

3^o ano da Licenciatura em Ciências da Computação

Projeto NetLang

Jorge Brandão Gonçalves
(a84585)

Pedro Pinheiro
(a85181)

Constança Elias
(a83543)

Maria Barbosa
(a85290)

3 de Setembro de 2021

Resumo

Este trabalho foi desenvolvido no âmbito de um projeto internacional de investigação suportado pela FCT, Netlang (<https://sites.google.com/site/projectnetlang/home>) e enquadrado na Unidade Curricular "Projeto" da licenciatura em Ciências da Computação. Trata-se da criação de corpora (em PT/EN) anotado que permita estudar formas de insulto e discriminação (diálogos em termos socialmente não-admissíveis) extraídos de meios usados para Comunicação Mediada por Computador (jornais online e redes sociais). Pretende-se, para isso, criar uma ferramenta que permita analisar uma coleção de posts e respetivos comentários, extraídos de fontes diversas, mas representados num formato standard em JSON. No final deve ser fornecida uma série de estatísticas sobre o uso de um conjunto de palavras reservadas que permitam posteriormente classificar o tipo de injúrias encontradas (racistas, sexistas, machistas, etc.).

Área de Aplicação: Criação e Catalogação de Repositórios de Conhecimento

Agradecimentos

Este trabalho não seria possível sem a proposta do professor Pedro Rangel Henriques e da professora Cristiana Araújo, a quem agradecemos a ajuda e o entusiasmo com que nos cativaram para o projecto. Gostaríamos também de prestar o nosso agradecimento ao Engenheiro Aragão, do grupo de técnicos do Departamento de Informática, pela disponibilidade imediata para ajudar em todos os problemas técnicos que tivemos com o servidor e pela sempre rápida resposta aos pedidos feitos.

Conteúdo

1	Introdução	4
1.1	Contextualização	4
1.2	Estrutura do relatório	4
2	Análise e Especificação	5
2.1	Descrição informal do problema	5
2.2	Especificação de Requisitos	5
2.3	Diagramas de Arquitetura e Flowchart	6
2.3.1	Flowchart	6
2.3.2	Diagrama Geral da Arquitetura	6
3	Concepção da Resolução	14
3.1	Estruturas de Dados	14
3.1.1	Dicionários	14
3.2	Implementação	14
4	Codificação, Testes e Resultados	18
4.1	NetLang Analyzer	18
4.2	Preenchimento do ficheiro JSON	20
5	Alternativas, Decisões e Problemas de Implementação	21
5.1	Decisões	21
5.2	Problemas de Implementação	21
5.2.1	Problema do Encoding	21
5.2.2	Problemas dos Smiles	22
5.2.3	Problema das percentagens.	23
5.2.4	Problema no Match	23
5.2.5	Problema de falsos positivos	24
6	Conclusão	25
A	Reuniões e trabalho desenvolvido	26
B	Explicação do Json	29
C	Compilar os módulos pelo servidor	32

Lista de Figuras

2.1	Flowchart	7
2.2	Diagrama de Arquitetura Geral	7
2.3	Diagrama <i>Analyzer and Classifier</i>	8
2.4	Diagrama <i>Analyzer</i>	9
2.5	Diagrama Update Dictionary	10
2.6	Diagrama View Dictionary	11
2.7	Diagrama Novo Idioma	12
2.8	Diagrama Upload de ficheiros	13
3.1	Dicionário inglês (fragmento)	15
3.2	Tabela que efectua a análise de cada comentário no Post.	16
3.3	Tabela que apresenta uma síntese dos resultados obtidos para cada uma das variáveis sociolinguísticas presentes	16
3.4	Análise global do ficheiro	17
4.1	Página inicial do site	18
4.2	Página que permite efectuar a análise de um ou mais ficheiros Json	19
4.3	Página que permite consultar o dicionário numa tabela	19
4.4	Página que permite adicionar ou remover keywords dos dicionários	19
4.5	Comentário modificado num ficheiro JSON (fragmento)	20
5.1	Erro de Enconding.	22
B.1	Header do ficheiro Json	30
B.2	Comment thread do ficheiro Json (fragmento)	31

Capítulo 1

Introdução

1.1 Contextualização

Este trabalho foi desenvolvido para integrar o projeto NetLang e realizado como projecto final de curso.

O grupo decidiu escolher este tema não só por ser desafiante como também pela sua elevada utilidade. O bullying é usado há centenas e possivelmente milhares de anos, desde que a internet começou a ser consumida mundialmente. Espalhou-se como o fogo e por isso o seu controlo tornou-se mais difícil. Todos os passos que possam a vir ser utilizados para detetar este problema são bem-vindos.

O objetivo principal deste projeto é a criação de uma ferramenta que, recebendo um ficheiro json com comentários associados a um vídeo ou um post, capte o *hate speech* presente e efetue a sua análise. Complementou-se esta ferramenta com a criação de um site, que facilita a visualização dos dados tratados.

Deixamos a ideia de, futuramente, a nossa ferramenta ser implementada em todos os sites e redes sociais da universidade através de um *bot* que reporte na plataforma as ocorrências de discursos de ódio.

1.2 Estrutura do relatório

Este relatório divide-se em cinco partes principais:

- Na primeira parte, fazemos uma análise do problema, dos requisitos e da arquitetura da ferramenta.
- Na segunda parte, é explicada a implementação do projeto e o trabalho desenvolvido.
- A terceira corresponde à explicação da interface web e dos resultados obtidos.
- A quarta aborda as decisões tomadas e os problemas de implementação.
- Na última parte é feita uma conclusão do trabalho realizado.

O documento contém ainda três anexos: um com a cronologia do trabalho realizado e das reuniões efetuadas; outro com uma explicação detalhada dos ficheiros JSON usados na nossa ferramenta, de modo a ser possível gerar ficheiros semelhantes; e um último com a apresentação dos comandos necessários para correr os módulos do nosso programa pelo servidor.

Capítulo 2

Análise e Especificação

2.1 Descrição informal do problema

Este projeto surge da necessidade da criação de uma ferramenta que, dado um dicionário de palavras relacionadas com *hate speech* e um conjunto de ficheiro com comentários, reporte a análise detalhada do discurso de ódio encontrado. Esta deverá fornecer um relatório, em formato *pdf*, com a síntese das informações recolhidas sobre os casos de *hate speech* presentes nos ficheiro analisados, para posterior análise por parte de linguistas.

2.2 Especificação de Requisitos

Neste trabalho pretende-se:

- Desenvolver um analisador léxico, recorrendo à linguagem de programação **Python**, de modo a "apanhar" o hate speech.
- Analisar os dados recolhidos, efectuando os cálculos necessários de modo a identificar o tipo de preconceito presente no documento.
- Desenvolver uma interface web que permita visualizar os resultados obtidos de forma mais simples e intuitiva.

2.3 Diagramas de Arquitetura e Flowchart

2.3.1 Flowchart

A figura 2.1, que apresenta o flowchart do projeto, indica as funcionalidades que existem no nosso programa.

A funcionalidade mais importante do site, e que implementa o objectivo do projeto, é a análise dos ficheiros json. É possível efetuar uma análise total e uma análise por variável sociolinguística. Na primeira podemos escolher um ou mais ficheiros json e analisá-los conforme o idioma escolhido; no final é devolvido um pequeno sumário da análise assim como um pdf com os resultados totais e os ficheiros json preenchidos. Na análise sociolinguística escolhem-se não só os ficheiros json como também uma variável sociolinguística, sendo apenas efetuada uma análise de acordo com as *keywords* desta variável; no fim é devolvido um pequeno sumário sobre os resultados.

Outra funcionalidade existente é possibilidade de gestão das keywords existentes no dicionário de entrada. De modo a facilitar a posterior utilização da ferramenta, achámos importante ter uma forma simples de adicionar e retirar palavras do dicionário. Sendo assim, foi criado um módulo que recebe a acção a realizar, uma palavra passe (diferente para adicionar e remover) e os dados sobre a keyword a adicionar ou retirar (o tipo de preconceito e a variável sociolinguística), e coloca-a no lugar correspondente do dicionário.

Para facilitar a posterior análise de resultados, introduziu-se a possibilidade de consultar os dicionários. Esta funcionalidade, apesar da simples implementação, fornecerá uma grande ajuda aos utilizadores para analisar os dados que a ferramenta lançou. Esta permite consultar as palavras que se encontram registadas como *hate speech*, sendo possível verificar, por exemplo, palavras erradamente adicionadas ou removidas. Deste modo, a acção "ver dicionário" apresenta o dicionário em forma de tabela.

Tendo em conta a implementação da acção anterior, o passo seguinte é permitir que se possam adicionar mais idiomas à ferramenta, de forma a esta poder abranger um maior número de utilizadores. Para isso, foi criada uma função que recebe uma string com o idioma que se pretende adicionar e, caso este idioma exista numa lista de línguas criada por nós e não estiver ainda registado na plataforma, cria o dicionário para essa língua. Este dicionário pode ser posteriormente preenchido através da funcionalidade explicada anteriormente.

