Universidade do Minho
Escola de Engenharia

Processamento de Linguagem Natural em Engenharia Biomédica
2024-2025

Trabalho Prático de Grupo

Trabalho Prático 1

Beatriz Amorim, PG56112
Carolina Santos, PG56116
Catarina Nunes, PG56117

Mestrado em Engenharia Biomédica | Informática Médica

Conteúdo

1 Introdução

A informação terminológica presente em glossáriros e dicionários especializados desempenha um papel fundamental na comunicação científica e técnica, especialmente na área da saúde, onde a precisão dos conceitos é crucial. No entanto, muitos deste recursos são disponibilizados em formatos pouco estruturados, o que dificulta a sua reutilização em aplicações como sistemas de apoio à decisão clínica ou ferramentas de tradução automática.

Assim, este trabalho tem como objetivo aplicar técnicas de Processamento de Linguagem Natural para extrair e estruturar informação terminológica a partir de documentos em formato PDF relacionados com a área da saúde. Foram utilizados diferentes glossários, nomeadamente, o *Glossário de Neologismos da Saúde Humana*, o *Dicionário Multilingue da COVID-19*, o *Glossário de Termos Médicos Técnicos e Populares* e o *Glossário Temático de Monitoramento e Avaliação*.

Com recurso a ferramentas de extração de texto e à aplicação de expressões regulares em *Python*, foi desenvolvido um processo que converte os conteúdos textuais dos documentos referidos em estruturas organizadas no formato JSON. Esta estruturação permite representar cada entrada terminológica com as suas respetivas propriedades, facilitando a análise e a integração em sistemas informáticos.

Nas secções seguintes, é descrita em detalhe a metodologia adotada para processar cada documento e, por fim, é apresentada a estrutura final que reúne todos os resultados obtidos.

2 Análise dos documentos

- **Dicionário Multilingue da COVID-19**: O foco deste dicionário está na tradução dos termos entre línguas como occitano (oc), basco (eu), galego (gl), espanhol castelhano (es), inglês (en), francês (fr), português de Portugal (pt [PT]), português do Brasil (pt [BR]), holandês (nl) e árabe (ar). Existem ainda anotações morfológicas que indicam características gramaticais das palavras, como o género, número e classe lexical. Estas anotações permitem não só compreender melhor a utilização dos termos nas diferentes línguas, como também contribui para sistemas de Processamento de Linguagem Natural, como a lematização ou a tradução automática. Adicionalmente, contém também informação sobre as siglas, os símbolos e as possíveis denominações comerciais ou científicas que o termo pode tomar e a respetiva definição do mesmo.

- **Glossário de neologismos da Saúde Humana**: Este glossário, apesar de conter traduções em inglês e espanhol, tem como objetivo principal contextualizar termos técnicos, com descrições, citações e, até, informação enciclopédica. Ou seja, o foco está no conteúdo e significado dos termos no contexto da saúde.

- **Glossário de Termos Médicos Técnicos e Populares**: Este glossário apresenta uma lista de termos médicos técnicos acompanhados pelas suas variantes populares em português de Portugal.

- **Glossário Temático - Monitoramento e Avaliação**: Apresenta uma lista de termos técnicos e conceitos especializados que são utilizados no contexto de gestão de saúde pública, monitorização de políticas públicas, avaliação de programas, análise de dados e informações estratégicas.

3 Metodologia

3.1 Dicionário Multilingue da COVID-19

Após a análise da estrutura do documento, constatou-se que, para um único termo, existia um total de 13 campos que poderiam ser extraídos. O primeiro passo do processamento deste ficheiro foi, então, a definição das estruturas de dados para as quais a informação deveria ser extraída. Devido à ampla gama de línguas para o qual este dicionário oferece traduções, e tendo em conta a existência de casos de múltiplas traduções para a mesma língua, decidiu-se organizar todos estes elementos sobre a forma de um dicionário, onde as chaves correspondem às diferentes línguas e os valores à lista de possíveis traduções para determinada língua. Para campos como o das siglas, as notas ou dos sinónimos complementares, que podem apresentar mais do que um elemento, adotou-se o armazenamento em lista, enquanto que para campos de texto corrido, como o campo da categoria ou da descrição, a informação foi simplesmente armazenada numa *string*.

