativo.

FIGURA 3: Campos da notícia a serem comparados.

Data de publicação: as notícias contêm informação temporal importante para a contextualização do evento. Assumimos que existe um intervalo de tempo restrito dentro do qual há uma maior tendência para o aparecimento de notícias duplicadas. Por exemplo, é mais provável a existência de notícias duplicadas com intervalo de datas de publicação de 24 horas do que numa semana. Deste modo, o fator tempo serve como delimitador do intervalo temporal de notícias comparáveis.

Comparação de Notícias

Podem ser utilizadas diferentes métricas para o cálculo da similaridade. Neste trabalho, consideramos as seguintes: Hamming (He et al. 2004), Levensthein (Levenshtein 1965) e Jaro (Bilenko et al. 2003).

De forma a que os resultados destas métricas possam ser comparáveis, é necessário proceder à normalização dos mesmos, aplicamos a seguinte fórmula (Expressão 1) aos resultados retornados pelos métodos de edição de distância.

$$D'(s,t) = 1 - \frac{D(s,t)}{\max\left(|s|,|t|\right)}, D \in \mathbb{Q} | D \in [0;1] \quad (1)$$

Onde:
$D(s,t)$ é a distância obtida pela métrica de edição de distância entre a string *s* e *t*;
$\max\left(|s|,|t|\right)$ é o comprimento da string de maior dimensão entre *s* e *t*;
$D'(s,t)$ é a distância normalizada entre *s* e *t*.

Para cada par de notícias é calculado o D'. A decisão sobre a similaridade é decidida no passo posterior.

Decisão da similaridade entre notícias

Usamos diversos métodos de aprendizagem supervisionada para a classificação de notícias duplicadas. Os algoritmos usados foram: Support Vector Classifier (SVC), SVC Linear, Decision Tree e Random Forest. Estes algoritmos estão disponíveis, através de bibliotecas python, no *scikit learn* (Pedregosa et al. 2011).

A partir das distâncias calculadas na secção [3.1.3] tiramos partido de algoritmos de aprendizagem supervisionada para classificar pares de notícias como duplicadas ou não duplicadas.

[3.2] *Agrupamento de Notícias*

Este módulo é responsável pela criação de grupos de notícias duplicadas usando os resultados dos pares de notícias previamente classificadas (ver secção [3.1.4]).

Um exemplo ilustrativo dos passos necessários desde a receção das notícias até à composição dos agrupamentos pode ser ilustrado pela Figura 1. Neste caso, estamos perante seis notícias (A,B,C,D,E,F) que formam quinze pares distintos (AB, AC, AD, AE, AF,BC, BD, BE, BF,CD, CE, BF,DE, DF, EF). Estes pares de notícias são comparados na secção [3.1] e deste módulo são considerados como duplicados os pares AD, AE, BF e DE. Pela observação do exemplo, constatamos que são formados dois grupos (BF e ADE).

[3.3] *Extração de Termos Chave*

Para cada grupo de notícias é necessário e essencial, sintetizar a informação contida nesses grupos.

Na nossa abordagem, vamos representar as notícias por um conjunto de termos chave. Os termos chave podem ser considerados termos que transmitem informação relevante do texto, como: o tópico da notícia, nomes de personalidades, locais e outros. Consideramos três tipos de termos chave: (i) palavras isoladas (*uni-grams*) (ii) expressões relevantes (*n-grams*) e (iii) entidades.

Palavras Isoladas

As palavras isoladas correspondem a palavras compostas por um *token* que aparecem explicitamente no conteúdo noticioso. De forma a obtermos estas palavras executamos três tarefas: POS Tagger, normalização e análise da frequência da palavra.

POS Tagger: visa a identificação das categorias gramaticais das palavras que compõe o texto da notícia. Utilizamos nesta tarefa o *TreeTagger* (Schmid 1994) adaptado para a língua portuguesa, disponibilizado por Garcia & Gamallo (2013).

Normalização: corresponde à remoção de padrões linguísticos e frases recorrentes do corpo da notícia obtidos por inspeção manual, como: expressões de datas

(Porto, 12 Agosto 2014), resultados de futebol (2–1) e padrões jornalísticos (Porto, 12 Agosto 2014 (Lusa)).

Análise da frequência da palavra: pela utilização da métrica estatística *Term Frequency-Inverse Document Frequency* (TF-IDF). No seu cálculo, esta métrica relaciona o aparecimento de um termo na notícia com o aparecimento do mesmo na coleção permitindo assim detetar a existência de termos relevantes.

Da análise da frequência de palavras no texto resulta uma lista de palavras com peso associado. Consideramos como palavras relevantes, aquelas com maior peso e pertencentes à categoria gramatical nome.

Expressões Relevantes

As expressões relevantes correspondem a *ngrams* que aparecem explicitamente no conteúdo noticioso e que de uma forma simplificada podem transmitir informação relevante contida no texto.

Para a extração deste elemento do texto foi adicionado um passo intermédio à abordagem apresentada na secção [3.3.1]. Para tal, após a normalização foi aplicado um filtro de forma a obter expressões do texto. As expressões são *ngrams*, que obedecem a certos padrões gramaticais, como: sequências de nomes (Domingos Paciência), nome e adjetivo (homens encapuzados) entre outros.

A análise da frequência é neste caso efetuada sobre os padrões. O resultado retornado pela análise de frequência indica-nos quais as expressões relevantes para a notícia em questão. A última etapa consiste na atribuição das expressões relevantes à notícia.

Entidades

O reconhecimento de entidades mencionadas, nomeadamente o nome de personalidades, é essencial no contexto de extração de termos e expressões chave das notícias.

Existem disponíveis vários recursos para o reconhecimento de entidades mencionadas para a língua portuguesa, como os mencionados pela Linguateca[3]. No entanto e no âmbito deste trabalho, estamos perante um domínio muito dinâmico, as notícias, onde constantemente aparecem novas entidades (Charlie Hebdon, Fukushima). Optamos por implementar um sistema que se adapta a estas características.

Foi implementado um algoritmo com o objetivo de verificar, numa primeira fase, quais as palavras no texto que se iniciam com um caráter maiúsculo. Das palavras encontradas, se a palavra maiúscula estiver posicionada no inicio da frase é verificado se a palavra é ou não uma *stop-word*, e caso seja, então não é considerada. Para as palavras que passarem a fase anterior é verificado se são precedidas

[3] http://www.linguateca.pt/LivroSegundoHAREM/

de outras palavras capitalizadas, sendo permitido uma palavra de ligação entre termos capitalizados inicializada a minúscula. Um exemplo de entidades extraídas pelo algoritmo é dado pelos seguintes termos: "Passos", "Paulo Portas".

De forma a enriquecer os termos chave extraídos para o conjunto de expressões e entidades extraídas de cada noticia tentamos identificar quais desses termos relevantes são nomes de personalidades. Para tal comparamos esses termos com um recurso externo, o Verbetes[4].

[3.4] *Atribuição de termos relevantes aos agrupamentos*

Depois da junção de notícias similares em agrupamentos (secção [3.2]) e após realizada a extração de termos relevantes de cada notícia (secção [3.3]), é possível fazer a atribuição dos termos chave aos agrupamentos de notícias.

Os termos chave associados a cada agrupamento correspondem aos termos relevantes que estão associados a cada uma das notícias do agrupamento. É de referir que cada termo chave tem um peso (w), que está relacionado com a sua frequência (f) no agrupamento. A importância de um termo é dado pela relação entre o número de notícias em que o termo aparece e número total de notícias que compõe o agrupamento. Um exemplo de palavras relevantes associadas a um agrupamento e respetiva importância é dado por:

reclusos[f=9;w=1];presos[f=9;w=1];
cárcere[f=7;w=0.78];sudoeste[f=7;w=0.78];
representantes[f=6;w=0.67];
violação[f=6;w=0.67];cadeia[f=5;w=0.56];
quilómetros[f=4;w=0.44];irmãos[f=4;w=0.44];

Neste agrupamento, o termo *reclusos* é mais representativo do conjunto do que o termo *irmãos*. Isto porque, considerando que o agrupamento em questão tem nove notícias, o primeiro termo aparece associado a todas as notícias do agrupamento ($f = 9$), tendo um peso de $w = \frac{9}{9}$, ou seja 1; enquanto o segundo termo só se encontra associado a 4 notícias do conjunto ($f = 4$), tendo um peso de $w = 0.44$.

[3.5] *Ligações entre Agrupamentos*

Este módulo visa identificar as ligações entre os agrupamentos de notícias duplicadas previamente calculadas com os respetivos termos relevantes associados (ver Figura 2).

Partimos do pressuposto que as cadeias noticiosas só podem existir para a mesma categoria de notícias, de forma a simplificar esta tarefa. Para isso, fizemos a atribuição das categorias aos grupos de notícias, através de uma fonte de conhecimento externo que mapeia as *tags* atribuídas pelos jornalistas com a categoria a que a notícia fica associada. As categorias indicam de uma forma geral a

[4] https://store.services.sapo.pt/pt/Catalog/other/free-api-information-retrieval-verbetes

área a que a notícia pertence como: desporto, sociedade, política, economia, entre outros.

Detalhamos nas subsecções apresentadas de seguida a abordagem utilizada para o processo de ligação de pontos entre os agrupamentos. Este foi realizado em duas etapas: cálculo da distância entre termos relevantes e determinação das ligações entre agrupamentos.

Similaridade de termos relevantes

Começamos por fazer a normalização dos termos relevantes. Para as palavras isoladas, expressões, entidades e personalidades, o texto é convertido para letra minúscula. Para as palavras isoladas que são constituídas apenas por *uni-grams* também se efetua a redução ao seu radical. Após a normalização do texto, é efetuado o cálculo da similaridade entre os termos de cada agrupamentos através do cálculo da distância entre: palavras isoladas, expressões, entidades e personalidades.

Para o cálculo da similaridade entre palavras isoladas, entidades e personalidades, consideramos o peso de cada palavra individual no agrupamento que é dada pelas Expressões 2 e 3.

$$D_1(a,b) = 0.3 \times \frac{|k_a| \wedge |k_b|}{\max(|k_a|,|k_b|)} + 0.7 \times \frac{\sum_{i=1}^{|k_a|}(\sum_{j=1 \wedge a_j=b_i}^{|k_b|} Wk_{aj} \times Wk_{bi})}{|k_a| \wedge |k_b|} \quad (2)$$

$$D_2(a,b) = \frac{|k_a| \wedge |k_b|}{\max(|k_a|,|k_b|)} \times \frac{\sum_{i=1}^{|k_a|}(\sum_{j=1 \wedge a_j=b_i}^{|k_b|} Wk_{aj} \times Wk_{bi})}{|k_a| \wedge |k_b|} \quad (3)$$

Onde:
Wk_{aj} é o peso da palavra-chave j no agrupamento a;
Wk_{bi} é o peso da palavra-chave i no agrupamento b;
$|k_a|$ e $|k_b|$ são o número de palavras-chave iguais entre os agrupamentos a e b;
$\max(|k_a|,|k_b|)$ é o número máximo de palavras-chave distintas.

As distâncias $D_1(a,b)$ e $D_2(a,b)$ têm em conta a percentagem de termos em comum entre os dois agrupamentos e a relação dos pesos que os termos em comum têm nos seus agrupamentos. $D_1(a,b)$ estabelece um peso entre as duas parcelas, dando um maior relevo à parcela que mede o relacionamento dos pesos das palavras em comum; em $D_2(a,b)$ não existem pesos associados às parcelas, mas sim, uma relação entre elas.

Para o cálculo da similaridade entre as expressões relevantes a abordagem utilizada foi distinta. Para este caso, a normalização incluiu um passo adicional,

remoção das *stop-words*. Após esta tarefa foi construída uma *string* com todas as expressões pertencentes a cada agrupamento, não considerando para este tipo de termo relevante o seu peso. O cálculo da similaridade entre as expressões foi baseado num algoritmo de edição de distância o *qgrams* (Ullmann 1977) ($q = 3$).

Determinação das ligações entre agrupamentos

Esta etapa tem como objetivo determinar a partir dos valores de similaridade calculados anteriormente quais as ligações mais relevantes. É a partir destas ligações que se formam as cadeias noticiosas.

Para a ligação de agrupamentos, utilizamos algoritmos de aprendizagem supervisionada. Estes algoritmos recebem um conjunto de treino manualmente anotado com ligações relevantes entre agrupamentos, sobre o qual vão inferir regras para determinar, a existência de ligações válidas e relevantes. Utilizamos como características (*features*) a distância entre as palavras isoladas, expressões, entidades e personalidades. Os algoritmos utilizados foram: *Support Vector Classifier* (SVC), *SVC Linear*, *Decision Tree* e o *Random Forest*.