Por fim, de modo a proteger a importância e o acesso global à ferramenta, colocámos uma explicação concisa dos nossos ficheiros json de modo a que possam ser feitos extractores fora da ferramenta para criar novos ficheiros JSON com o objetivo de serem submetidos ao analisador. Para isso, foi adicionado ao site um último separador que dá ao utilizador a possibilidade de fazer upload de ficheiros para o servidor.

2.3.2 Diagrama Geral da Arquitetura

O diagrama 2.2 apresenta uma visão global das funcionalidades do projeto, isto é, dado um input e uma ordem, chama a funcionalidade correspondente e devolve um resultado ao utilizador.

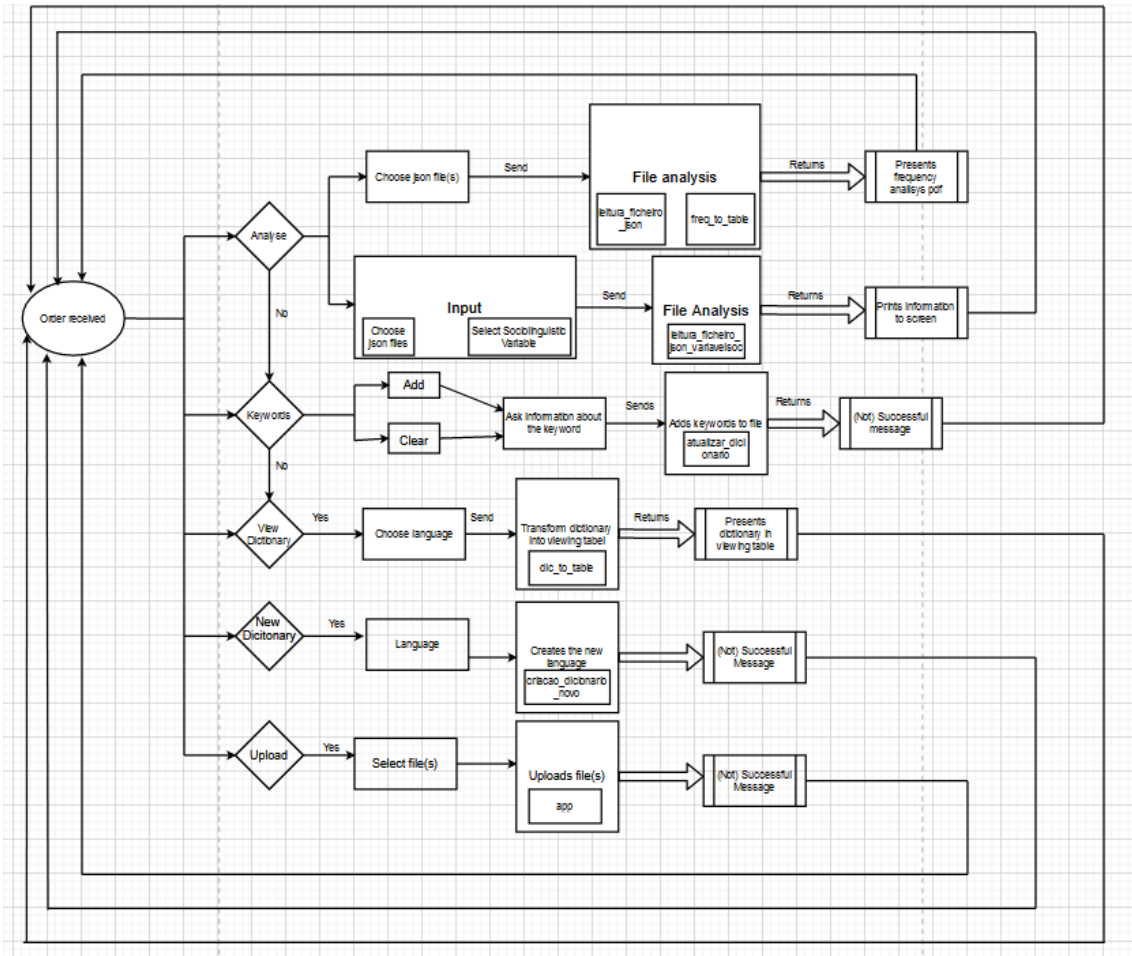


Figura 2.1: Flowchart

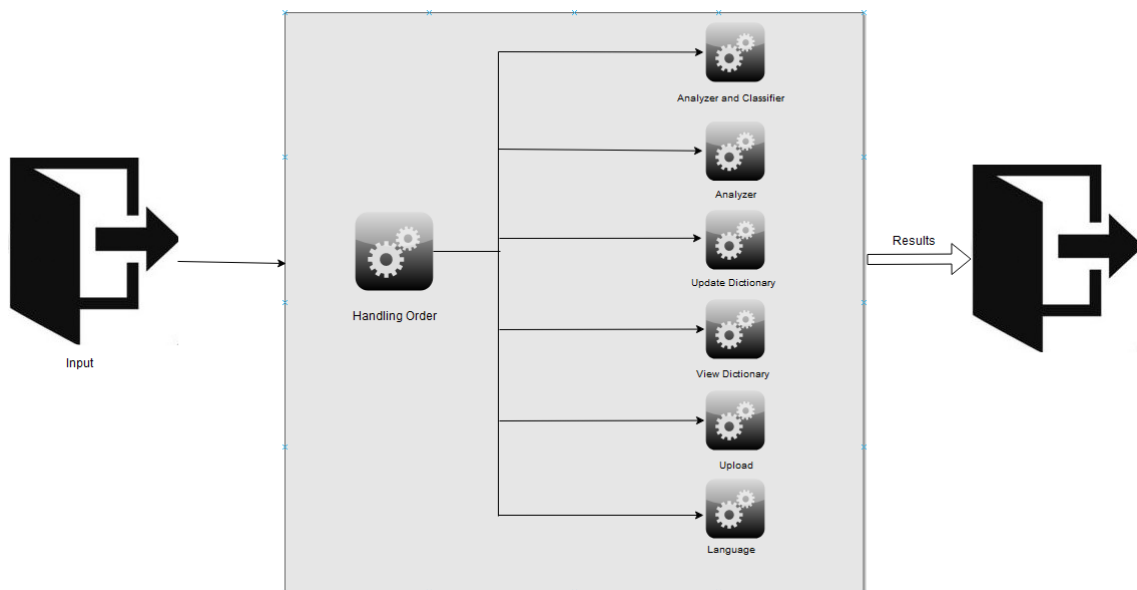


Figura 2.2: Diagrama de Arquitetura Geral

Diagrama do Analisador Geral

Este programa, como podemos ver no diagrama 2.3, recebe um idioma, que será usado para obter o dicionário correto, e um conjunto de ficheiros json a analisar através do módulo *leitura_ficheiro_json*. O módulo *freq_to_table* é também usado, no final da análise, para criar um ficheiro em pdf com os resultados obtidos. No fim, este ficheiro em pdf é apresentado ao utilizador.

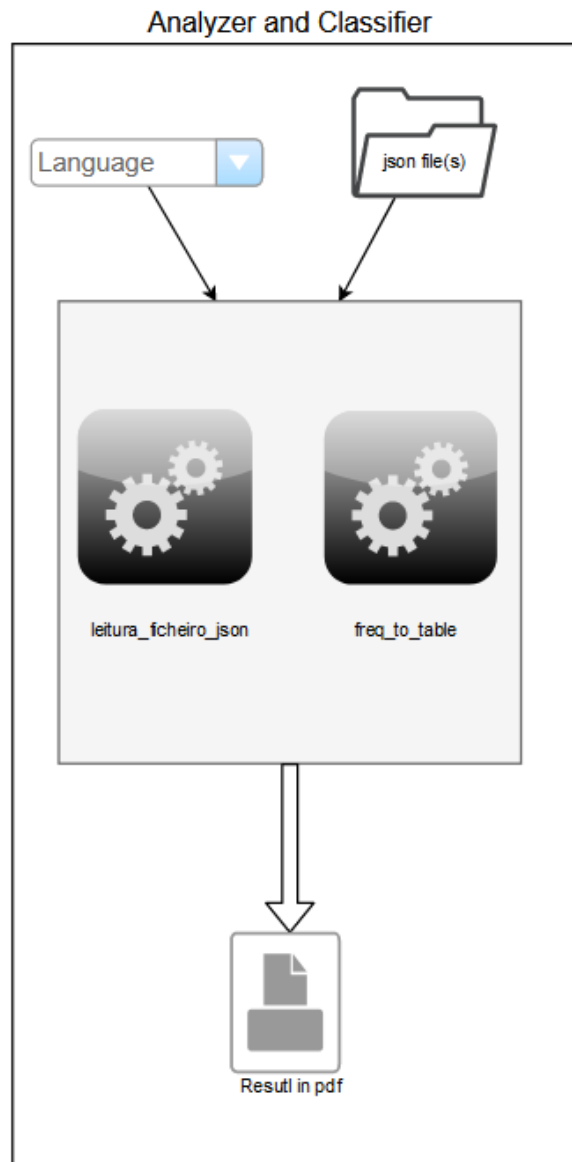


Figura 2.3: Diagrama *Analyzer and Classifier*

Diagrama do Analisador Sociolinguístico

O diagrama 2.4 mostra a arquitetura do programa que efectua a análise por variável sociolinguística. Este recebe um idioma, que será usado para obter o dicionário correto, uma variável sociolinguística e um conjunto de ficheiros json que irão ser analisados através do módulo *leitura_ficheiro_json_variaveisSoc*. O módulo *freq_to_table* é também usado no final da análise de modo a criar o ficheiro em pdf com os resultados obtidos, ficando este disponível ao utilizador.