Em seguida, segue-se um exemplo da estrutura geral adotada para cada termo.

```
{
"Conceito": {
"Categoria Lexical": "",
"Traduções": { 'oc': [], 'eu':[], 'gl': [], 'es': [], 'en':
 ↪  [], 'fr': [], 'pt': [], 'pt_PT': [], 'pt_BR': [], 'nl':
 ↪  [], 'ar': []},
"Categoria": "",
"Descrição": "",
"Notas": [],
"Sigla": [],
"Símbolo": "",
"Entrada Principal": "",
"Sinónimo": "",
"Sinónimo Complementar": [],
"Denominação Comercial": "",
"Nome Científico": "",
"Número CAS": []
}
```

Como nem todos os campos estão presentes em todos os termos, optou-se por se inicializar o dicionário vazio, de maneira a que a cada termo apenas correspondessem campos que efetivamente existem e evitando o armazenamento de informação vazia.

Um dos maiores desafios da extração de informação deste documento prendeu-se na divisão de cada página em duas colunas de dados. De maneira a obter um ficheiro com toda a informação ordenada, procedeu-se, inicialmente, à conversão do ficheiro pdf para um ficheiro xml, através do comando *pdftohtml -xml*. Neste ficheiro xml inicial, a informação encontrava-se desorganizada, devido a ter sido extraída simultaneamente de duas colunas, pelo que os valores fornecidos pelos campos *left* do deste ficheiro foram essenciais à separação e ordenação do conteúdo das duas colunas. Assim, este ficheiro xml inicial foi percorrido página a página e, tendo em conta um valor limite definido para o parâmetro *left*, cada linha foi adicionada à respetiva coluna (direita ou esquerda). No final de cada página, as duas colunas foram concatenadas, dando origem a um ficheiro completo, organizado e mais fácil de trabalhar.

De seguida, para facilitar a sua manipulação, todo o conteúdo do ficheiro que não correspondia à secção do dicionário foi removida manualmente. Procedeu-se também a uma limpeza geral do documento, onde foram removidas as quebras e números de página, os cabeçalhos e as linhas vazias, de maneira a obter um documento apenas com informação útil. Para tornar o documento mais legível, cada entrada do dicionário foi marcada com um caracter facilmente distinguível (@), de modo a facilitar a sua futura extração. Por fim, removeram-se os campos *text*, *top*, *left* e *width* de todas as linhas do ficheiro, deixando apenas os campos *height* e *font* para auxiliar na distinção entre os diferentes atributos.

Passando à extração da informação propriamente dita, foi feito o *split* do documento anteriormente obtido através do caracter "@", obtendo-se uma lista com todas as entradas do dicionário. Esta lista de entradas foi, então, percorrida sequencialmente.

Em cada entrada, o primeiro passo foi sempre extrair o respetivo conceito, sendo que este corresponde à primeira linha em negrito, através da expressão regular seguinte.

```
concept_match = re.search(
            r'<height="\d+"
             ↪  font="\d+"><b>\s*(.*)\s*</b>(\n<.*><b>(.*)</b>)?',
            entry
        )
```

Tanto neste caso como nos restantes atributos, devido à estrutura do ficheiro xml gerado e à sua tendência para quebrar linhas de texto, foi sempre necessário ter em atenção se o conteúdo pretendido não se estendia também para a linha seguinte, como pode ser verificado na expressão regular fornecida.

Nesta procura pelos conceitos do dicionário, foi encontrada uma exceção: o termo *èczema* não se encontra a negrito. Tendo sido o único caso detetado, procedeu-se à sua correção manualmente no ficheiro txt, de maneira a que fosse capturado corretamente pelas expressões regulares utilizadas.

Os atributos de cada termo foram extraídos de modo semelhante para a estrutura anteriormente referida, recorrendo sempre a expressões regulares e tendo em atenção a sua extensão por diversas linhas distintas. Foram ainda encontradas duas outras exceções na formatação do documento, no termo *taxa de mortalitat*, onde o sinónimo complementar, identificado por *sin. compl.* não se encontra a negrito, e no termo *AMY-101*, onde a variante *BR* da língua *pt* se encontrava na linha errada. Mais uma vez, tendo estes sido os únicos casos encontrados para cada uma das situações, procedeu-se à sua correção manual no ficheiro xml.

O resultado final foi um dicionário com um total de 739 entradas, que contrastam com as 743 entradas presentes no ficheiro pdf inicialmente fornecido. Esta diferença deve-se ao facto de o dicionário do ficheiro pdf apresentar quatro entradas com termos repetidos, sendo elas *mascareta*, *immunològic -a*, *immune*, *careta*. Estas quatro entradas encontram-se duplicadas no ficheiro original, o que não é permitido na estrutura de um dicionário, pelo que foram tomadas duas abordagens: nos casos em que as duas entradas apresentavam exatamente a mesma informação, a segunda entrada foi ignorada; nos casos em que o termo era o mesmo, mas as informações compreendidas eram distintas, estas foram concatenadas, de modo a evitar a perda de informação.