Ao longo desta secção apresentamos a metodologia utilizada na deteção de notícias duplicadas e na geração automática de cadeias noticiosas.

[4] RECURSOS LINGUÍSTICOS

Nesta secção caracterizamos o conjunto de dados e as fontes de conhecimento externo utilizadas na elaboração deste trabalho.

[4.1] *Caracterização do conjunto de dados*

Para a realização deste trabalho foram utilizadas notícias publicadas *online*, escritas na língua portuguesa e provenientes de diversas fontes noticiosas da imprensa portuguesas. O conjunto de dados compreende mais de 4 milhões de notícias publicadas entre 2008 e 2014.

As notícias são provenientes de 73Número de fontes com mais de 100 notícias publicadas. fontes noticiosas distintas e compostas em média[5] por: 9 palavras no título; 204 palavras no conteúdo; 10 frases no conteúdo.

Na imprensa portuguesa são publicadas *online* diariamente aproximadamente 2.500 notícias[6]. A Figura 4 representa a distribuição de notícias durante mês de Março de 2014. Através da observação da mesma é possível constatar que tendencialmente são publicadas menos notícias durante o fim de semana.

[5] Análise de aproximadamente 74000 notícias selecionadas de um mês aleatório de 2014.

[6] Dados relativos às notícias publicadas na imprensa portuguesa, no formato digital, no mês de Março de 2014

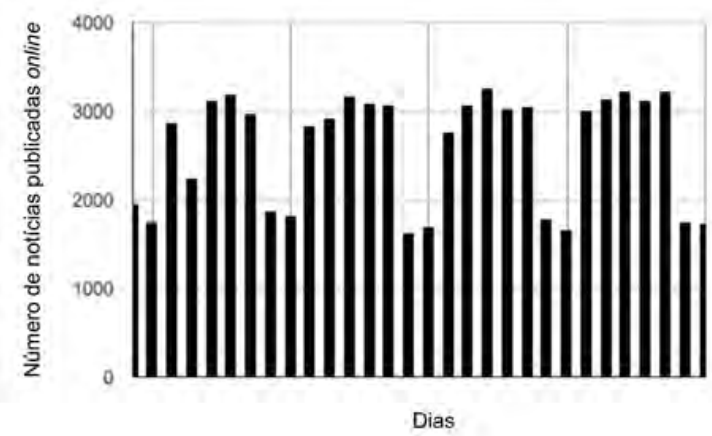


FIGURA 4: Número de notícias publicadas por dia no mês de Março de 2014.

Estima-se que aproximadamente 45%[7] das notícias publicadas diariamente sejam duplicadas ou quase duplicadas. A relação entre o número de notícias publicadas mensalmente com o número de notícias utilizadas para a criação dos agrupamentos pode ser visualizada na Figura 5. Para os primeiros oito meses de 2014 o número médio de notícias por grupo é de 3.8, os dados referentes ao número médio de notícias por grupo relativo a cada mês pode ser observado na Figura 6.

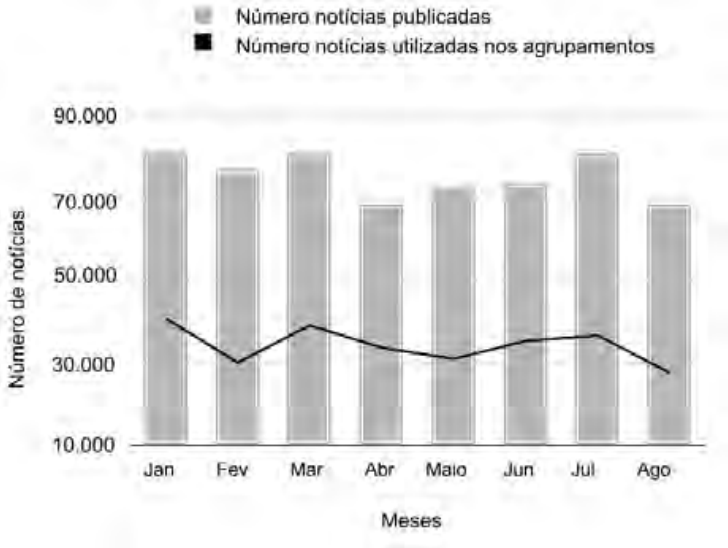


FIGURA 5: Relação entre o número de notícias publicadas por mês com o número de notícias utilizadas na criação dos agrupamentos (Janeiro a Agosto de 2014)

[7] Número médio de notícias *online* diárias duplicadas, publicadas na imprensa portuguesa, de 10 a 15 de Março de 2014

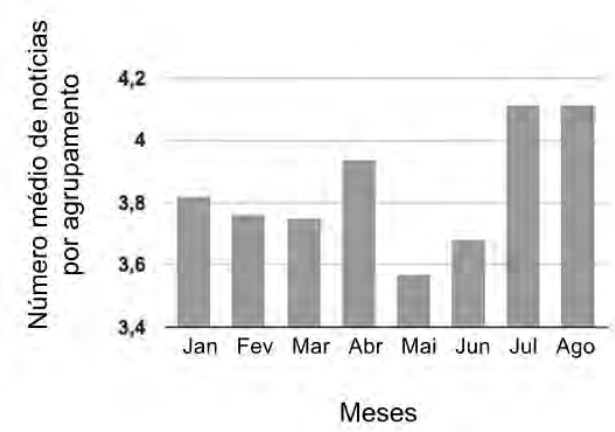


FIGURA 6: Número médio de notícias por agrupamento (Janeiro a Agosto de 2014)

Na Figura 7 podemos constatar que maioritariamente os grupos são constituídos por 2 notícias similares. É possível observar que o número de grupos existentes é inversamente proporcional ao número de notícias que o compõe.

FIGURA 7: Constituição dos agrupamentos (seleção aleatória de 5 dias de 2014)

Definimos nove categorias associadas aos agrupamentos que são as categorias tipicamente usadas nos media digitais para organizar as notícias publicadas *online*: política, economia, desporto, saúde, ciências e tecnologias, sociedade, cultura, local e educação. Dos agrupamentos com apenas uma categoria associada a distribuição dos mesmos por áreas pode ser observado na Figura 8. É possível observar que a categoria com maior expressão é a categoria desporto (54.4%) e assim sucessivamente.

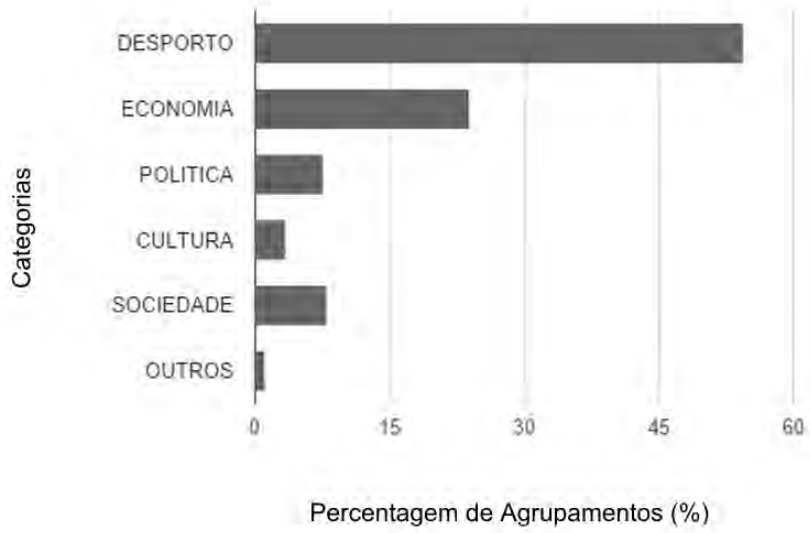


FIGURA 8: Distribuição dos agrupamentos por categoria

[4.2] *Enunciação de fontes de conhecimento externo*

No decorrer deste trabalho foram utilizadas as seguintes fontes de conhecimento:

Lista *stop-words*: Lista de *stop-words* específica para a língua portuguesa disponibilizada pela snowball.

Verbetes: O Verbetes é um sistema de recolha automática de informação a partir das notícias. Para este trabalho utilizamos uma lista de personalidades extraída deste sistema.

Lista de *Tags* e Categorias: Lista elaborada manualmente por jornalistas que relaciona a *tag* associada à notícia com a sua categoria principal.

Nesta secção foi caracterizado o conjunto de dados e as fontes de conhecimento externo utilizadas na elaboração deste trabalho.

[5] EXPERIMENTAÇÃO

Nesta secção são referidas as diferentes métricas de avaliação utilizadas e descrito o conjunto de experiências realizadas.

[5.1] *Métricas de Avaliação*

Para avaliar o módulo de similaridade (ver secção [3.1]) e ligações entre agrupamentos (ver secção [3.5.2]), foram utilizadas quatro métricas de avaliação: a precisão (*precision*), a abrangência (*recall*), a *accuracy* e a *F-measure* (F_1). No contexto deste trabalho, a precisão indica a taxa de notícias consideradas similares que realmente o são e a taxa de ligações efetuadas entre agrupamentos que realmente

existem. A abrangência (*recall*) indica-nos, neste contexto, taxa de notícias duplicadas encontradas face às realmente existentes mas que não conseguimos identificar manualmente. A medida F_1 estabelece uma relação entre a precisão e a abrangência. A *accuracy* indica-nos a avaliação geral do sistema.

A avaliação aos termos relevantes focou-se em avaliar, dos termos extraídos, quais são de facto realmente representativos da notícia. A avaliação foi realizada usando a Expressão 4. A avaliação geral do sistema é dada pelo somatório percentagem de termos representativos das notícias analisadas, Expressão 5.

$$E(ni) = \frac{Termos_{Representativos}}{Termos_{Atribuídos}} \tag{4}$$

$$Avaliação = \frac{\sum_{i=1}^{||N||}(E(n_i))}{||N||} \tag{5}$$

Onde:
$Termos_{Representativos}$ corresponde ao número de termos relevantes ou entidades atribuídos pelo método, que realmente representam o conteúdo noticioso;
$Termos_{Atribuídos}$ corresponde ao número total de termos relevantes ou entidades atribuídas ao documento;
$||N||$: número de notícias da coleção N;
n_i: corresponde à notícia de índice i do conjunto de notícias N.

[5.2] *Enunciação e definição das experiências*

Nesta secção são apresentadas as cinco experiências realizadas. Começamos por apresentar três experiências relativas à determinação da similaridade entre notícias. Na primeira experiência pretendemos perceber qual o algoritmo mais adequado ao cálculo da similaridade entre notícias. A segunda experiência visa entender qual a influência do fator tempo neste domínio, ou seja, se as notícias duplicadas ou quase duplicadas surgem em intervalos temporais longos ou curtos. Por fim a terceira experiência tem como objetivo perceber qual o método de aprendizagem supervisionado mais apto para a determinação da similaridade entre notícias.

A quarta experiência enunciada está relacionada com os termos chaves extraídos. Por fim a quinta experiência refere-se às ligações entre agrupamentos.

Será usado $Exp_{i,j}$ para representar a j-ésima configuração de parâmetros para a experiência i.

Similaridade - Algoritmos de Edição de Distância
A similaridade entre notícias é obtida através do cálculo da:

- Similaridade do título (ST) que corresponde à percentagem de semelhança entre os títulos;
- Similaridade do 1º parágrafo (SB) que corresponde ao resultado de comparação entre a parte das notícias que foca o evento em si;
- Similaridade de conteúdo noticioso (SC) que corresponde ao resultado da comparação do corpo das respetivas notícias.

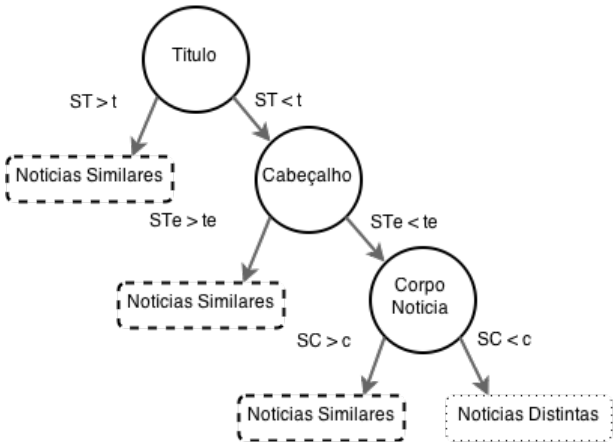


FIGURA 9: Árvore de decisão elaborada para verificar se um par de notícias é ou não similar.