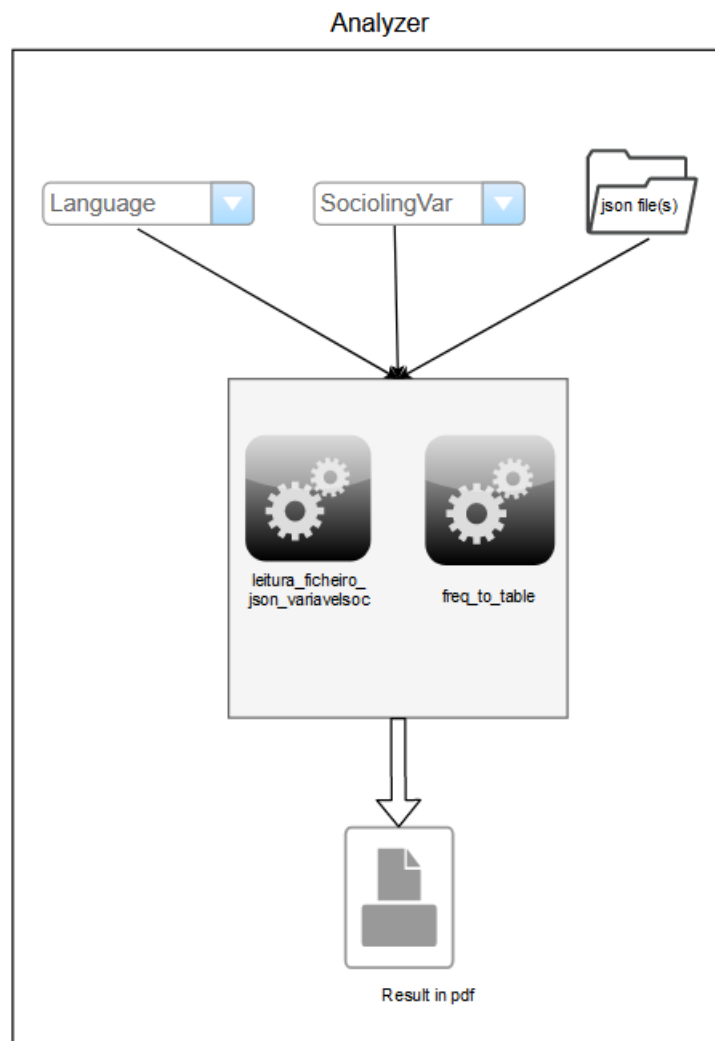


Figura 2.4: Diagrama *Analyzer*

Atualizar Dicionário

No diagrama 2.5 podemos ver que, numa primeira fase, é necessário receber uma ação, que poderá ser adicionar ou remover uma palavra, e uma palavra passe. Se a palavra passe estiver correta, será pedido para preencher toda a informação sobre a palavra a adicionar/remover e, usando o módulo *atualizar_dicionario*, o dicionário escolhido será atualizado.

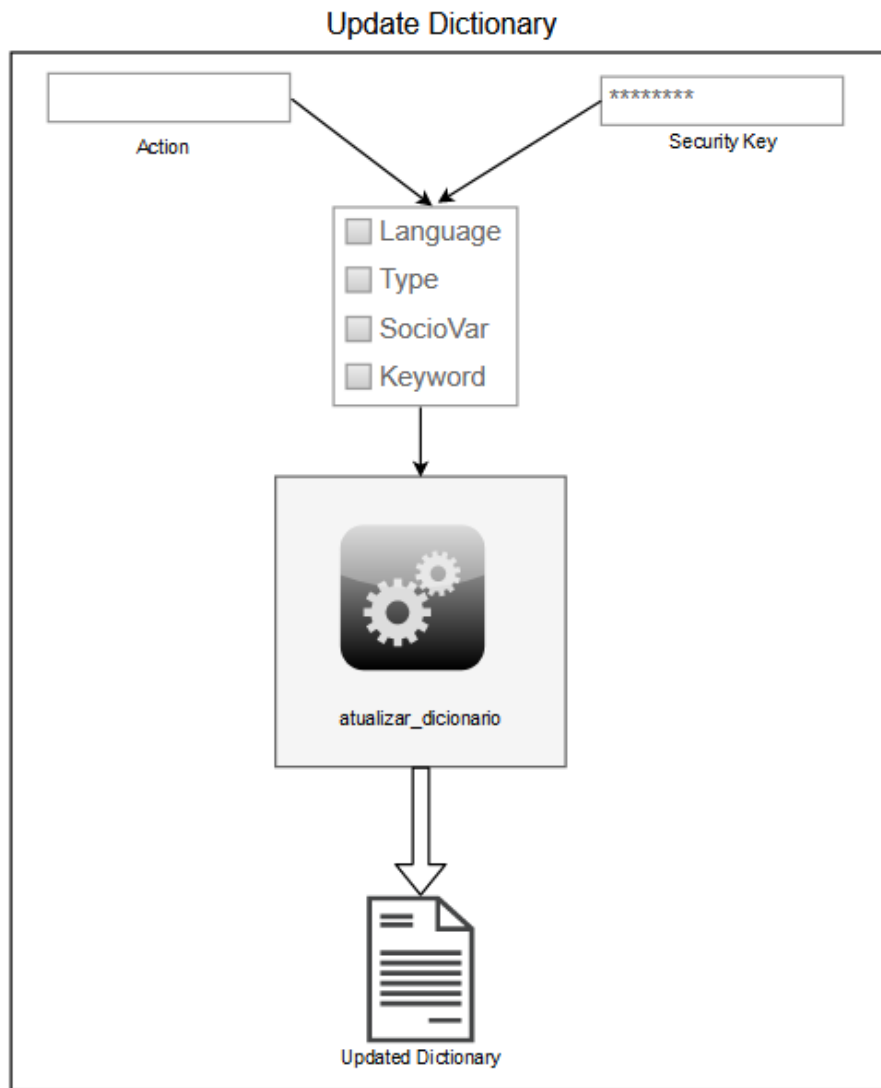


Figura 2.5: Diagrama Update Dictionary

Ver Dicionário

Esta funcionalidade, detalhada no diagrama 2.6, recebe um idioma, que será usado para obter o dicionário correto, e através do módulo *dict_to_table* é mostrado, no ecrã, o dicionário na forma de uma tabela.

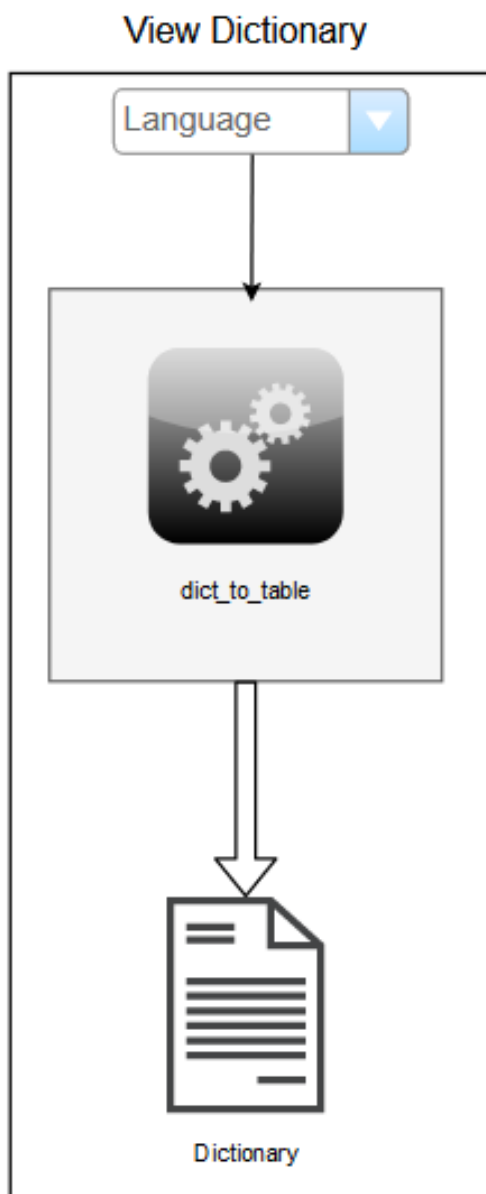


Figura 2.6: Diagrama View Dictionary

Novo Idioma

Esta funcionalidade, como podemos ver no diagrama 2.7, vai receber um idioma, que será usado para criar um dicionário vazio para este novo idioma, através do módulo *criação_Dicionario_Novo*.