Esta experiência — Exp_1 — visou perceber qual o algoritmo com o melhor desempenho para o cálculo da similaridade entre pares de notícias. Esta experiência foi efetuada sobre uma estrutura em forma de árvore de decisão, representada na Figura 9. Esta foi criada manualmente, onde os valores *t, f, c,* correspondem aos valores de decisão para o título, foco e conteúdo da notícia. L, H, J correspondem respetivamente aos algoritmos Leventshein, Hamming e Jaro. A parametrização usada nesta experiência encontra-se enunciada na Tabela 3. Por exemplo, a $Exp_{1,1}$ é efetuada individualmente para os algoritmos Levenshtein, Hamming e Jaro, com um *threshold* de 0.6 para *t, f* e *c*. As diferentes experiências visam perceber a influência que os diferentes *thresholds* têm nos algoritmos.

Para a realização desta experiência foram comparadas aleatoriamente 124750 notícias, para um dia aleatório de 2014.

Similaridade - Fator Tempo
A experiência sobre o fator tempo (intervalo temporal) tem como objetivo verificar a influência do intervalo temporal no que diz respeito à identificação e classificação de notícias similares. Para tal, foram considerados cinco intervalos de

Exp	Algoritmos	t	f	c
$1,1$	L H J	0.60	0.60	0.60
$1,2$	L H J	0,70	0,60	0,60
$1,3$	L H J	0,70	0,70	0,60
$1,4$	L H J	0,70	0,70	0,70
$1,5$	L H J	0,80	0,70	0,70
$1,6$	L H J	0,80	0,80	0,70
$1,7$	L H J	0,80	0,80	0,80

TABELA 3: Parametrização para a experiência do cálculo da similaridade.

tempo distintos: 3, 6, 12, 24, 48 horas; e foram utilizados quatro métodos de classificação para a determinação da similaridade: SVC, SVC Linear, *Decision Tree* e o *Random Forest*. Esta experiência foi elaborada utilizando uma técnica de avaliação cruzada, o *k-fold cross validation* ($k = 5$). O conjunto de dados utilizado resulta da seleção aleatória de 500 notícias de dois dias distintos e consecutivos, anotadas manualmente.

Similaridade - Decisão da similaridade entre notícias
Foi efetuada uma experiência com o objetivo de perceber qual o algoritmo de aprendizagem supervisionada com o melhor desempenho na determinação da similaridade entre pares de notícias. A experiência foi efetuada em 500 notícias selecionadas de forma aleatória de um dia aleatório de 2014.

Extração de Termos relevantes
Esta experiência tem como objetivo testar a abordagem utilizada para a extração de termos chave (palavras isoladas, expressões e entidades). Para a realização desta experiência foi selecionado aleatoriamente um dia de cada mês do ano 2012. De cada dia foi selecionado um intervalo de três horas, e dessas três horas foram selecionadas aleatoriamente dez notícias sobre as quais se efetuou a inspeção manual das palavras-chave atribuídas.

Ligações entre agrupamentos
Para a determinação das ligações entre agrupamentos de notícias, é realizado o cálculo da distância entre: palavras isoladas, expressões, entidades e personalidades.

Esta experiência — Exp_2 — tem como objetivo avaliar qual a abordagem mais adequada para o cálculo da similaridade e qual o método de aprendizagem supervisionado mais eficiente para a determinação das ligações. Todas as experiências consideraram o cálculo distância pelo algoritmo Q-grams, para as expressões. A avaliação resultante das diferentes experiências realizadas entre grupos de notí-

cias ao longo do tempo, para a formação de ligações entre agrupamentos de notícias, encontra-se na Tabela 4. O conjunto de dados é composto por agrupamentos pertencentes aos meses de março e abril de 2014. Desses agrupamentos, foram selecionados aleatoriamente 10 cadeias de notícias com tamanho variável para cada uma das seguintes categorias: desporto, economia, política, cultura e sociedade. O conjunto de dados compreende, em média, 317 comparações por categoria.

Exp	Palavras	Entidades	Personalidades
$2,1$	D_1	D_2	D_1
$2,2$	D_2	D_2	D_1
$2,3$	D_1	D_1	D_1
$2,4$	D_1	D_2	D_2

TABELA 4: Descrição das experiências para o cálculo das ligações.

Nesta secção foram apresentadas as diferentes métricas de avaliação utilizadas e descrito o conjunto de experiências realizadas.

[6] RESULTADOS E ANÁLISE

[6.1] *Experiências*

Similaridade - Algoritmos de Edição de Distância

Os resultados obtidos nesta experiência — Exp_1 — podem ser observados na Tabela 5. Desta tabela excluímos os resultados obtidos para algoritmo Jaro, devido ao seu desempenho constante.

	Levensthein			Hamming		
Exp	P	R	F	P	R	F
$1,1$	0,941	0,761	0,841	0,941	0,289	0,442
$1,2$	0,950	0,655	0,775	0,940	0,284	0,436
$1,3$	0,951	0,645	0,769	0,940	0,284	0,436
$1,4$	**0,972**	**0,637**	**0,770**	0,940	0,284	0,436
$1,5$	0,965	0,507	0,665	0,939	0,279	0,430
$1,6$	0,964	0,483	0,643	0,939	0,279	0,430
$1,7$	0,962	0,463	0,625	0,938	0,279	0,430

TABELA 5: Resultados dos testes aos algoritmos de edição de distância.

Da comparação entre o algoritmo *Levensthein* e o *Hamming* em $Exp_{1,1}$ podemos verificar que os valores da precisão são semelhantes, o que indica que a percentagem de notícias consideradas similares que realmente o são (*true positive*) é igual. Para o mesmo caso podemos verificar uma melhoria para o algoritmo *Levensthein* para o *recall*.

Similaridade - Fator Tempo

O resultado obtido desta análise pode ser observado no gráfico apresentado na Figura 10. Como podemos constatar pela análise do gráfico, o aumento do intervalo de tempo faz com que os valores se tornem constantes. Ao alargar o intervalo de tempo de 24 para 48 horas não há variação nos valores de *precision*, *recall* e da métrica F_1.

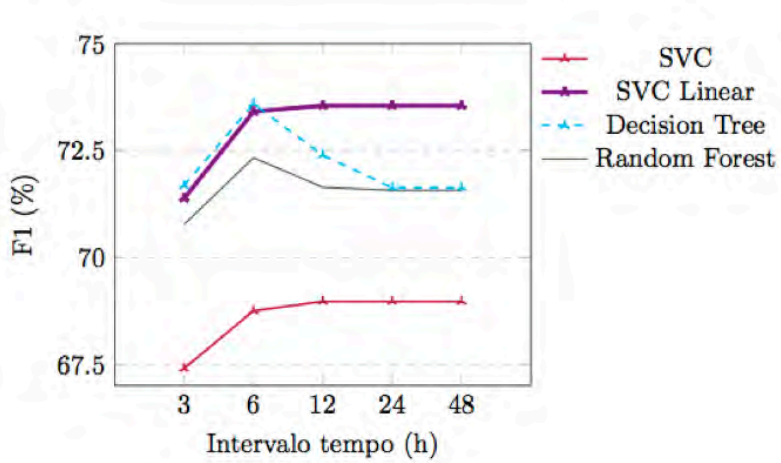


FIGURA 10: Valor da métrica F_1 obtido pelos diferentes algoritmos nos diferentes intervalos de tempo.

Similaridade - Determinação Semelhança

Os resultados dos algoritmos de aprendizagem supervisionados na determinação da similaridade podem ser observados na Tabela 6. Pela visualização da tabela é possível constatar que apesar do valor do *recall* ser baixo, o valor obtido pela *precision* é alto, o que garante a elevada qualidade da informação recolhida. O algoritmo que apresenta um melhor desempenho é o SVC Linear.

	P	**R**	**F_1**	**A**
Decision Tree	0,863	0,679	0,760	0,998
SVC	0,931	0,508	0,657	0,997
SVC Linear	**0,938**	**0,561**	**0,702**	**0,998**
Random Forest	0,803	0,542	0,647	0,998

TABELA 6: Resultado médio das métricas de avaliação obtidas pelo *k fold cross validation*.

Extração de Termos Relevantes

Os resultados da extração de termos relevantes pode ser observado na Tabela 7. A representatividade das palavras extraídas face à informação contida nas notícias é de: 73,2% para as palavras isoladas, 76,2% para as expressões e 80.4% para as entidades.

	Avaliação
Palavras	0,732
Expressões	0,762
Entidades	0,804

TABELA 7: Avaliação dos termos chave.

Ligações entre agrupamentos

Na Tabela 8 são apresentados os resultados da precisão para as ligações entre agrupamentos. A partir da análise dos resultados podemos verificar que o método com um melhor desempenho é o *SVC Linear* e que em 93.3% dos casos analisados as ligações entre notícias são verdadeiras.

Exp	**SVC**	**Decision Tree**	**Random Forest**
$2,1$	**0.931**	0.849	0.859
$2,2$	0.921	0.821	0.852
$2,3$	0.906	0.764	0.824
$2,4$	0.931	0.834	0.858

TABELA 8: Valor da precisão na determinação de ligações entre agrupamentos de notícias.

[6.2] *Análise dos resultados obtidos*

Similaridade - Algoritmos de Edição de Distância

Dos resultados obtidos nestas experiências, podemos observar na Tabela 5 que o algoritmo *Jaro* é o que apresenta a nível global um pior desempenho. No entanto, segundo estudos realizados, este algoritmo tem um melhor desempenho aquando da comparação de pequenas *strings* (Bilenko et al. 2003), o que não acontece no domínio das notícias. Os valores da precisão entre a utilização do algoritmo *Levensthein* e o *Hamming* são muito próximos, obtendo o algoritmo *Levensthein* ao longo das diferentes experiências um melhor desempenho nesta métrica. Comparando as restantes métricas de avaliação, para estes dois algoritmos, é possível observar que o *Levensthein* obtém uma melhor performance a nível da métrica *re-*

call, o que significa que consegue detetar mais casos do que o *Hamming*. Uma razão para que isto suceda está relacionado com uma particularidade deste último algoritmo que é a comparação de *strings* do mesmo comprimento; a nível da métrica F_1, também o *Levensthein* obtém um melhor resultado. Através da análise efetuada a estes três algoritmos é possível concluir que o *Levensthein* é o algoritmo mais indicado para o cálculo da similaridade entre pares de notícias.

Similaridade - Fator Tempo

Um fator importante para a comparação das notícias é a sua data de publicação. Dos resultados apresentados, os algoritmos que apresentam uma melhor precisão são o SVC e o SVC Linear. Sendo que destes dois, o SVC Linear tem um desempenho superior a nível do *recall* e da métrica F_1. Relativamente à questão temporal, podemos perceber, que todos os algoritmos têm um comportamento semelhante à medida que o intervalo temporal aumenta. Pela análise do gráfico é possível verificar que não existem variações dos resultados quando o intervalo de tempo é alargado de 24 para 48 horas. Isto pode indicar que os casos de notícias duplicadas ou quase duplicadas surgem quase sempre num intervalo inferior ou igual a 24 horas. Com base nos resultados obtidos constatou-se que um intervalo de tempo de 24 horas era o mais adequado para a comparação de notícias.

Similaridade - Determinação Semelhança

Para a determinação da similaridade das notícias, os algoritmos que apresentam um melhor desempenho, considerando o $\Delta T = 24$ horas, são: a nível da precisão o SVC Linear (93.8%) e SVC (93.1%) ; em relação à métrica recall e a métrica F_1 o *Decision Tree* (67.9% e 76.0%) e SVC Linear (56.1% e 70.2%). Comprando o desempenho dos diferentes algoritmos para as diferentes fases de processamento e tendo em conta as opções escolhidas a nível de algoritmo de cálculo da similaridade e intervalo de tempo considerado, podemos constatar que o algoritmo que apresenta um melhor desempenho a nível global é o SVC Linear.

Extração de Termos Relevantes

A avaliação manual à relevância das palavras-chave extraídas consistiu em analisar a representatividade dos termos extraídos do texto em relação ao conteúdo da notícia. O resultado da avaliação a estes elementos pode ser observada na Tabela 7. Os resultados indicam que 73,2% das palavras, 76,2% das expressões e 80,5% das entidades são representativas do conjunto. Através da análise ao teor dos termos extraídos foi possível constatar que as palavras relevantes dizem respeito a palavras que descrevem de forma genérica o conteúdo da notícia; por sua vez, as expressões relevantes já transmitem com mais especificidade o assunto da notícia. No exemplo das notícias sobre o desaparecimento do Avião da Malaysia Airlines, temos como palavra relevante *avião* e como expressão *avião Malaysia Airlines*.

Ligações entre agrupamentos

Da análise aos resultados obtidos pela comparação da $Exp_{2,1}$ com a $Exp_{2,2}$, em que o que foi modificada a fórmula de cálculo da distância entre as palavras isoladas, é possível observar que todos os algoritmos conseguem um melhor desempenho considerando a fórmula de cálculo *D1*; face à diferença da precisão entre os algoritmos: 0.010 no SVC Linear; 0.028 no *Decision Tree* e 0.007 no *Random Forest* . Estabelecendo uma comparação entre as experiências $Exp_{2,1}$ e a $Exp_{2,3}$, que divergem apenas na fórmula de cálculo da distância entre as entidades, temos que: a utilização da fórmula *D2* no cálculo da proximidade de entidades entre dois conjuntos reflete um aumento de desempenho. Confrontando os valores obtidos para a experiência $Exp_{2,1}$ em relação à experiência $Exp_{2,3}$ é possível constatar que independentemente do algoritmo de aprendizagem supervisionada os resultados da $Exp_{2,1}$ são os que apresentam um melhor desempenho. Os valores da precisão obtidos para a experiência $Exp_{2,1}$ e a $Exp_{2,4}$ são bastante próximos. Esta experiência difere da primeira na fórmula de cálculo da distância entre personalidades. A partir dos resultados obtidos conclui-se que as personalidades não têm grande impacto na formação das ligações comparativamente com as palavras isoladas e entidades, uma vez que a mudança de cálculo para este elemento não reflete uma variação considerável no resultado. Podemos ainda observar que o melhor desempenho continua a ser o resultante da experiência $Exp_{2,1}$. Após o estudo dos resultados obtidos, podemos concluir que a fórmula mais apta para cada tipo de palavra-chave é a seguinte: *D1* — personalidades e palavras isolada; *D2* — entidades; sendo que esta combinação se refere à experiência $Exp_{2,1}$. Comparando os resultados obtidos pelos diferentes métodos de aprendizagem supervisionada para $Exp_{2,1}$ podemos observar que o método com um melhor desempenho é o SVC Linear (93.1%).

[7] INTERFACE

Desenvolvemos uma interface *web* para permitir ao leitor a navegação entre cadeias de notícias. A interface que elaboramos pode ser observada na Figura 11.

A interface é composta por cinco secções distintas. A primeira secção permite que o utilizador defina as características das cadeias de notícias a visualizar. É permitido definir o intervalo temporal, a categoria das notícias e ainda as palavras-chave. A segunda secção, informa o utilizador quais as características das histórias que estão representadas na interface.

As histórias são representadas visualmente na terceira secção. O gráfico com a representação das histórias pode ser repartido em três elementos interconectados. Começando pela parte inferior do gráfico, em 3.3, as linhas representam os agrupamentos de notícias existentes. O comprimento destas barras varia consoante o número de notícias que compõe cada agrupamento. Na parte superior do gráfico, em 3.1, os arcos representam as ligações existentes entre os agrupa-

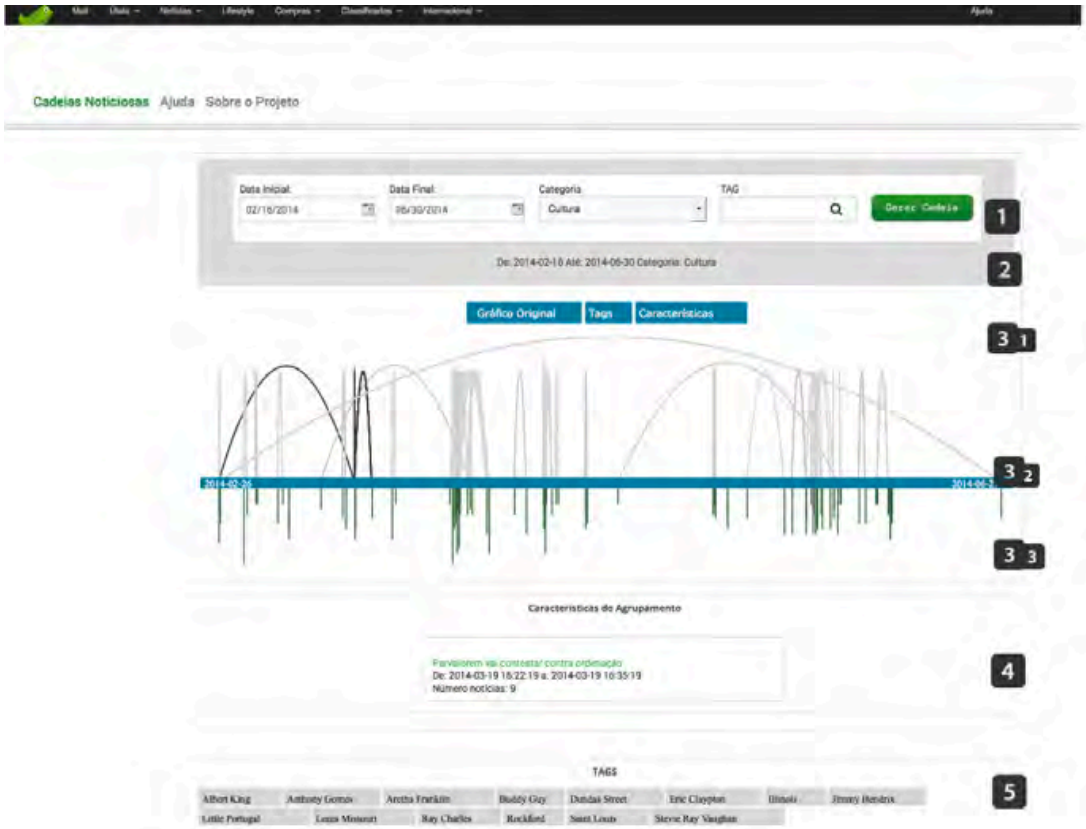


FIGURA 11: Interface do sistema.

mentos de notícias (em 3.3). A barra situada em 3.2 posiciona temporalmente a informação apresentada (em 3.1 e 3.3).

A informação presente na quarta secção varia consoante a interação do utilizador com o gráfico. Se o utilizador navegar sobre a parte 3.3 do gráfico a informação que aparece nesta secção informa o utilizador das características do agrupamento. Porém, se o utilizador navegar na parte 3.1 do gráfico, a informação contida na secção quatro informará o utilizador da história noticiosa. A quinta secção apresenta a lista de palavras-chave mais relevantes dentro do intervalo temporal considerado.

A Figura 12 apresenta parte de uma cadeia obtida pelo sistema (para a categoria Cultura de 31 de Janeiro até 17 de Fevereiro de 2014). A interface será brevemente lançada ao público.

[8] CONCLUSÕES E TRABALHO FUTURO

Este artigo pretende definir e avaliar técnicas para o encadeamento automático de notícias com vista à construção de histórias noticiosas temporais. A abordagem utilizada para a criação das cadeias baseia-se: (i) deteção de notícias (quase) duplicadas e (ii) a criação de ligações entre notícias relacionadas ao longo do tempo.

FIGURA 12: Parte de uma cadeia obtida pelo sistema.

Para a deteção de notícias duplicadas usamos uma abordagem baseada na semântica para o cálculo da similaridade entre notícias. Foi também utilizado um algoritmo de aprendizagem supervisionado na determinação da semelhança entre as mesmas. Adicionalmente, as notícias incluem informação temporal e, tal como acreditávamos, existe um intervalo onde há uma maior tendência para o aparecimento de notícias cujo grau de similaridade aponta para a (quase) duplicação. O nosso estudo indicou que tendencialmente as notícias consideradas duplicadas aparecem num intervalo inferior a 24 horas. A nossa abordagem, para a determinação de notícias cujo grau de similaridade as classifica como (quse) duplicadas, num intervalo de tempo de 24 horas, obteve uma precisão de 93.8% quando usado o par Levenshtein, SVC Linear.

Para a criação de ligações entre grupos de notícias similares, a nossa abordagem consistiu na medição do grau de semelhança entre os diferentes grupos. Para esta etapa, sugerimos uma nova forma de medição de distância que tem em conta os termos em comum e a expressão de cada termo nos agrupamentos de notícias similares. Para a determinação das ligações, foram também utilizados algoritmos de aprendizagem supervisionada. A abordagem proposta para a realização desta segunda tarefa apresenta uma precisão de 93.1%. Este resultado, não representa, no entanto a precisão global do sistema, uma vez que há propagação de erro entre as várias etapas.

Como trabalho futuro será importante criar testes mais exaustivos e objetivos para as cadeias de notícias. Tais testes, consistirão, entre outros melhoramentos, na medição da familiaridade do leitor com um tema em específico antes e depois da utilização da plataforma e na medição do erro propagado pelo sistema.

Também pretendemos melhorar o sistema através da:(i) introdução de sumários das notícias, (ii) deteção de novos factos e (iii) hierarquização de notícias.

AGRADECIMENTOS

Agradecemos a colaboração do Labs SAPO UP pela disponibilização dos dados utilizados neste trabalho.

REFERÊNCIAS

Allan, James, Jaime G. Carbonell, George Doddington, Jonathan Yamron & Yiming Yang. 1998a. Topic detection and tracking pilot study final report. Em *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*, 194–218.

Allan, James, Ron Papka & Victor Lavrenko. 1998b. On-line new event detection and tracking. Em *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval*, 37–45. ACM.

Banerjee, Somnath, Krishnan Ramanathan & Ajay Gupta. 2007. Clustering short texts using Wikipedia. Em *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07*, 787–788. ACM.

Bilenko, Mikhail, Raymond Mooney, William Cohen, Pradeep Ravikumar & Stephen Fienberg. 2003. Adaptive Name Matching in Information Integration. *IEEE Intelligent Systems* 18(5). 16–23.

Elmagarmid, Ahmed K., Panagiotis G. Ipeirotis & Vassilios S. Verykios. 2007. Duplicate record detection: A survey. *IEEE Transactions on Knowledge and Data Engineering* 19(1). 1–16.

Garcia, Marcos & Pablo Gamallo. 2013. FreeLing e TreeTagger: um estudo comparativo no âmbito do Português. Relatório técnico. ProLab Technical Report, vol. 01. http://gramatica.usc.es/~gamallo/artigos-web/PROLNAT_Report_01.pdf.

He, Matthew X., Sergei V. Petoukhov & Paolo E. Ricci. 2004. Genetic code, Hamming distance and stochastic matrices. *Bulletin of mathematical biology* 66(5). 1405–1421.

Kumar, J. Prasanna & P. Govindarajulu. 2009. Duplicate and Near Duplicate Documents Detection: A Review. *European Journal of Scientific Research* 32. 514–527.

Kumar, Ravi, Uma Mahadevan & Alan D. Sivakumar. 2004. A Graph-theoretic Approach to Extract Storylines from Search Results. Em *Proceedings of the tenth international conference on knowledge discovery and data mining*, 216–225.

Lawrie, Dawn & W Bruce Croft. 2000. Discovering and Comparing Topic Hierarchies. Em *Proceedings of the RIAO 2000 conference*, 314–330.

Levenshtein, Vladimir. 1965. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Doklady Akademii Nauk SSSR* 163. 845–848.

Li, Tao, Shenghuo Zhu & Mitsunori Ogihara. 2007. Hierarchical document classification using automatically generated hierarchy. *Journal of Intelligent Information Systems* 29(2). 211–230.

Lin, Chen, Chun Lin, Jingxuan Li, Dingding Wang, Yang Chen & Tao Li. 2012. Generating Event Storylines from Microblogs. Em *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, 175–184.

Lin, Fu-ren & Chia-Hao Liang. 2008. Storyline-based summarization for news topic retrospection. *Decision Support Systems* 45(3). 473–490.

McKeown, Kathleen R., Regina Barzilay, David Evans, Vasileios Hatzivassiloglou, Judith L. Klavans, Ani Nenkova, Carl Sable, Barry Schiffman & Sergey Sigelman. 2002. Tracking and summarizing news on a daily basis with Columbia's Newsblaster. Em *Proceedings of the second international conference on Human Language Technology Research*, 280–285.

Mei, Qiaozhu & ChengXiang Zhai. 2005. Discovering Evolutionary Theme Patterns from Text: An Exploration of Temporal Text Mining. Em *Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining, KDD'05*, 198–207. ACM.

Nallapati, Ramesh, Ao Feng, Fuchun Peng & James Allan. 2004. Event threading within news topics. Em *Proceedings of the thirteenth ACM international conference on Information and knowledge management*, 446–453.