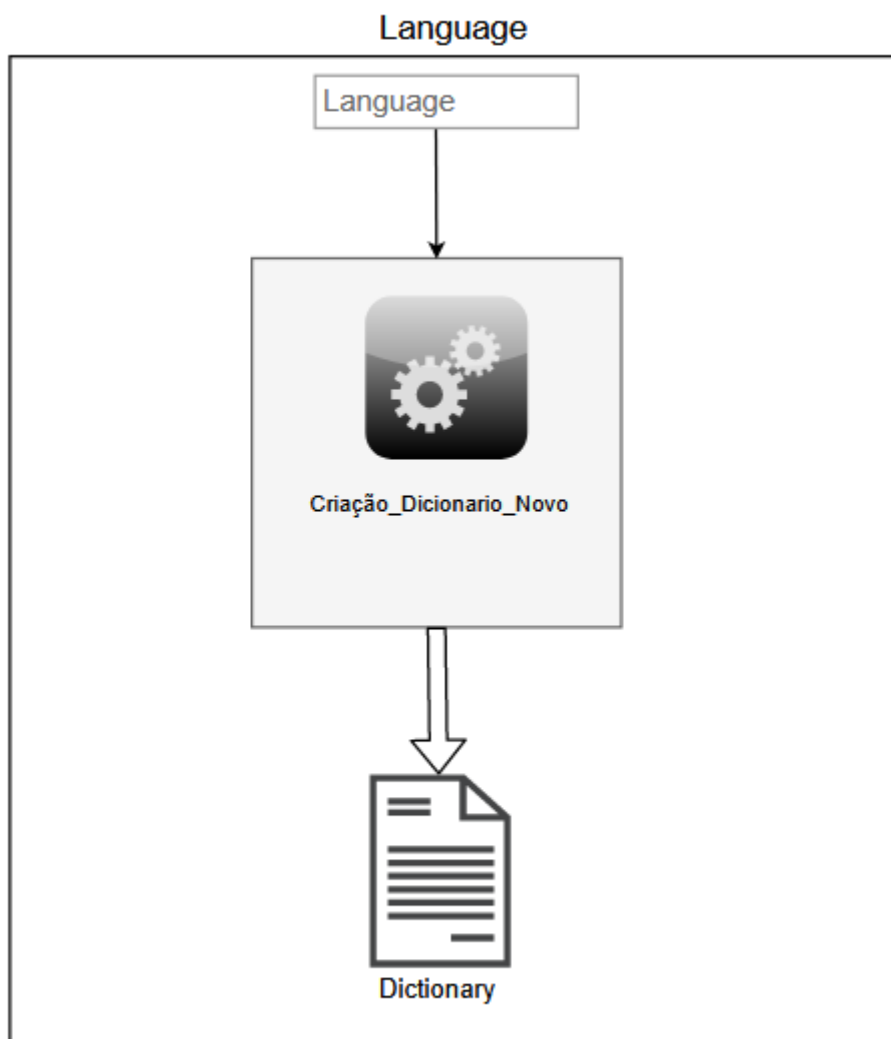


Figura 2.7: Diagrama Novo Idioma

Upload de ficheiros

Neste diagrama 2.8 podemos ver que, para efetuar upload de ficheiros, é necessário indicar o idioma do ficheiro, uma palavra passe e um ou mais ficheiros json que, através do módulo *app*, são guardados na pasta dos ficheiros Json no servidor. Esta ação só é efectuada se a palavra passe introduzida for a correta.

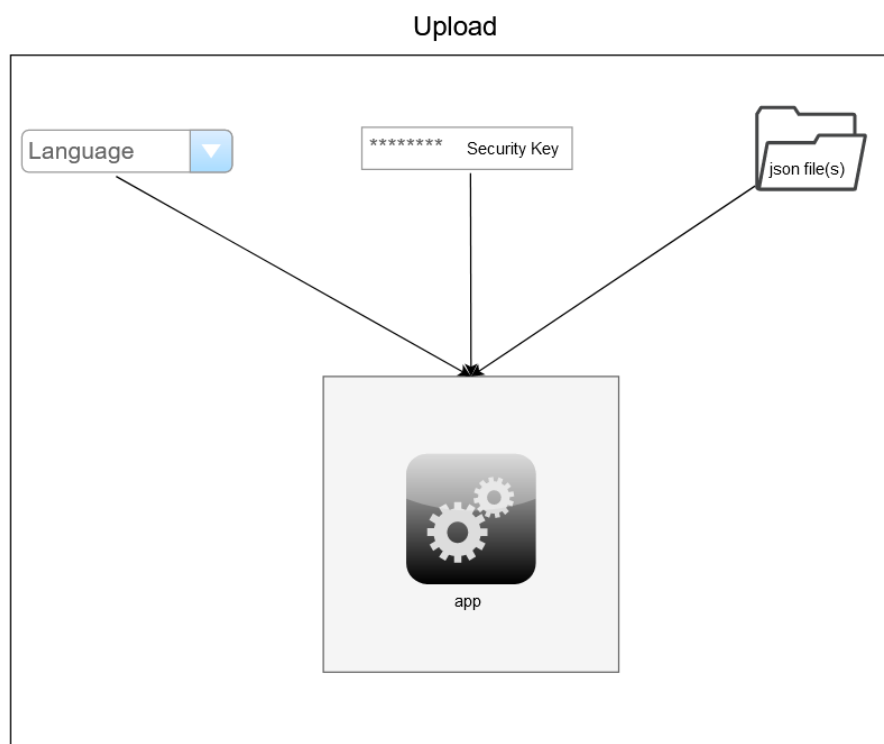


Figura 2.8: Diagrama Upload de ficheiros

Capítulo 3

Concepção da Resolução

3.1 Estruturas de Dados

3.1.1 Dicionários

A peça fundamental do nosso trabalho são os dicionários com as palavras associadas a *hate speech*. Possuímos dois ficheiros *dictionary_English.txt* e *dictionary_Portuguese.txt*, que contêm, respetivamente, um dicionário com as palavras de discurso de ódio inglês e português. As chaves do dicionário são tuplos ("tipo", "variável sociolinguística"). Cada variável sociolinguística possui um hiperónimo e um hipónimo, e tem associada uma lista com as keywords dessa variável.

Apresenta-se em seguida uma entrada do dicionário em Inglês.

```
('Ageism', 'Age - General'): ['Age', 'Ageing', 'Ageism', 'Ageist'],
```

3.2 Implementação

O nosso programa pode ser dividido em dois scripts principais:

O *atualizar_dicionario.py* é responsável por manter o dicionário actualizado, permitindo acrescentar *Keywords* a variáveis sociolinguísticas já existentes e adicionar novos tipos de variáveis. Possui uma única função, *atualizar_dicionario*, que recebe o nome do dicionário que pretende atualizar. As novas palavras devem ser escritas no ficheiro de texto *novas_palavras.txt* com o seguinte formato:

*tipo * variavel sociolinguistica * keyword*

Depois de executar e adicionar as palavras ao dicionário, o ficheiro é limpo.

Além disso, esta função invoca um *script* responsável por gerar o dicionário em formato tabela, recorrendo a *latex*. Na figura 3.1, podemos ver um fragmento da tabela do dicionário inglês.

O "motor" do trabalho é a função *leitura_ficheiro_json* presente no módulo *leitura_ficheiro_json.py* e que se mostra no código 3.1. Esta função recebe o nome do dicionário que se pretende usar para a análise e um ficheiro json. Começa por seleccionar a parte do Json correspondente aos comentários e procura fazer *match* com as palavras do dicionário. Para isso, percorre todas as *keywords* do dicionário e vê se algumas delas está presente no ficheiro.