Oliveira, Pedro. 2008. Ptstemmer - a stemming toolkit for the portuguese language. Obtido em Maio 2014. http://code.google.com/p/ptstemmer.

Pedregosa, Fabian, Gael Varoquaux, Alexandre. Gramfort, Vicent Michel, Bertrand Thirion, Oliver Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vicent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau,

Matthieu Brucher, Matthieu Perrot & Édouard Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12. 2825–2830.

Qamra, Arun, Belle Tseng & Edward Y Chang. 2006. Mining blog stories using community-based and temporal clustering. Em *Proceedings of the 15th ACM international conference on Information and knowledge management*, 58–67. ACM.

Schmid, Helmut. 1994. Probabilistic Part-of-Speech Tagging Using Decision Trees. Em *International Conference on New Methods in Language Processing*, 44–49.

Shahaf, Dafna & Carlos Guestrin. 2010. Connecting the Dots Between News Articles. Em *Proceedings of the 16th International Conference on Knowledge Discovery and Data Mining*, 623–632.

Sun, Aixin & Ee-Peng Lim. 2001. Hierarchical text classification and evaluation. Em *Proceedings IEEE International Conference on Data Mining*, 521–528.

Ullmann, Julian R. 1977. A binary n-gram technique for automatic correction of substitution, deletion, insertion and reversal errors in words. *The Computer Journal* 20(2). 141–147.

Vadrevu, Srinivas, Choon Hui Teo, Suju Rajan, Kunal Punera, Byron Dom, Alexander J. Smola, Yi Chang & Zhaohui Zheng. 2011. Scalable clustering of news search results. Em *Proceedings of the fourth ACM International Conference on Web Search and Data Mining, wsdm'11*, 675–684. ACM.

Waterman, Michael S., Temple F. Smith & William A. Beyer. 1976. Some biological sequence metrics. *Advances in Mathematics* 20(3). 367–387.

Yancey, William E. 2005. Evaluating string comparator performance for record linkage. Relatório técnico. Statistical Research Division. http://www.census.gov/srd/papers/pdf/rrs2005-05.pdf.

CONTACTOS

Carla Abreu
Faculdade de Engenharia da Universidade do Porto
cfma@fe.up.pt

Jorge Teixeira
Faculdade de Engenharia da Universidade do Porto
jft@fe.up.pt

Eugénio Oliveira
Faculdade de Engenharia da Universidade do Porto
eco@fe.up.pt

Apêndice D

Sugestão Interface

ENCADEAr

ENCADEAmento automático de notícias

Equipa: Carla Abreu, Jorge Teixeira, Prof. Eugénio Oliveira

Projeto

- Diariamente são publicadas, *online*, milhares de notícias.
- Cerca de 50% das notícias publicadas referem-se à descrição de eventos já noticiados.
- A quantidade de informação disponível dificulta o acompanhamento de novos eventos que se referem a un acontecimento em particular.

Projeto

Acontecimento:

Desaparecimento *Malaysia Airlines*

Restrição temporal:

8 Março a 31 Julho 2014

Número de Resultados:

1022*

* Fonte - Google News - Edição Portugal

Projeto

Acontecimento:

Desaparecimento *Malaysia Airlines*

Restrição temporal:

8 Março a 31 Julho 2014

Número de Resultados:

1022*

Neste conjunto de resultados torna-se complicado ou até mesmo impossível não só a deteção de todos os eventos como apenas os mais relevantes para a história.

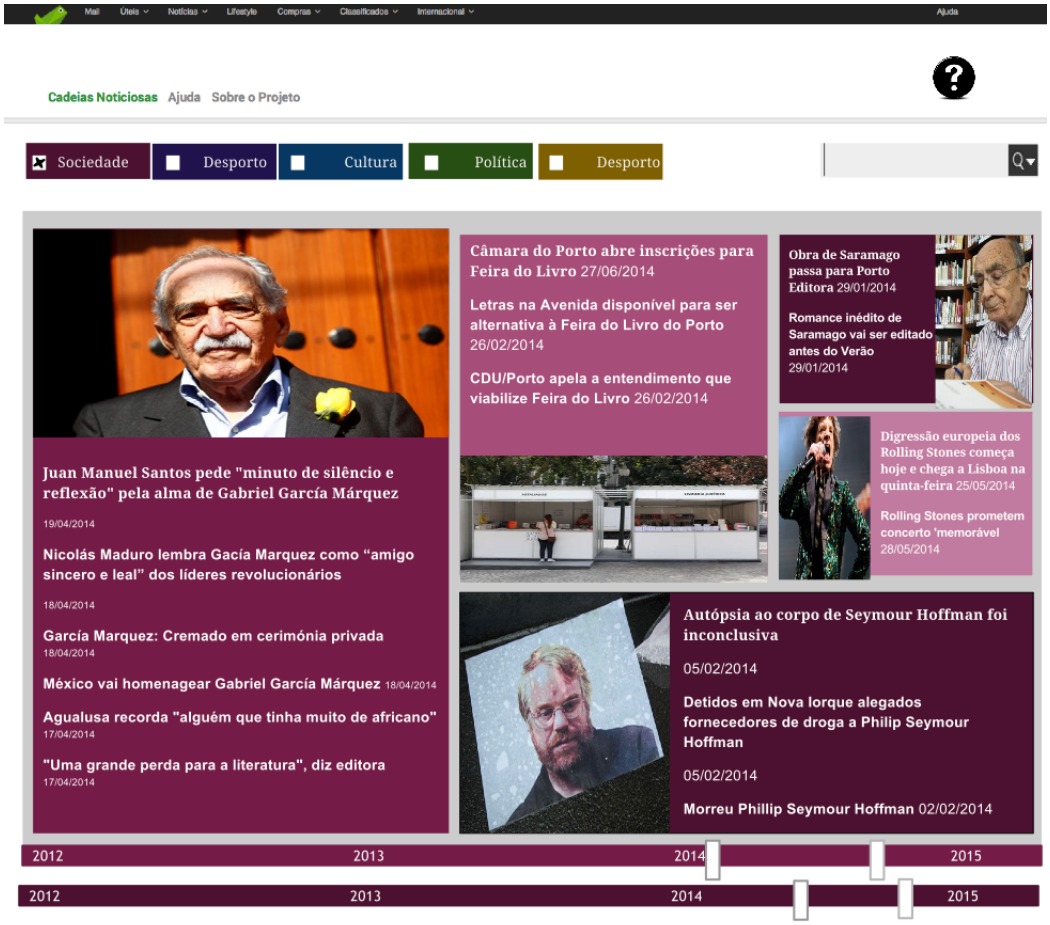
* Fonte - Google News - Edição Portugal

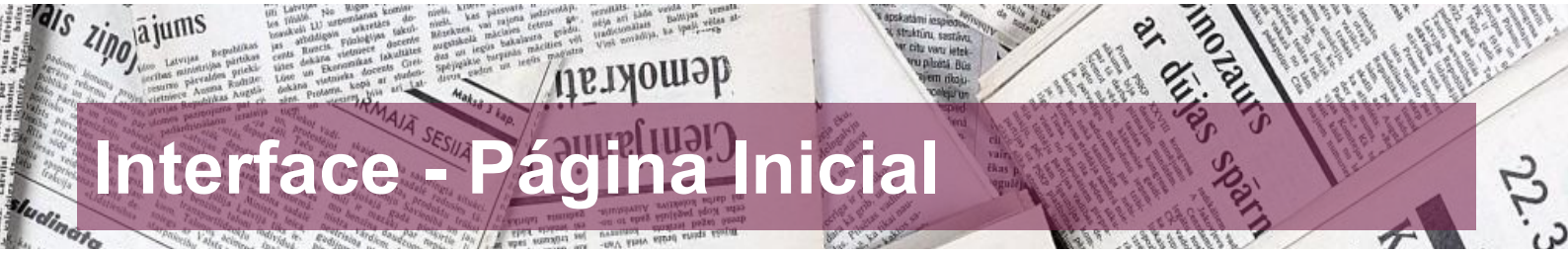
Projeto - Objetivo

- Detetar e agrupar notícias similares:

- Encadear temporalmente notícias relacionadas com o mesmo acontecimento.

Interface - Página Inicial





Interface - Página Inicial

Área Superior:

Botão Ajuda:

Botão que aparece apenas na primeira vez que o utilizador visualiza o *website*. Verificação em *Javascript* da variável *Local Storage*. Ao clicar sobre este elemento aparecerá uma *popup windows* que indicará ao utilizador a forma de funcionamento do *website*.

Categorias:

Existem 5 categorias: Sociedade Desporto, Cultura, Política, Desporto (*HTML-select*). Nesta área o utilizador deverá selecionar apenas uma categoria. Ao fazer a seleção a informação contida na área central irá ser modificada.

Área de Pesquisa:

Introdução de texto e sugestão de palavras (*HTML - Input text* + *Javascript - autocomplete*)

Interface - Página Inicial

Área Central:

Cadeias de Notícias:

Blocos (*D3-SVG* ou *Ink*). O bloco tem apenas uma área selecionável que apresenta em detalhe a informação da cadeia noticiosa. Cada bloco é composto por: várias imagens representativas da cadeia (*Javascript* - renovada a cada 5 segundos), pelos títulos e data de publicação das notícias. Navegação na horizontal (*slide bar*).

Interface - Página Inicial

Área inferior:

Barras Temporais:

Permitem a definição da janela temporal das notícias a apresentar (*slide bar*).

Interface - Descrição Grupo Notícias

Popup window - Cadeia Notícias

<u>Topo:</u>

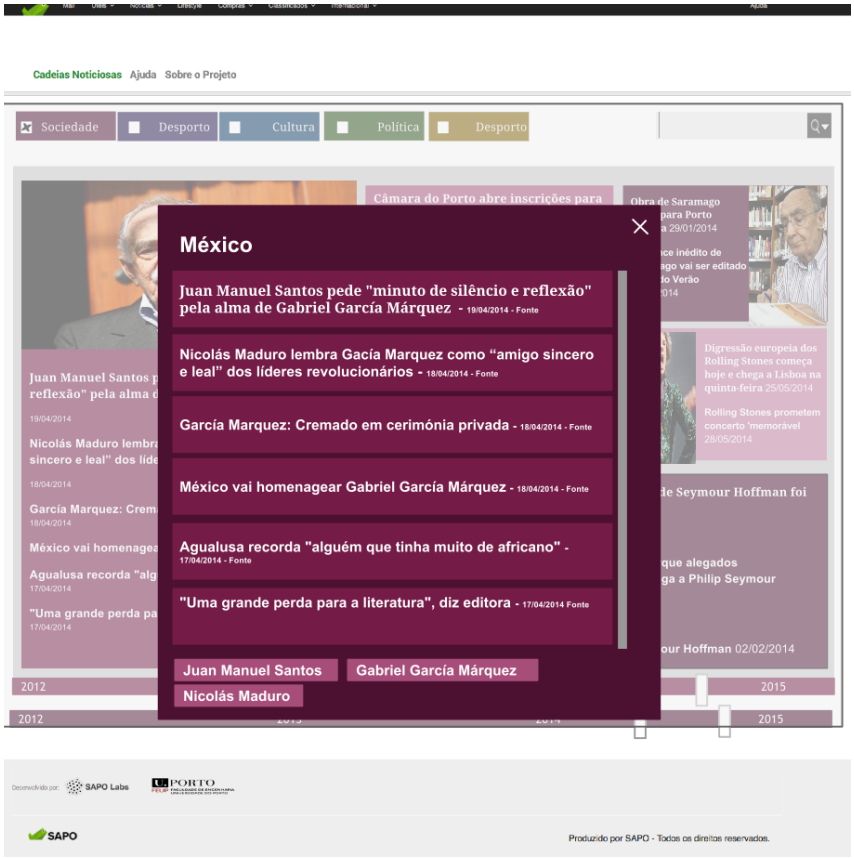
Local da notícia

<u>Centro:</u>

Blocos selecionáveis com os títulos das notícias (a seleção encaminha o utilizador para o website onde a notícia foi publicada)

Parte Inferior:

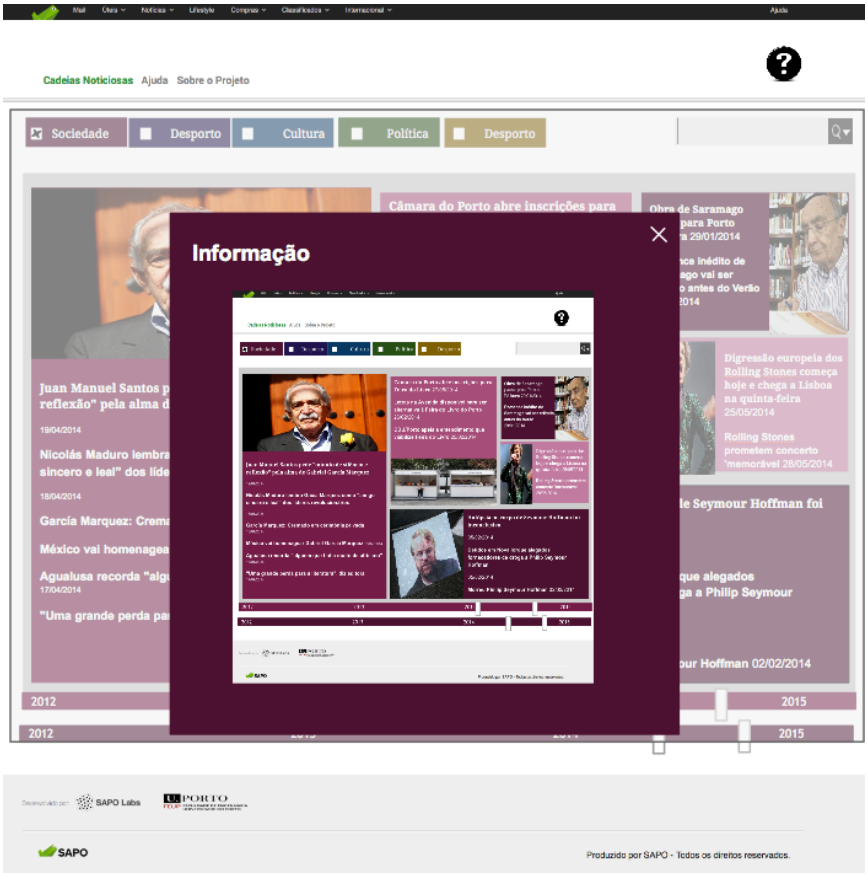
Área selecionável com o nome das personalidades (seleção encaminha para a máquina do tempo).



Interface - Popup Window

Popup window - Informação

Área com um esquema de utilização da página.



Apêndice E

Interface Final

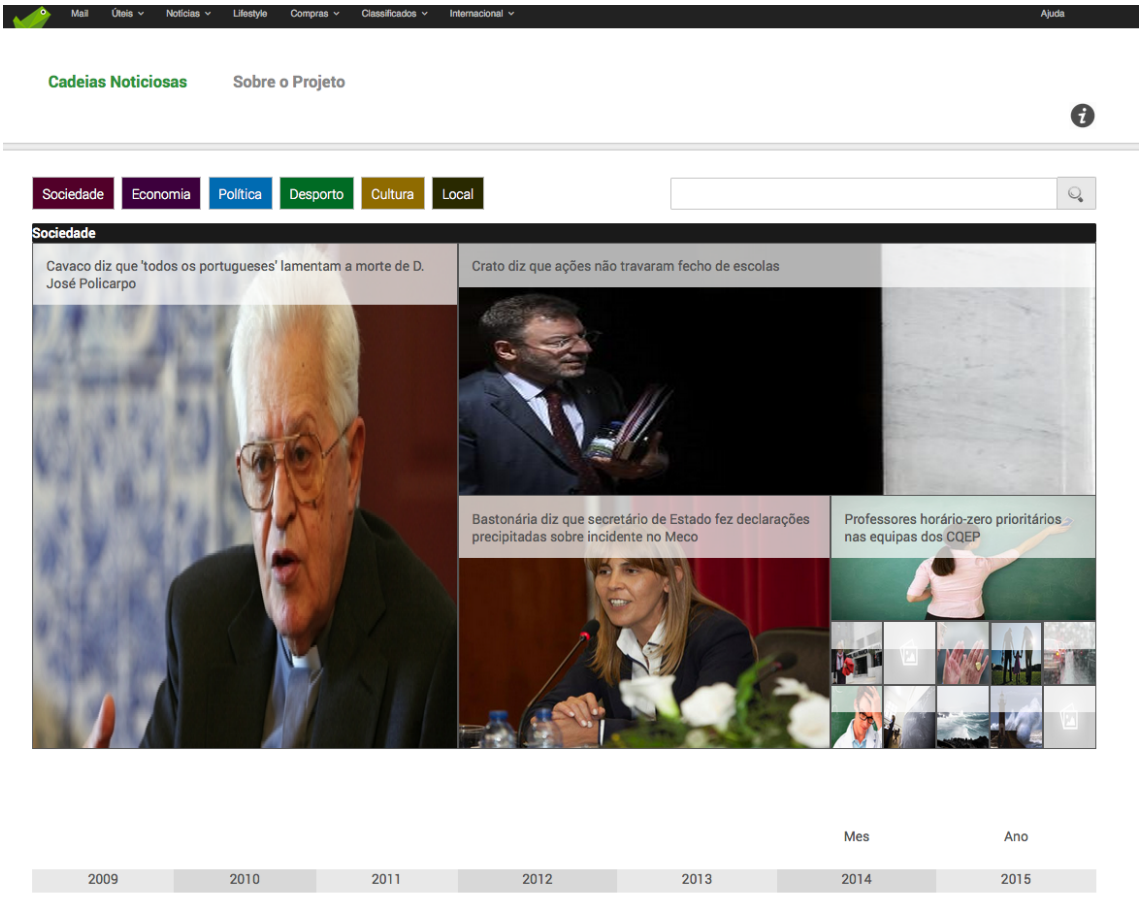


Figura E.1: Interface Final

Apêndice F

Documentação - Módulo Interface

Interface

A interface está dividida em duas componentes distintas: a componente da interface - que se refere à estrutura da interface e a componte flask - que tem como objectivo responder e fornecer dados à interface.

Ambos os compontes encontram-se no projeto ConnectingDots_Interface.

Os componentes relacionados com a estrutura da interface estão no directório ConnectingDots_Interface/InterfaceResources e os componentes relacionados com o flask estão no directório ConnectingDots_Interface/FlaskResources.

Para executar a interface:
Directório: ConnectingDots_Interface/InterfaceResources
Comando: python website.py PORTA

Variável PORTA: refere-se à porta onde a interface será disponibilizada.

Para executar o Flask:
Directório: ConnectingDots_Interface
Comando: python FlaskResources/flaskIni.py -p PORTA

Notas:

- A Interface está programada a obter os recursos do flask disponibilizados apenas na porta 8000
- Base de dados: Neo4j - porta 7474 (essencial para o funcionamento desta componente).

Requisitos:
webpy
flask
base de dados Neo4j

Apêndice G

Documentação - API's

API's

-getSimilarNews

O objetivo desta API é a determinação de notícias similares.

- getKeywords

O objetivo desta API é a extração de termos considerados relevantes.

-getChains

O objetivo desta API é a determinação de notícias similares; extração de termos relevantes, elaboração de agrupamentos tendo por base os pares de notícias consideradas como similares, associação dos termos relevantes aos respetivos agrupamentos, elaboração de ligações entre agrupamentos de notícias.

Requisitos:

Ambas as API's recebem como *input* o ficheiro config.json. Este ficheiro contém as notícias sobre as quais se pretende elaborar a ação e ainda as configurações genéricas. Este ficheiro segue a seguinte estrutura:

```
{
        "news":[
                        {
                                "content": …,
                                "id": …,
                                "pubdate": …,
                                "title": ...

                        };
                ….
                ],
        "parameters":
                "MethodBody": …
                "MethodSentence": …
                "MethodTitle": …
                "RatioSentece": ….
                "EndDate": ...
                "DeltaBody": ...
                "DeltaTime": ..
}
```

Onde o *news* contém a lista de notícias. As notícias possuem a seguinte estrutura:

- content: conteúdo da notícia;
- title: título da notícia;
- id: identificador da notícia;
- pubdate: data de publicação da notícia.

Quanto aos parâmetros, estes correspondem:

- MethodBody/Sentence/Title: que corresponde ao algoritmo a utilizar para o cálculo da distância entre o conteúdo da notícia, o primeiro paragrafo e o titulo respetivamente. Este campo pode assumir os seguintes valores: *levenshtein* (distância de *Levenshtein*); damerau (distância Damerau-Levenshtein); *jaro* (distância Jaro) ou *hamming* (distância Hamming).
- RatioSentence: corresponde ao rácio do conteúdo de notícia que é considerado como sendo o primeiro parágrafo.
- EndDate: Corresponde à data final até onde se pretende analisar as notícias
- DeltaTime: corresponde ao intervalo de tempo entre notícias permitido para se realizar o processo de similaridade.
- DeltaBody: valor mínimo de distância entre o conteúdo de duas notícias de forma a que as mesmas possam passar ao processo de calculo da similaridade.

Modo de Execução:

Modo de Utilização:

Correr o comando:

python webservices/zFlask.py

input:

- config.json
  - news
  - parameters

Opções:

- Obter notícias similares: sudo curl [http://127.0.0.1:5000]/getSimilarNews
- Extrair palavras chave: sudo curl [http://127.0.0.1:5000]/getKeywords
- Construir cadeias de notícias: sudo curl [http://127.0.0.1:5000]/getChains

Apêndice H

Documentação - Módulo de Geração Automática da Hierarquia

Hierarquia

Este é o módulo responsável pela geração automática da hierarquia. A hierarquia pode ser realizada de duas formas distintas, considerando que um tópico só pode pertencer a uma categoria ou considerando que um tópico pode ser transversal a várias categorias.

A abordagem utilizada para a geração automática de categorias baseia-se numa abordagem probabilística..

O módulo de geração automática de hierarquia encontra-se no projeto ConnectingDots_Hierarchy.

Execução do módulo:

Directório: ConnectingDots_Hierarchy
Comando: python hierarchy/createHierarchy.py "CATEGORIA" TIPO

onde,
CATEGORIA: refere-se à categoria sobre a qual se pretende obter a hierarquia. Pode tomar os seguintes valores: “Desporto”, “Sociedade”, “Local”, “Cultura”, “Economia”, “Política”

TIPO: refere-se a tipologia da hierarquia, ou seja, se um tópico pode pertencer a mais do que uma categoria. Esta variável pode tomar os seguintes valores “inter” quando um tópico pertence apenas a uma categoria e “all” quando um tópico pertence a mais do que uma categoria.

Nota: Este módulo requer a utilização de uma base de dados para obter as informações necessárias. Esta base de dados encontra-se na máquina 10.135.2.198 na base de dados “similar”.

Apêndice I

Documentação - Módulo de Encadeamento de notícias

Encadeamento de Notícias

Índice

Encadeamento de Notícias

Overview

O trabalho desenvolvido tem quatro componentes principais, são elas: determinação da similaridade e agrupamento das notícias (Figura 1), extração de termos chave, associação dos termos chave aos agrupamentos e estabelecimento de ligações entre agrupamentos (Figura 2).

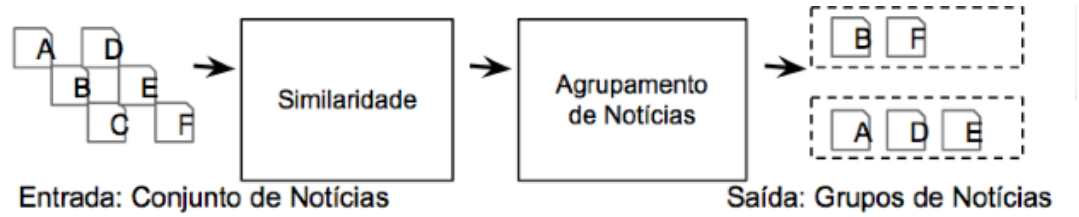


Figura 1: Similaridade e Agrupamento de Notícias

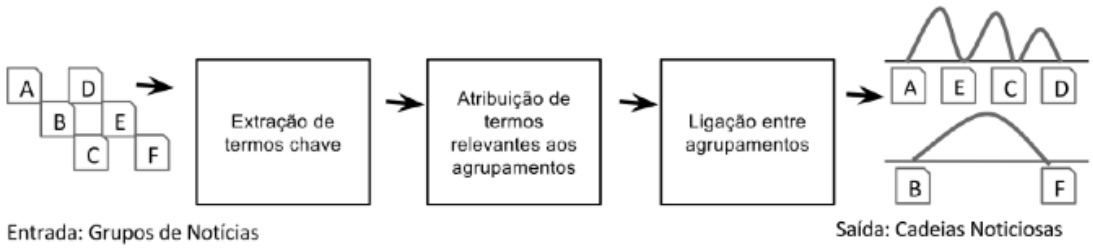


Figura 2: Extração de termos chave, associação dos termos aos agrupamentos, e estabelecimento de ligações entre agrupamentos.