Fonte de código 3.1: *Leitura_ficheiro_json*

Types of Prejudice	Sociolinguistic variables (Hiper - Hipo)	Keywords
Addiction Shaming	Behavioural Addictions - Alcohol	Drunkie, alcohol
	Behavioural Addictions - Drugs	Junkie, Druggie, Drug addict
	Behavioural Addictions - General	
	Behavioural Addictions - Pornography	
	Behavioural Addictions - Sex	
Ageism	Age - General	Age, Ageing, Ageism, Ageist
	Age - Over 65s	Old, Elder, Elderly, Senior, OAP, Pensioner, Bed-blocker, Wrinklies
	Age - Youngsters	Brat, Half-pint, Insect, Lightweight, Morsel, Nobody, Nonentity, Nothing, Twerp, Whippersnapper, Zero, Zilch
Anti-Clericalism	Religious Identity - Christian	Fundie, Bible thumper, Bible beater, God botherer
	Religious Identity - General	
	Religious Identity - Jewish	Abbie, Abe, Abie, Christ-killer, Ikey, Ike, Iky, Kike, Heeb, Hebe, Hymie, Ikey-mo, Ikeymo, Mocky, moky, moxy, mockey, mockie, mocky, Oven, dodger, Sheeny, Shapeshifter, Shlomo, Shylock, Yid
	Religious Identity - Muslim	Asslifter, Carpet kisser, Goatfucker, Kebab, Koran basher, Koranimal, Mussie, Quran, Koran thumper, Hajji, Haji, Hodgie, Mudslum, Muslintard, Musloid, Muzrat, Pisslamist, Terrorist, Raghead, Towelhead
	Religious Identity - Protestant	Bible basher, Holly roller, Orange, Snout, Prod, Proddy dog, Proddywoddy, Proddywhoddy, Russellite, Spike, Soup taker
	Religious Identity - Roman Catholic	Creeping Jesus, Left-footer, Mackerel Snapper, Mick, Papist, Taig
	Religious Identity - Sikh	Raghead, Towelhead
	Physical Identity - General	Body Shaming
Body Shaming	Physical Identity - Physical (and Mental) Impairments	Ableism, Eugenics, Blind, Deaf, Dumb, Stutterer, Dwarf, Hunchback, Lame, Limp, Midget, Cripple, Disabled, Freak, Handicapped, Wheelchair bound, Bonkers, Crazy, Dim, Feeble-minded, Lunatic, Loony, Mental, Mong, Mongoloid, Nutter, Psycho, Retard, Spazz, Spazzie, Spastic, Weird, Weirdo, Window licker
	Physical Identity - Physical Features	Fat, Fatso, Fatty, Butterball, Chubby, Lardass, Lump, Obese, Porker, Thin, Bag-of-bones, Beanpole, Beanstalk, Bony, Skinny, Skin-and-bones, Scrawny, Scraggy, Twiggy, Double eyes, Double chin, Bald
		Bum, Vagrant, HavenotOutcast, Underdog
Classism and Aporophobia	Social Class - Destitute	Classism, Aporophobia, Social climber, Nouveau riche
	Social Class - General	

Figura 3.1: Dicionário inglês (fragmento)

```
def leitura_ficheiro_json(fileDicionario,ficheiroJson):

    (...)

    for (t,v) in keywordTabEN:
        tp = t
        slv = v

    for j in keywordTabEN[(t,v)]:

        if (' ' + j.lower() + ' ') in comentario.lower():
            tp = t
            slv = v

        quantidadeHate += comentario.lower().count(' ' + j.lower() + ' ')

        if coment not in comentario_Tabela:
            comentario_Tabela[coment] = ([], [], [])

        len(comentario_Tabela[comentario][1])
        comentario_Tabela[coment][0].append(slv)
        i = 0
        while i < coment.lower().count(j.lower()):
            comentario_Tabela[coment][1].append(j)
            idxCo.append((res[i], len(j)))
            i = i + 1
            i2 = coment.lower().count(' ' + j.lower() + ' ')
```

```

print("I", i, i2)

if slv not in variaveisSoc:
    variaveisSoc[slv] = []

if j not in variaveisSoc[slv]:
    variaveisSoc[slv].append(j)

if slv not in ocorrencias:
    ocorrencias[slv] = 1
else:
    ocorrencias[slv] += comentario.lower().count(' ' + j.lower() + ' ')

(...)

```

Quando é encontrada uma *keyword*, guarda-se uma série de informações que serão relevantes para a análise de resultados, nomeadamente o número de vezes que ocorreu e a variável sociolinguística a que pertence.

Esta função é ainda responsável por chamar o script *freq_to_Table*, que gera um documento em pdf com a seguinte estrutura (ver figuras 3.2 e 3.3): uma tabela que contém a análise feita por comentário e outra tabela com a análise por variável sociolinguística.

Comment	KeyWords	Sociolinguistic Variables	Hate Speech Frequency	Hate Speech Frequency(%)
Global world population.65 Asian [EMOJI] growing [EMOJI] .17 Black [EMOJI] growing rapidly [EMOJI] .10 White [EMOJI] in decline [EMOJI] . 8 other.	Black	Ethnicity - Black	1/12	8.333
Sounds like white people should stop being the only race trying to start a war among races	Race	Ethnicity - General	2/17	11.765
If you say it's okay to be a racist , I say it's okay to jail you for domestic terrorism.	Racist	Ethnicity - General	1/19	5.263
Trump supporters suck, they are the Lowest people in the nation .	Nation	Nationality - General	1/11	9.091

Figura 3.2: Tabela que efectua a análise de cada comentário no Post.

Sociolinguistic Variables	KeyWords	Number of occurrences	Frequency	Frequency(%)
Ethnicity - Black	Black	1	1/271	0.37
Ethnicity - General	Race, Racist	2	2/271	0.74
Nationality - General	Nation	1	1/271	0.37

Figura 3.3: Tabela que apresenta uma síntese dos resultados obtidos para cada uma das variáveis sociolinguísticas presentes

No fim do documento é apresentado um resumo detalhado da análise efetuada, como se mostra na figura 3.4. Este resumo contém o número total de palavras ofensivas detetadas, a percentagem de *hate speech* presente no documento, as variáveis sociolinguísticas associadas às keywords encontradas e a indicação da variável predominante.

Este trabalho de análise só é feito uma vez pelo que, se for escolhido para análise um ficheiro já analisado anteriormente, o programa apenas irá buscar o ficheiro em pdf

correspondente e não repete o processo toda outra vez. Desta forma, garantimos maior eficácia ao programa.

- Taking into account the words that were detected, we can reach the conclusion these comments are associated with : : Ethnicity - Black; Ethnicity - General; Nationality - General;
- The percentage of hate speech related words is 1.476.
- Considering that the variable **Ethnicity - General** has the most occurrences in the post, we can interpret that this is the predominant hate speech.
- Overall there were 4/16 occurrences of hate speech related comments.

Figura 3.4: Análise global do ficheiro

Capítulo 4

Codificação, Testes e Resultados

4.1 NetLang Analyzer

Tendo em conta que a nossa ferramenta será utilizada por pessoas com poucos conhecimentos informáticos, achámos conveniente criar uma interface Web que simplificasse o processo de análise de um ficheiro *Json*. Para isso utilizamos a framework *Flask*.

Apresentamos agora algumas das funcionalidades presentes no site. A imagem 4.1 mostra a página inicial do site. Esta é constituída por uma descrição muito breve do projeto e por botões que conduzem às várias funções da ferramenta.

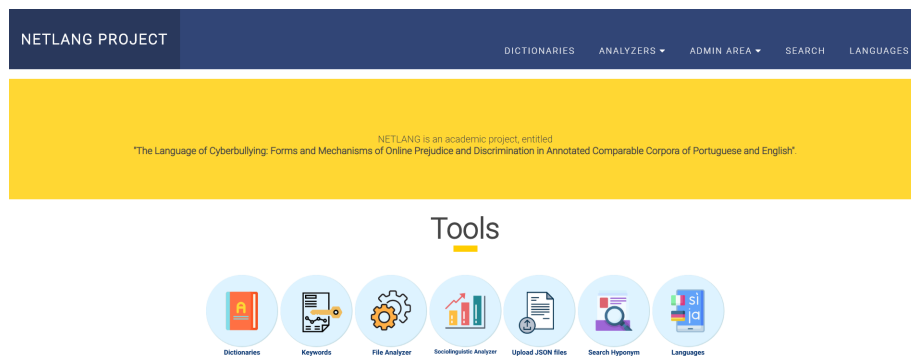


Figura 4.1: Página inicial do site

Na figura 4.2 é possível ver o separador correspondente ao *file analyzer*, que contém os vários campos necessários para a análise do(s) ficheiro(s).

The screenshot shows the 'ANALYZERS' section of the NETLANG PROJECT website. At the top, a dark blue navigation bar contains the project name and links to 'DICTIONARIES', 'ANALYZERS', 'ADMIN AREA', 'SEARCH', and 'LANGUAGES'. Below the navigation bar, the heading 'Here you can analyze your files' is centered. The main content area includes a 'Select a language:' dropdown menu with 'English' selected. Below this is a 'Select one or more files:' section with a text area containing a list of JSON files: 'Youtube_extraction_english_1.json', 'Youtube_extraction_english_10.json', 'Youtube_extraction_english_2.json', 'Youtube_extraction_english_3.json', 'Youtube_extraction_english_4.json', 'Youtube_extraction_english_5.json', 'Youtube_extraction_english_6.json', 'Youtube_extraction_english_7.json', and 'Youtube_extraction_english_8.json'. A 'Submit' button is located at the bottom of the file selection area.

Figura 4.2: Página que permite efectuar a análise de um ou mais ficheiros Json

A figura 4.3 mostra a página que apresenta o ficheiro em pdf do dicionário seleccionado e, por fim, na imagem 4.4 pode-se ver o separador que corresponde à atualização do dicionário. Primeiro é necessário escolher a ação pretendida (introduzir ou remover uma *keyword*) e uma palavra passe. Só no caso de estar correta é que aparecerá o formulário para preencher com as informação da *keyword*.

The screenshot shows the 'ANALYZERS' section of the NETLANG PROJECT website. At the top, a dark blue navigation bar contains the project name and links to 'DICTIONARIES', 'ANALYZERS', 'ADMIN AREA', 'SEARCH', and 'LANGUAGES'. Below the navigation bar, the heading 'On this page, you can consult the keywords of each dictionary.' is centered. The main content area includes a 'Select a language:' dropdown menu with 'English' selected. Below this are two buttons: 'Dictionary' and 'Home'.

Figura 4.3: Página que permite consultar o dicionário numa tabela

The screenshot shows the 'ADMIN AREA' section of the NETLANG PROJECT website. At the top, a dark blue navigation bar contains the project name and links to 'DICTIONARIES', 'ANALYZERS', 'ADMIN AREA', 'SEARCH', and 'LANGUAGES'. Below the navigation bar, the heading 'The area is reserved for administrators.' is centered. The main content area includes a message: 'Please select the action you want to specify and indicate the respective security key.' Below this is a 'Select action:' dropdown menu with 'Add word' selected. Below the dropdown is a 'Security key' input field. A 'Submit' button is located at the bottom of the form. A 'Home' button is also visible at the bottom left.

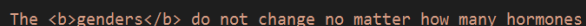
Figura 4.4: Página que permite adicionar ou remover keywords dos dicionários

O *NetLang analyzer* contém ainda uma hiperligação, no separador *search*, para um site desenvolvido pelos coordenadores do projeto. Este site permite filtrar os JSONs que foram classificados de acordo com uma determinada variável sociolinguística.

4.2 Preenchimento do ficheiro JSON

O módulo `leitura_ficheiro_json.py` é responsável por preencher no *header* do ficheiro JSON os campos **keywords** e **socioLingVar** com as keywords e variáveis Sociolinguísticas presentes nos comentários. Cada um desses campos é preenchido com uma string que contém, respetivamente, todas as keywords e variáveis sociolinguísticas. Estas variáveis são apresentadas por ordem decrescente de maior ocorrência. Além disto, as keywords são ordenadas também, tendo em conta a ordem das variáveis sociolinguísticas.

Para além disso, os comentários do ficheiro são alterados e preenchidos com tags HTML "``" e "``" para colocar as keywords encontradas a negrito. Esta alteração irá ser útil na ferramenta de "search" que explicámos anteriormente. A figura 4.5 mostra um exemplo de um comentário modificado. Como já explicámos na secção 3.2, relativamente à análise dos ficheiros JSON, este processo também só é feito uma vez pelo que, se um ficheiro JSON submetido no *file analyzer* já tiver sido analisado, os comentários deste mesmo ficheiro não serão alterados.



```
The <b>genders</b> do not change no matter how many hormones
```

Figura 4.5: Comentário modificado num ficheiro JSON (fragmento)

Capítulo 5

Alternativas, Decisões e Problemas de Implementação

5.1 Decisões

Ao longo deste projecto e à medida que os resultados foram surgindo, deparámo-nos com a necessidade tomar algumas decisões.

A primeira consistiu em escolher um método para **Guardar os Dados**. Tendo em vista o utilizador final da ferramenta, decidimos guardar os dados organizados em dicionários no formato de ficheiro de texto. Os resultados da análise podem ser consultados num ficheiro em formato pdf, gerado pelo servidor, que fica disponível no site.

Verificámos que eventualmente iria ocorrer o **Problema dos falsos positivos**, isto é, algumas palavras poderiam ser detectadas incorrectamente como *hate speech*, quando na realidade se encontram num outro contexto. Verificámos que este problema seria difícil de corrigir, tendo em conta o tempo e as ferramentas de que dispomos. Decidimos então contabilizar todas as ocorrências de uma dada palavra, contando-as como se tratassem efectivamente de um contexto ofensivo. Esta opção foi escolhida por ser a mais simples e facilitar uma melhor implementação futura.

5.2 Problemas de Implementação

Foram surgindo pequenos problemas ao longo da implementação da ferramenta. Apresentamos aqui os que consideramos principais.

5.2.1 Problema do Encoding

O problema do encoding (representado na figura 5.1) ocorreu com muita frequência no servidor após transferirmos ficheiros que estavam a ser manuseados nos nossos computadores (em sistema operativo Windows) para o servidor. Verificou-se que o encoding que estávamos a usar não tinha capacidade para compilar caracteres e letras especiais como, por exemplo, 'ç', entre outros.

Agradecemos neste ponto a ajuda dos coordenadores do projeto e principalmente a ajuda do Engenheiro Aragão que teve um papel determinante na resolução deste problema. Percebemos, com a sua ajuda, que os ficheiros JSON e o ficheiro do dicionário

estavam em ASCII (depois de executar o comando *file nome_ficheiro.json*). Tivemos então de converter os ficheiros para UTF-8, com a ajuda deste comando: *iconv -f latin1 -t UTF-8 < dictionary_English.txt > dictionary_English2.txt*. Com isto, o problema de leitura nos ficheiros ficou resolvido. Em termos de geração do *pdf*, foi necessário utilizar estes *packages*:

```
\usepackage[utf8]{inputenc}
\usepackage{lmodern,textcomp}
\usepackage[T1]{fontenc}
```

Assim ficou garantido que caracteres portugueses eram reconhecidos.

O erro seguinte, que levou algum tempo a ser ultrapassado, foi o facto de caracteres especiais não serem aceites na geração do pdf, com o *pdflatex*. Para resolver o problema foi necessário criar uma função que remove os caracteres especiais, não detetados por UTF-8, e todos os links presentes no comentário. Isto porque contêm associações de caracteres que podem ser interpretados como comandos *latex* e, conseqüentemente, dar erro. Mostramos, como exemplo, a expressão regular que usamos para detetar os links. Quando esta é encontrada, é substituída por "[LINK]".

```
coment = re.sub(r'\w+:\{2\}[\d\w-]+(\.[\d\w-]+)*(?:\s(?:\s|/[\^\\s/]*))*', " [LINK] ", coment)
```

```
gepl@gepl:~/gepl_ds/public_html/NetLang/Projeto_LCC$ python3 leitura_ficheiro_json.py teste_3.json
File "leitura_ficheiro_json.py", line 142,
SyntaxError: Non-UTF-8 code starting with '\xe7' in file leitura_ficheiro_json.py on line 142, but no encoding declared; see http://python.org/dev/peps/pep-0263/ for details
gepl@gepl:~/gepl_ds/public_html/NetLang/Projeto_LCC$
```

Figura 5.1: Erro de Encoding.

5.2.2 Problemas dos Smiles

Apresentamos no código 5.1, parte da função que remove os emojis e caracteres não reconhecidos pelo package de input do *Latex*. Esta função procura pelas expressões regulares que correspondem à codificação dos emojis, como também pelas codificações de caracteres de outros alfabetos, que não são aceites pelo *pdflatex*. As primeiras são substituídas pela string "[EMOJI]" e as últimas por um espaço em branco.

Fonte de código 5.1: Função que remove os smiles

```
def give_emoji_free_text(text):
    result = re.sub('([\U0001F600-\U0001F64F|\U0001F300-\U0001F5FF|
\U0001F68-\U0001F6FF|\U0001F1E0-\U0001F1FF|\U0001F1E6-\U0001F1FF|
\U00002702-\U000027B0|\U00002500-\U00002BEF|\U000024C2-\U0001F251|
\U0001f926-\U0001f937|\U0001F1F2|\U0001F1F4|\U0001F620|\U0001F9E2|
\U0001F900-\U0001F9FF|\U0000200D|\U00002640|\U0000FE0F|
\U0001F3FB])+', " [EMOJI] ", text)
    result = re.sub('([\U00000400-\U000004FF])+', " ", result) #CYRILLIC
    result = re.sub('([\U00000600-\U000006FF])+', " ", result) #ARABIC
    result = re.sub('([\U00000590-\U000005FF])+', " ", result) #HEBREW
    result = re.sub('([\U00000100-\U0000017F])+', " ", result) #LATIN
```



```
result = re.sub('[(\U00000180-\U0000024F)]+', " ", result) #LATIN extended-B
result = re.sub('[(\U00000370-\U000003FF)]+', " ", result) #LATIN extended-addict
```

A meio do desenvolvimento do projeto, surgiu também o problema do mau reconhecimento do carácter `"/`, cujo código em HTML é `%2F`. Esta questão foi resolvida usando a propriedade `href`, na tag `<a>` do HTML, em vez de `url_for...` .