Encadeamento de Notícias

Fluxo de Processamento

O fluxo de processamento do encadeamento de notícias é apresentado na Figura 1. No repositório o mesmo encontra-se dentro do projeto *ConnectingDots_Encadear* no ficheiro *script_diario.sh*.

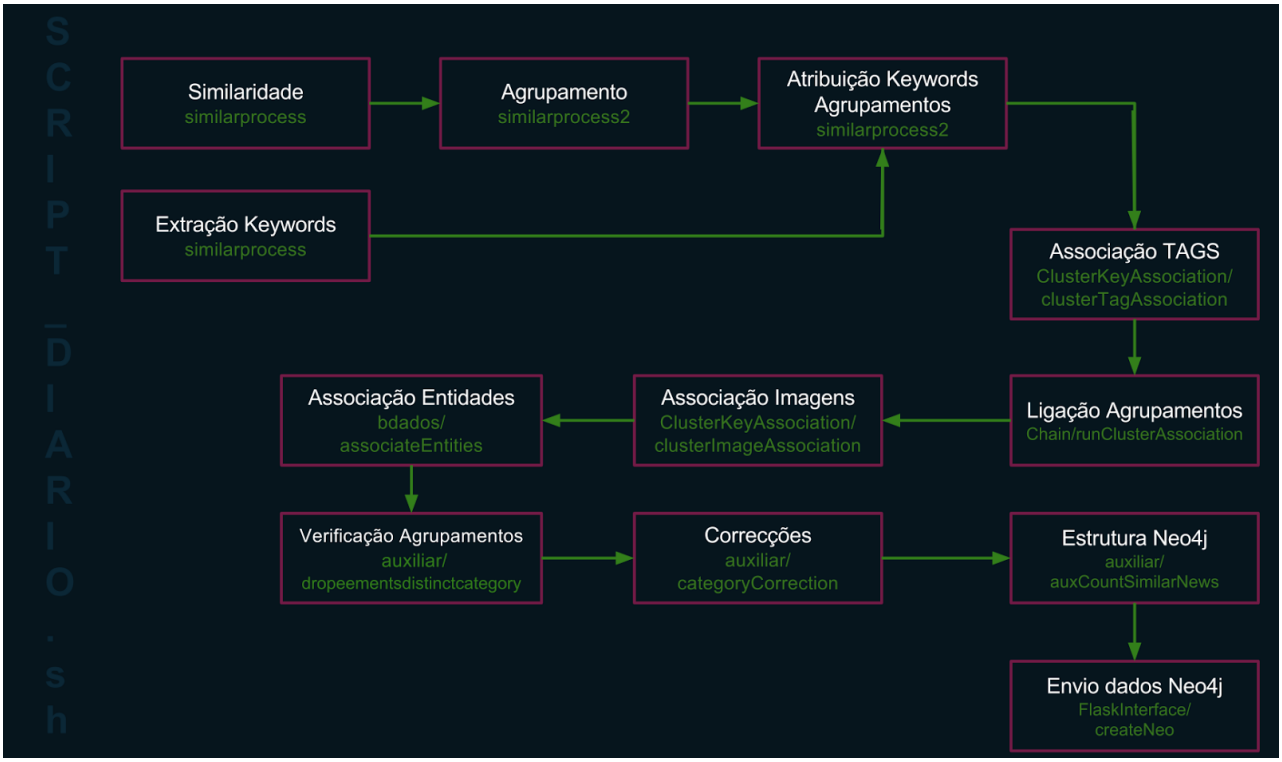


Figura 1: Fluxo de processamento

Encadeamento de Notícias

Requisitos Globais

Base de Dados

Notícias

A base de dados das notícias é composta pelas seguintes tabelas:

news: descrição da notícia.
- id - *int(11)* - identificador;
- pubdate - *datetime* - data de publicação;
- source - *varchar(64)* - origem;
- url - *text* - url;
- title - *text* - título;
- content - *mediumtext* - conteúdo;
- original_id - *int(11)* - identificador original;
- tags - *text* - tags associadas;
- image - *text* - link da imagem.

occurrences: ocorrência de entidades por notícia.
- id - *int(11)* - identificador da ocorrência;
- entity_id - *int(11)* - identificador da entidade;
- news_id - *int(11)* - identificador da notícia.

entities: decrição da entidade.
- id - *int(11)* - identificador da entidade;
- name_id - *int(11)* - identificador do nome da entidade.

names: nome da entidade.
- id - *int(11)* - identificador do nome;
- names - *varchar(256)* - nome da entidade.

Armazenamento

A base de dados para o armazenamento dos dados relativos ao encadeamento de notícias é composta pelas seguintes tabelas:

CDHierarchyCategories: Tópicos por categoria.

ID - *int(11)* - Identificador do par tópico por categoria;
category - *varchar(512)* - Categoria atribuídas pelos jornalistas às notícias;
categories - *varchar(512)* - Tópico a aparecer na hierarquia;
Status - *int(11)* - estado do par.

CDarchCategoryFinal: Categoria de uma ligação.

IDCluster1 - *int(11)* - identificador do agrupamento temporalmente mais antigo;
IDCluster2 - *int(11)* - identificador do agrupamento temporalmente mais recente;
category - *varchar(512)* - categoria das notícias associadas à ligação

CDarchCategoryValuesFinal: Categoria e valores obtidos pela ligação de dois elementos.

ID - *int(11)* - Identificador da ligação.
IDCluster1 - *int(11)* - identificador do agrupamento temporalmente mais antigo;
IDCluster2 - *int(11)* - identificador do agrupamento temporalmente mais recente;
category - *varchar(512)* - categoria das notícias associadas à ligação;
S1 - *float* - Valor da comparação dos termos chave simples;
S2 - *float* - Valor da comparação dos termos chave compostos;
S3 - *float* - Valor da comparação das entidades;
S4 - *float* - Valor da comparação das personalidades;
VAL - *int(11)* - Indica se a ligação foi ou não processada;
Status - *int(11)* - Estado da ligação.

CDcluster: Definição dos agrupamentos.

ID - *int(11)* - Identificador do agrupamento;
IdNews - *varchar(512)* - Lista com os identificadores das notícias;
Status - *int(11)* - Estado do agrupamento;
NumberElements - *int(11)* - Número de notícias associadas ao agrupamento;
PrincipalNews - *int(11)* - Identificador da notícia principal;
Date - *datetime* - Data inicial do agrupamento;
OldID - *int(11)* - Identificador do agrupamento noutra base de dados;
id_mdt - *int(11)* - Identificador da notícia principal (pela máquina do tempo);
idnews_mdt - *varchar(512)* - Lista com os identificadores de notícias (pela máquina do tempo);
oMDT - *int(11)* - Se os campos já estão de acordo com a máquina do tempo;

CDkeyword: Palavras-chave associadas às notícias.

ID - *int(11)* - Identificador da notícia;
expSimple - *varchar(512)* - Lista com os termos chave simples;
expComp - *varchar(512)* - Lista com os termos chave compostos;

entity - *varchar(512)* - Lista com as entidades;
bMDT - *int(11)* - Se o identificador da notícia já está de acordo com a máquina do tempo.

CDkeywordCluster: Palavras-chave associadas aos agrupamentos.
ID - *int(11)* - Identificador do agrupamento;
expSimple - *varchar(512)* - Lista de termos chave simples;
expComp - *varchar(512)* - Lista de termos chave compostos;
entity - *varchar(512)* - Lista de entidades;
size - *int(11)* - Número de notícias associadas ao agrupamento;
dateBegin - *datetime* - Data da ocorrência da primeira notícia;
dateEnd - *datetime* - Data da ocorrência da última notícia.

CDsimilar: Similaridade entre pares de notícias.
ID - *int(11)* - Identificador da comparação;
EXP - *varchar(255)* - Identificador da experiência;
Idnews1 - *int(11)* - Identificador da notícia 1;
Idnews2 - *int(11)* - Identificador da notícia 2;
Path - *varchar(255)* - Caminho utilizado na determinação da similaridade;
State - *varchar(255)* - Estado da comparação;
ST - *float* - Semelhança do título;
SB - *float* - Semelhança do primeiro parágrafo;
SC - *float* - Semelhança do conteúdo;

CDtagCluster: Associação de *tags*, imagens e entidades ao agrupamento.
ID - *int(11)* - Identificador do agrupamento;
tag - *varchar(512)* - lista de *tags* associadas ao agrupamento;
category - *varchar(512)* - Categoria do agrupamento;
Status - *int(11)* - Estado do agrupamento;
StatusArch - *int(11)* - Estado do agrupamento;
StatusArch1 - *int(11)* - Estado do agrupamento;
StatusArch2 - *int(11)* - Estado do agrupamento;
StatusArch3 - *int(11)* - Estado do agrupamento;
bImage - *int(11)* - Estado da imagem;
urlImage - *varchar(512)* - Url da imagem;
bEntity - *int(11)* - Estado da entidade;
listEntity - *varchar(512)* - Lista de entidades.

Acesso aos dados

Ambas as bases de dados são configuradas nos ficheiros *json*: *main_prod* (base de dados de produção) e *main_test* (base de dados de teste). O ficheiro *json* segue a estrutura

apresentada na Figura 1, onde *BDnews* se refere à base de dados que tem as notícias e *BDstorage* à base de dados que armazena a informação dos agrupamentos.

Figura 1: Ficheiro com as configurações das bases de dados.

Ficheiro de configuração das variáveis de comparação

O ficheiro de configuração dos parâmetros necessários para o módulo de similaridade denomina-se *config.json* e segue a estrutura apresentada na Figura 2.

Figura 2: Ficheiro de configuração para o método similaridade.

MethodBody Identificação do algoritmo de medição de distância pretendido para o cálculo da similaridade do conteúdo. Este parâmetro pode tomar os seguintes valores: l*evensthein, hamming* ou *jaro*.

MethodSentence Identificação do algoritmo de medição de distância pretendido para o cálculo da similaridade do foco da notícia. Este parâmetro pode tomar os seguintes valores: *levensthein, hamming* ou *jaro*.

MethodTitle Identificação do algoritmo de medição de distância pretendido para o cálculo da similaridade do título. Este parâmetro pode tomar os seguintes valores: *levensthein, hamming* ou *jaro*.

BeginDate Representa o inicio do intervalo temporal sobre o qual se quer efetuar o processamento de notícias

EndDate Representa o fim do intervalo temporal sobre o qual se quer efetuar o processamento de notícias.

DeltaTime Refere-se ao intervalo máximo de tempo que duas notícias estão aptas para comparação.

DeltaBody Corresponde ao valor de decisão mínimo que duas notícias devem ter na comparação do conteúdo para serem determinadas como similares.

DeltaSentence Corresponde ao valor de decisão mínimo que duas notícias devem ter na comparação do foco para serem determinadas como similares.

DeltaTitle Corresponde ao valor de decisão mínimo que duas notícias devem ter na comparação do título para serem determinadas como similares.

TreeTagger

É de notar que o sistema requer a instalação prévia do tree-tagger. A localização do mesmo deve ser incluída da variável TAGDIR (Figura 3).

```
#TAGGER - INICIALIZACAO
tagger = ttw.TreeTagger(TAGLANG='en',TAGDIR='/home/carlafabreu/tree-tagger', TAGPARFILE='pt.par')
```

Figura 3: TreeTagger

Python

timeneo; json; sklearn; datetime; random; time; neo4jrestclient; py2neo; flask; re; operator; math; cProfile; logging MySQLdb; jellyfish

Encadeamento de Notícias

Descrição dos Módulos Principais - similarprocess.py

Modo de execução:

1. python similarprocess.py
2. python similarprocess.py -h “2013-01-02 23:59:59”

No modo de execução 1, é processado o dia anterior de notícias. A base de dados utilizada é a base de dados de produção.

No modo de execução 2, é processado o dia anterior ao mencionado. É utilizada a base de dados de teste.

Requisitos:

Base de dados com as notícias
Base de dados para acesso e armazenamento de dados
Tree Tagger

Processamento:

O objetivo deste *script* é o de determinar a similaridade entre notícias extrair os termos chave.

Inicialização

São efetuadas as seguintes iniciações na função *ini*:

- definição do valor das variáveis existentes no ficheiro de configuração (*config.json*);
- carregamento e inicialização o modelo de classificação (o modelo encontra-se no diretório *similarity/svc_lastest.model*);
- iniciação do *treetagger*;

- obtenção e normalização das notícias (esta normalização é dada pela chamada à função *methods.addfirstsentence* e consiste na: redução das palavras ao seu radical, exclusão das *stop-words* e obtenção do primeiro parágrafo).