5.2.3 Problema das percentagens.

Ao gerar o *pdf* e efetuar a sua análise, verificou-se um problema no cálculo das percentagens de *Hate speech* presentes nos comentários analisados. Este problema aconteceu de maneira grosseira, devido maioritariamente a faltas de atenção. O contador das ocorrências estava a ser mal incrementado e em alguns casos substituído por 1 em vez de incrementar. Este problema foi facilmente resolvido após algum tempo de *debugging*.

5.2.4 Problema no Match

Inicialmente o nosso programa apresentava falhas ao filtrar o texto. Quando uma palavra do nosso dicionário estava incluída numa outra maior, por exemplo a palavra `"ho"` em `"hot"` ou `"ho"` em `"horse"`, o programa detetava a existência de *hate speech* no comentário, destacando a palavra `horse` como ofensiva. Como resultado, esta era acrescentada ao número de ocorrências de palavras ofensivas nos comentários e, por sua vez, no ficheiro *pdf* com os resultados. Como estas palavras não estavam realmente presentes, o resultado ficava detorpedo. Este problema foi resolvido com a criação de uma função que limpa do comentário os caracteres indesejados e acrescenta um espaço no início e no fim de cada palavra. A função *clean_str* do módulo *leitura_ficheiro_json.py*, apresentada no código 5.2, é responsável por introduzir essas alterações.

Fonte de código 5.2: Função que limpa uma string

```
def clean_str(string):
    caracteres_indesejados = [',', '!', '?', '.', '"', '@', '+', '^', '(', ')']

    for c in caracteres_indesejados:
        string = string.replace(c, ' ')    #substituir o carater por um espaço

    string = " ".join(string.split())

    string = ' ' + string + ' '
    return string
```

Depois, ao fazer a filtragem dos texto, basta verificar se cada palavra do dicionário se encontra no comentário.

5.2.5 Problema de falsos positivos

Este problema foi detectado, mas não foi ainda resolvido apesar de considerarmos que, futuramente, seria interessante melhorar a ferramenta introduzindo estas alterações. A explicação deste problema é muito simples: como saber se, por exemplo, um título que contém a palavra "vaca" está a insultar alguém ou se está a reportar uma notícia sobre o animal. Esta questão não é exequível, na nossa opinião, sem usar alguma estratégia mais avançada, com recurso a redes neuronais e inteligência artificial. Como esta área não se encontra no âmbito do trabalho e ocuparia demasiado tempo, não seria de todo viável conseguir terminar no prazo estipulado pelo que decidimos não o fazer. No entanto, deixamos a nota de que seria uma adaptação interessante a introduzir na ferramenta.

Outra forma de encarar este problema seria criar as ocorrências possíveis de uma palavra para calcular uma margem de erro. Para isso seria necessário um dicionário com palavras como "vaca" e retirar estas do dicionário principal, para no final apresentar a percentagem +/- margem de erro. Poderia pensar-se em utilizar uma estratégia de validação manual, perguntando ao utilizador se concorda com a decisão tomada. Caso verificasse que, pelo contexto, não se tratava de discurso ofensivo, procederia às alterações. Tendo em conta a dimensão dos comentários não considerámos ser uma opção viável.

Capítulo 6

Conclusão

Este relatório descreveu o problema, fez uma análise da conceção da resolução e forneceu uma explicação de todo o trabalho desenvolvido. Foram também detalhados os problemas que surgiram ao longo da implementação. Estes, de alguma forma, poderão ser uma ajuda a quem for confrontado com o mesmo tipo de erros no futuro.

O projeto foi para nós um trabalho foi desafiante. Apesar das dificuldades que foram ocorrendo, foi completado com sucesso. Não só foi criada a ferramenta pedida, como também se desenvolveu um site de modo a simplificar o uso da ferramenta e obtenção de resultados. O trabalho foi feito de modo a permitir ser alastrado a mais línguas e a ferramenta foi implementada de modo tão geral que pode ser usada para outros fins para além de deteção de discurso socialmente inaceitável. Tendo o dicionário que se pretende usar, este analisador pode ser usado, por exemplo, para encontrar certas palavras para propósitos comerciais.

Decidimos apresentar a documentação do código em inglês para, futuramente, ser mais fácil de aumentar a sua capacidade e exactidão para, por exemplo, usar redes neurais para apanhar falsos positivos e falsos negativos.

O facto de termos começado o trabalho com bastante antecedência, fez com que pudéssemos ir aperfeiçoando pequenas falhas detectadas, de forma a obter o resultado final pretendido.

A nível de trabalho futuro, vemos que o aperfeiçoamento da deteção de *hate speech*, à custa da inteligência artificial, poderá melhorar significativamente o poder de análise desta ferramenta. Desta forma poderá ser resolvido, pelo menos em parte, o problema dos falsos positivos que foi referido no documento.

Apêndice A

Reuniões e trabalho desenvolvido

Apresentamos, em seguida, um índice que apresenta algumas das fases pelas quais passámos ao longo do projecto bem como as datas em que se obtiveram resultados nas várias etapas.

10/3 - A função *actualiza_dicionário* também atualiza a tabela com as keywords existentes até ao momento. Feito com o módulo *dict_to_table*.

18/3- Reunião. Foi sugerido que se efetuasse o tratamento dos dados obtidos. Fazendo o cálculo de percentagens relativas à frequência de cada palavra no post global e em cada comentário do post.

22/3- Apresentação das frequências das palavras encontradas nos ficheiros json.

22/3- Correção de erros detectados na leitura dos ficheiros json (estava a ser detectado "ho" em hot por exemplo, ou seja, era detetada uma palavra que não existia no comentário). Foi criada uma nova versão que primeiro limpa o comentário, substituindo os caracteres que não interessam por espaços, e depois deteta se *espaço + keyword + espaço* está na string do comentário.

26/03 - O ficheiro *leitura_ficheir_json.py* gera duas tabelas. A primeira apresenta para cada comentário as keywords que lá se encontram e as variáveis sociolinguísticas a que pertencem e analisa a sua frequência. A segunda tabela faz uma síntese das variáveis presentes no post e as respectivas frequências.

28/03 - O código foi todo comentado, para facilitar uma futura compreensão do mesmo. Também foram corrigidas as frequências das keywords, que estão a ser apresentadas com 3 casas decimais.

28/03 - Nova tabela: comentário, keyword, nº ocorrência, frequência, percentagem.

28-03 - Melhorar a tabela das frequências por variável.

02/04 - Reorganização das tabelas.

02/04 - Adicionamos uma contagem de quantos comentários contêm hate speech no resumo final.

02/04 - Corrigir as percentagens de modo a apresentar os valores correctos.

02/04 - Adicionar % no resumo.

05/04 - Início do desenvolvimento do site.

07/04 - Meter a hiperligação no ficheiros pdf para a ultima tabela; analisar a questão do encoding nas tabelas das frequências.

07/04 - Traduzir para inglês.

07/04 - Mudar o texto para ficar menos denso.

10/04 - O site permite adicionar novas palavras ao dicionário

25/04 - O site já permite fazer upload

6/05 - Corrigidos erros no site, permite agora fazer upload de vários ficheiros e de varias palavras.

7/05 - Tratar erros de encoding no site.

10/05 - Desenvolvimento do relatório.

12/05 - Site funcional no servidor.

16/05 - Site constantemente ligado.

17/05 - Pedir dicionário no site.

20/05 - Testar ficheiros em português.

20/05 - Modificado o resumo que é feito no upload de vários ficheiros.

20/05 - Colocar a negrito a keyword identificada.

22/05 - Flowchart versão 1.

25/05 - Preenchimento do campo "keywords" dos ficheiros json.

28/05 - Flowchart versão 2.

29/05 - Adicionar marca emoji/link.

03/06 - Adicionar ao site a análise de variáveis sociolinguísticas.

04/06 - Preenchimento do campo "socioLingVar" dos ficheiros json.

06/06 - Adicionar o nome e título do ficheiro ao pdf resultante.

10/06 - Flowchart versão 3.

16/06 - Analisar a partir de uma variável sociolinguística.

17/06 - Flowchart versão 4 e Diagramas.

17/06 - Corrigir a escolha do dicionário PT/EN.

19/06 - Apresentar o dicionário correto no site.

22/06 - Limpeza estética do words/file-analyzer no site.

25/06 - Fazer "drill-down" dos comentários.

25/06 - Retirar palavras erradas colocadas no dicionário.

25/06 - Não permitir adicionar tipos e variáveis sociolinguísticas pelo site.

25/06 - Ao adicionar e retirar uma palavra pedir uma palavra passe.

25/06 - Modificar nomes para aumentar a coerência.

26/06 - Limpar dicionários.

26/06 - Adicionar a possibilidade de analisar por novas línguas.

28/06 - Criação do módulo clean_string.

30/06 - Meter no site a funcionalidade de acrescentar novos idiomas.

01/07 - Arranjar o problema das passwords deixarem entrar quando é dada a password errada

02/07 - Meter o flowchart no site

02/07 - Acrescentar a tab search no site.

03/07 - Deixar fazer upload para o servidor de ficheiros Json externos.

03/07 - Correção de comportamentos irregulares na análise de ficheiros.

03/07 - Quando se faz uma análise apresentar o json modificado ao utilizador.

04/07 - Capitalizar idiomas no site para fins estéticos.

04/07 - Flowchart e Diagramas versão 4

05/07 - Modificação do tipo e variáveis sociolinguística para inglês para uniformizar o projecto.

06/07 - Correção na deteção errática de palavras na análise.

07/08 - Modificação dos dicionários, pdfs e site de modo a dividir a variável sociolinguística pelo para hiperónimo e hipónimo.

08/07 - Criação do apêndice C e pdf com a mesma informação em inglês para posterior publicação no site.

08/07 - Resolvido o problema na criação de novos idiomas e de adicionar palavras a estes.

09/07 - Mudanças estéticas no site.

10/07 - Correção dum problema com passwords.

11/07 - Atualizar a contagem de ocorrências no `leitura_ficheiro.Json.var`.

12/07 - Corrigido algumas coisas com os dicionários.

12/07 - Atualizado o link do search.

13/07 - Guião para apresentação Versão 1.

14/07 - Diagramas e explicação para o site.

22/07 - Modificar o JSON, adicionando tags HTML às keywords encontradas no ficheiro.

Apêndice B

Explicação do Json

O nosso ficheiro json está dividido em duas partes. Uma primeira parte (header) que contém os seguintes campos:

- title: correspondente ao título da notícia/vídeo para ser avaliado.
- subtitle: correspondente ao subtítulo da notícia/vídeo para ser avaliado.
- owner: correspondente a quem publicou o que queremos avaliar.
- views: corresponde ao número de visualizações.
- likes: corresponde ao número de gostos.
- dislikes: corresponde ao número de não gostos.
- shares: corresponde ao número de partilhas.
- datePosted: corresponde à data que foi publicado o post.
- dateExtraction: corresponde à data em que foi extraído o json.
- language: corresponde ao idioma principal do que está a ser avaliado.
- plataform: corresponde à plataforma donde foi extraído o json.
- url: corresponde ao link onde posteriormente se poderá aceder ao que estamos a extrair.
- postText: corresponde ao texto que poderá estar escrito por exemplo na descrição de um vídeo, ou de um *abstract* de uma notícia.
- numberPosts: corresponde ao número de comentários.
- srcType: corresponde ao tipo do que estamos avaliar por exemplo video.
- nameNewspaper: corresponde ao nome do jornal de onde está a ser retirada a notícia.
- socioLingVar: campo a ser preenchido pela ferramenta.
- listEvents: campo criado para colocar o tipo de evento que gerou aquela notícia ou vídeo.

- articleKeywords: campo a ser preenchido pela ferramenta.
- keywords: campo a ser preenchido pela ferramenta.
- commentsOpen: corresponde a "yes" se existe a possibilidade de comentar e "no" caso contrário.

Quando algum destes campo não pode ser preenchido deverá ser preenchido ou com NA ou com " ".

Estes metadados servem para dar informações sobre o que vai ser avaliado e irá ser preenchido em alguns campos conforme a avaliação feita. A segunda parte (com-

```

1 {
2   "header": {
3     "title": " Sexism is Worse Than I Realized ",
4     "subtitle": " NA ",
5     "owner": " The Young Turks ",
6     "views": " 96,079 views ",
7     "likes": " 4.2K ",
8     "dislikes": " 790 ",
9     "shares": " NA ",
10    "datePosted": " Streamed live on Sep 14, 2019 ",
11    "dateExtraction": " 2019-10-03 ",
12    "language": " en ",
13    "plataform": " YouTube ",
14    "url": " https://www.youtube.com/watch?v=5yhYHM43yu4 ",
15    "postText": " Cenk Uygur checks in live from the road to discuss sexism and it's possible
16    "numberPosts": " 3,521 Comments ",
17    "srcType": " video ",
18    "nameNewspaper": " NA ",
19    "socioLingVar": " Ethnicity,Gender,Physical (and Mental) Impairments,Black, Ideological and
20    "listEvents": " ",
21    "articleKeywords": " NA ",
22    "keywords": " Racism,Racist,Race,Ethnicity,Ethnic,Sexism,Woman,Gender,Misogynist,Sexual,Cha
23    "commentsOpen": " yes "
24  },

```

Figura B.1: Header do ficheiro Json

mentThread) é onde teremos toda a informação sobre os comentários relativos ao vídeo/notícia que estamos a extrair. Esta parte é composta por um conjunto de módulos que contém os seguintes campos para cada comentário:

- id: correspondente a um identificador do comentário.
- user: correspondente ao nome do utilizador que fez o comentário.
- date: correspondente á data onde foi feito o comentário.
- timestamp: correspondente a um código que é gerado automaticamente pelas plataformas de extração, tem a ver com a data e hora
- commentText: correspondente ao comentário feito.
- likes: correspondente ao número de likes que o comentário recebeu.
- hasReplies: correspondente a uma flag booleana que indica se há ou não respostas a este comentário.
- numberOfReplies: Correspondente ao número de respostas que o comentário recebeu.


```

25 "commentThread": [
26   {
27     "id": "Ugx6tEuY4exOzL8cniV4AaABAg",
28     "user": "I agree, but",
29     "date": "1 day ago",
30     "timestamp": 1570022644562,
31     "commentText": "I came here from r/comedyheaven",
32     "likes": 2,
33     "hasReplies": false,
34     "numberOfReplies": 0
35   },
36   {
37     "id": "UgzEFcqwwVv193qUVFZ4AaABAg",
38     "user": "Influx27",
39     "date": "5 days ago",
40     "timestamp": 1569677044564,
41     "commentText": "What's going on with Cenk's forehead?",
42     "likes": 0,
43     "hasReplies": false,
44     "numberOfReplies": 0
45   },
46   {
47     "id": "UgxUQgMfT_Z7oCZtu194AaABAg",
48     "user": "\u00c4u\u00dferst Rot",

```

Figura B.2: Comment thread do ficheiro Json (fragmento)

Apêndice C

Compilar os módulos pelo servidor

O nosso programa encontra-se dividido em módulos (ficheiros que terminam em .py). Dentro de cada módulo estão definidas algumas funções que são invocadas no módulo principal da nossa aplicação, app.py.

Cada um destes módulos pode ser corrido directamente no interpretador de python. Podemos importar as definições de um módulo directamente para a tabela de símbolos do módulo importador.

Apresentamos em seguida os comandos genéricos para correr os principais *scripts* directamente pelo servidor.

No interpretador de python:

- leitura_ficheiro_json.py:

```
from leitura_ficheiro_json import leitura_ficheiro_json

leitura_ficheiro_json(<dicionario>.txt,<ficheiro Json>)
```

- atualizar_dicionario.py:

```
from atualizar_dicionario import atualizar_dicionario

atualizar_dicionario(<dicionário>.txt)
```

- limpar_dicionario.py:

```
from limpar_dicionario import limpar_dicionario

limpar_dicionario(<dicionário>.txt)
```

- leitura_ficheiro_json_variavel soc.py:

```
from leitura_ficheiro_json_variavel soc
import leitura_ficheiro_json_variavel soc

leitura_ficheiro_json_variavel soc (<dicionario>.txt,
<ficheiro Json>, <variavelSociolinguistica>)
```

- **criarDicionario.py:**

```
from criarDicionario import criarDicionario

criarDicionario (<nova_linguagem>)
```

- **dict_to_table.py:**

```
from dict_to_table import dict_to_table

dict_to_table (<dicionário>.txt)
```

Sempre que se efetuam alterações nos módulos, para que as mesmas se tornem visíveis na aplicação web, é necessário efectuar os seguintes comandos no terminal do servidor:

```
ps fax | grep -i flask
```

```
kill -9 PID_DO_PROCESSO_OBTIDO_NO_COMANDO_ANTERIOR
```

```
export FLASK_RUN_PORT=10400
```

```
nohup flask run --host 127.0.0.1 &
```