Cálculo e determinação da similaridade

A parte do código que chama a função similaridade, *run_similarity*, pode ser observada na Figura 4.

```
start_time = time.time()
logging.warning('src/similarprocess.py: Calculo da similaridade')
run_similarity()
algTime =  time.time() - start_time
str_tempo = '>>>>>>>>>>>>>>>>>>> %s'%(str(algTime))
logging.warning(str_tempo)

start_time = time.time()
```

Figura 4. Chamada ao módulo similaridade

A função responsável pela comparação de notícias denomina-se *comparenews* e pode ser encontrada no diretório similarity/methods.

```
methods.comparenews(clf1, X, Y, nameExpI, ind[0])
```

Figura 5. Chamada ao método *comparenews* responsável pela comparação e determinação da similaridade entre notícias.

O método *comparenews* (Figura 5) recebe como argumento os seguintes parâmetros:

- clf1 - classificador;
- X - caracterização dos elementos que compõe o modelo de classificação;
- Y - estado dos elementos que compõe o modelo de classificação (se são ao não similares);
- nameExpI - nome da experiência;
- ind[0] - notícia a analisar.

O método *comparenews*, encontra-se em *methods.py* no diretório /*similarity*, tem como objetivo comparar as notícias e determinar se as mesmas são ou não similares. Esta comparação tem em conta um fator temporal predefinido. As notícias consideradas como similares são armazenadas na tabela *CDsimilar*.

Extração de Palavras-Chave

A chamada à extração das palavras-chave pode ser observada na Figura 6. Na chamada à função são passados como argumentos:

- d - data
- f - nome do ficheiro.

```
start_time = time.time()
logging.warning('src/similarprocess.py: Extracção das Keywords')
try:
    run_keywords(d, f)
except:
    print 'Erro na extracção de keywords'
algTime =  time.time() - start_time
str_tempo = '>>>>>>>>>>>>>>>>>>> %s'%(str(algTime))
logging.warning(str_tempo)
```

Figura 6. Chamada ao módulo extração de palavra-chave

A extração de palavras-chave propriamente dita é efetuada pelo *script* run que se encontra no diretório *kys*. Em *similarprocess2*, a chamada a este script dá-se através de *runkey.run* (e pode ser observada na Figura 7).

```
res = runkey.run(reslasthour, tagger, id_news_last_hour, d[0], f)
```

Figura 7. Extração de palavras-chave

A função responsável pela extração de palavras chave requer as seguintes variáveis:

- reslasthour - *array* com os elementos processados da última hora;
- tagger - *tree tagger*;
- id_news_last_hout - identificador das notícias da última hora;
- d[0] - data das notícias sobre as quais se pretende fazer a extração de palavras-chave (formato “AAAA-MM-DD”);
- f - ficheiro onde serão armazenadas temporalmente as palavras-chave.

A função *run*, encontra-se em run.py no diretório /*kys*. A função *run* tem como objectivo receber as notícias e retornar as palavras-chave associadas. O fluxo de processamento deste módulo é o seguinte:

getNews - Responsável por receber as notícias, normalizar o seu conteúdo, criar quatro listas de expressões: uma de palavras simples associadas ao título, outra ao conteúdo; uma de expressões compostas associadas ao título e outra ao conteúdo;

matrixDocumentWords.Matrix - Dado um conjunto de notícias e um conjunto de palavras, esta função tem como objetivo criar uma matriz de ocorrências: palavras por documento.

getExplicitKeywords - Para cada um dos elementos existentes: palavras simples do título e do conteúdo e expressões compostas do título e do conteúdo, calcula o *TF-IDF*, e determina consoante o resultado obtido e a tipologia das palavras se as mesmas estão aptas de serem consideradas palavras-chave.

No final deste módulo as palavras-chave obtidas são armazenadas na base de dados na tabela *CDkeywords*.

Encadeamento de Notícias

Descrição dos Módulos Principais - similarprocess2.py

Modo de execução:

1. python similarprocess2.py
2. python similarprocess2.py -h “2013-01-02 23:59:59”

No modo de execução 1, é processado o dia anterior de notícias. A base de dados utilizada é a base de dados de produção.

No modo de execução 2, é processado o dia anterior ao mencionado. É utilizada a base de dados de teste.

Requisitos:

Iguais ao *similarprocess.py*

Processamento:

Este *script* requer que o script anterior tenha sido executado, pois opera sobre os resultados obtidos no anterior. Este *script* contém dois módulos distintos, são eles: a formação de agrupamentos (que é realizado com base nas notícias consideradas como similares) e a atribuição de palavras-chave aos agrupamentos (que tem como base as palavras-chave extraídas das notícias e ainda as notícias que compõe os agrupamentos).

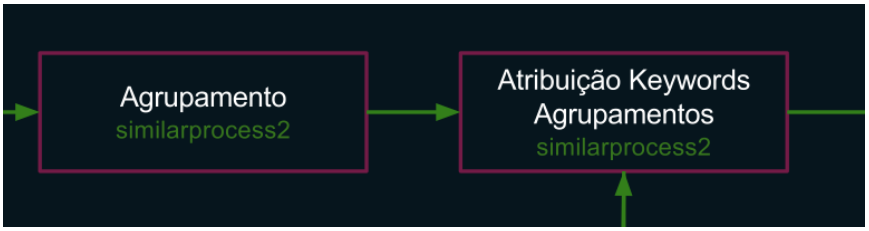


Figura 1. Módulos contidos no *script* similarprocess2.py

Agrupamentos

Para se efetuar a formação de agrupamentos é chamada a função *run_cluster*, que pode ser visualizada na Figura 2. Esta função é responsável por:

```
start_time = time.time()
logging.warning('src/similarprocess.py: Agrupamento de notícias similares')
try:
    run_clusters()
except:
    print 'Erro no agrupamento de noticias'
algTime =  time.time() - start_time
str_tempo = '>>>>>>>>>>>>>>>>>>> %s'%(str(algTime))
logging.warning(str_tempo)
```

Figura 2. Chamada à função formação de agrupamentos.

- Obter o resultado da similaridade para o dia que estamos a processar (*"Select Idnews1, Idnews2 from CDsimilar where (%s) AND (%s)"*);
- Efetuar os agrupamentos (cluster.processSimilarNews(arrayElems) - processo recursivo que a partir de uma notícia vai verificar todas as junções possíveis).

O resultado dos agrupamentos é armazenado na tabela *CDcluster*.

Atribuição de Palavras-chave Agrupamentos

A associação de palavras-chave a agrupamentos requer que os agrupamentos se encontrem formados e que as palavras-chave tenham sido atribuídas às notícias. A chamada ao módulo responsável pela extração de palavras-chaves é dada pela função run_keywordCluster(). Esta função tem como objetivo:

- Obter os agrupamentos que se querem processar (Select distinct ID from CDcluster where Date between "%s" and "%s" '%(str(d_last_day), str(d)) - tendo por base as notícias que compõe o intervalo pretendido;
- Fazer a atribuição das palavras-chave consoante a sua importância. A atribuição é efetuada através da chamada a cluster_key_association(b, clusterID, clusterIDE, stringQuery), onde:
    - b - assume o valor de *True* ou *False* consoante o que se quer processar, se uma lista predefinida de agrupamentos ou os agrupamentos da última hora, respetivamente;
    - ClusterID e clusterIDE - *string* com a lista de identificadores de agrupamentos a processar
    - stringQuery - *query* para obtenção dos agrupamentos do último dia;

  De uma forma genérica, este módulo efetua os seguintes passos:
    - Obtenção dos agrupamentos do último dia;

- Obtenção das palavras-chave atribuídas às notícias do último dia;
- Verificação do número de ocorrências de cada palavra no agrupamento (consoante o número de notícias do agrupamento a que a mesma está associada);
- Ordenação das palavras-chave no agrupamento consoante a sua importância.

Encadeamento de Notícias

Descrição dos Módulos Principais - ClusterKeyAssociation/clusterTagAssociation.py

---

Modo de execução:

1. python ClusterKeyAssociation/clusterTagAssociation.py
2. python ClusterKeyAssociation/clusterTagAssociation.py “2013-01-02 23:59:59”

No modo de execução 1, é processado o dia anterior de notícias. A base de dados utilizada é a base de dados de produção.

No modo de execução 2, é processado o dia anterior ao mencionado. É utilizada a base de dados de teste.

Processamento

Este módulo é responsável pela atribuição das *Tags* associadas pelos jornalistas às notícias aos agrupamentos existentes.

Associação das Tags

A associação das *tags* dá-se através da chamada à função *run_tag* que recebe como argumento a data inicial e final do intervalo a considerar. Para além da associação das *tags* aos agrupamentos esta função verifica, consoante as *tags* atribuídas qual é a categoria do grupo (se por exemplo: Desporto, Economia ou Politica).

Encadeamento de Notícias

Descrição dos Módulos Principais - Chain/runClusterAssociation.py

---

Modo de execução:

1. python Chain/runClusterAssociation.py

Processamento

Este módulo é responsável pelo encadeamento de notícias. O encadeamento de notícias é obtido através da comparação dos agrupamentos não processados com todos os outros que ocorreram dentro da mesma categoria.

Inicialização

Inicialmente são obtidos os seguintes elementos:

- Lista de personalidades (personalidades existente (personalities.getPersonalities());
- Lista de categorias que tem agrupamentos por processar (getCategory());
- Carregado o modelo de classificação para a formação de ligações (loadTrainSet());

Encadeamento de Notícias

Para cada categoria individual, o encadeamento de notícias é efetuado da seguinte forma:

1. getClusterKeywords - obtenção das palavras-chave associadas a cada agrupamento
    a. Obter as palavras-chave;
    b. Criar um dicionário com as palavras e o seu número de ocorrência no agrupamento;
    c. das palavras-chave compostas e das entidades obter as personalidades;
2. clusterAssociation - fazer uma associação de agrupamentos, ou seja, criar a lista de pares de agrupamentos a comparar;
3. compareKeys - comparar as palavras-chave entre os pares de agrupamentos;
4. chain - consoante os valores obtidos na comparação de pares de agrupamentos, verificar se o par é ou não passível de ser uma ligação.

Os resultados das ligações são armazenados em duas tabelas: *CDarchCategoryFinal* e *CDcarchCategoryValuesFinal*.

Encadeamento de Notícias

Descrição dos Módulos Principais - CusterKeyAssociation/clusterImageAssociation.py

---

Modo de execução:

2. python ClusterKeyAssociation/clusterImageAssociation.py

Processamento

Responsável pela atribuição do *URL* das imagens aos agrupamentos. Para todos os agrupamentos sem imagem associada, faz a atribuição de *URL*s válidos.

Os resultados provenientes desta etapa são armazenados na tabela *CDtagCluster.*

Encadeamento de Notícias

Descrição dos Módulos Principais - bdados/associateEntities.py

---

Modo de execução:

3. python bdados/associateEntities.py

Processamento

Responsável pela associação de entidades aos agrupamentos.

Os resultados provenientes desta etapa são armazenados na tabela *CDtagCluster*.

Encadeamento de Notícias

Descrição dos Módulos Principais - auxiliar/dropelementdistinctcategory.py

---

Modo de execução:

4. python ClusterKeyAssociation/clusterImageAssociation.py

Processamento

Responsável pela eliminação de ligações em que pelo menos um dos agrupamentos envolvidos tenha mais de que uma categoria associada.

Os resultados provenientes desta etapa são armazenados na tabela *CDarchCategoryValuesFinal*.

Encadeamento de Notícias

Descrição dos Módulos Principais - auxiliar/auxCountSimilarNews.py

---

Modo de execução:

5. python auxiliar/auxCountSimilarNews.py

Processamento

Responsável pela criação de dois ficheiros *json*, um com a descrição das ligações existentes e outro com a descrição dos agrupamentos. Estes ficheiros são criados para numa fase posterior se enviarem os dados para a base de dados *neo4j*.

Os ficheiros gerados nesta etapa denominam-se *arch_nodes.json* e *arch_rel.json*.

Encadeamento de Notícias

Descrição dos Módulos Principais - FlaskInterface/createNeo.py

---

Modo de execução:

6. python FlaskInterface/createNeo.py

Pré-requisitos

Este módulo requer que o *Flask* tenha sido inicializado (comando: python FlaskInterface/flaskIni.py PORTA).

Processamento

Responsável pela leitura dos ficheiros *json* criados no módulo anterior (*arch_nodes.json* e *arch_rel.json*) e envio dos dados para a base de dados *neo4j*.