3.2 Glossário de neologismos da Saúde Humana

O documento *glossario_neologismos_saude.pdf* apresenta um conjunto de neologismos utilizados na área da saúde, acompanhados de descrições, traduções e outras informações

complementares.
Primeiramente, foi efetuada a conversão do PDF para texto utilizando o comando *pdftotext*. Após a conversão, o texto foi limpo para remover secções não relacionadas com o glossário, como cabeçalhos, notas editoriais ou explicações metodológicas.

Foi observado que cada nova entrada começa com um termo seguido pela categoria lexical (s.f ou s.m). A separação de cada entrada no glossário foi feita com base no seguinte padrão:

```
re.split(r'\n(?=\S.*\s+s\.(?:f|m)\.)', content.strip())
```

Esta expressão divide o conteúdo sempre que encontra uma nova linha seguida de um termo que termina com uma classe lexical s.f ou s.m.

A partir de cada bloco bruto (uma entrada do glossário), foram extraídos os seguintes campos:

- Termo: obtido a partir da linha inicial, retirando a parte da categoria lexical;

```
termo = re.sub(r"\s(s\.f\.|s\.m\.)$", "",
↪  termo_entrada).strip()
```

- Categoria lexical: extraída diretamente com base na mesma linha;

```
categoria_lexical = re.search(r"\s(s\.f\.|s\.m\.)$",
↪  termo_entrada).group(1)
```

- Traduções: foram detetadas se a linha contivesse marcadores [ing] e [esp], com posterior extração dos valores com:

```
trad_match =
↪  re.match(r"^(.*?)\s*\[ing\];\s*(.*?)\s*\[esp\]$",
↪  traducoes_texto)
```

- Sigla: detetada caso o bloco contivesse a linha com o marcador 'Sigla: '

```
sigla_match = re.search(r"Sigla:\s*(.*)", resto_texto)
```

- Informação Enciclopédica: extraída a partir do marcador 'Inf. encicl.: ' até ao início de uma citação (detetada por aspas “);

```
inf_encicl_match = re.search(r"Inf\.
↪  encicl\.:([\s\S]+?)(?=“)", resto_texto)
```

- Citações: foram extraídas todas as expressões dentro de aspas que tinham uma referência bibliográfica a seguir;

```
citacoes = re.findall(r'"([^"]+)"\s*\(([\d,]+)\)',
 ↪  resto_texto)
```

- Descrição: foram consideradas todas as linhas consecutivas entre a categoria lexical e os campos "Sigla", "Inf. encicl." ou as citações. O código detetava a quebra nesse padrão e armazenava as linhas como descrição.

No final obteve-se um dicionário com 306 entradas cada entrada com a seguinte estrutura:

```
{
    conceito : {
        "categoria_lexical": "",
        "traducoes": {
          "en": "",
          "es": ""
        },
        "descricao": "",
        "sigla": "",
        "inf_encicl": "",
        "citacoes": [""]
      }, ...
}
```

3.3 Glossário de Termos Médicos Técnicos e Populares

Este glossário foi inicialmente convertido para XML, uma vez que, o documento original apresentava marcações visuais como o negrito e itálico, usadas para distinguir os termos médicos e os termos populares. A conversão para XML permitiu, assim, numa primeira fase, preservar estas marcações de forma estruturada.

Posteriormente, foi aplicado um processamento ao ficheiro XML, com o objetivo de simplificar o conteúdo apresentado e ser, assim, mais facilmente convertido para o formato JSON. Durante este processo, foram removidas todas as *tags* e elementos desnecessários, como os cabeçalhos, divisórias de alfabeto, formatação em itálico (uma vez que os conceitos populares se apresentavam todos desta forma, não havia relevância em manter essas *tags*) e marcações de *layout* como <text> e <page>.

Além disto, foi realizada uma limpeza final para eliminar linhas vazias e espaços a mais, mantendo no ficheiro apenas a informação essencial, como os termos destacados pelas *tags* de negrito, e os termos populares em texto normal. O ficheiro processado foi convertido então para *.txt*.

Após esta limpeza e preparação do ficheiro em formato de texto, foi realizado o processamento necessário para extrair os termos técnicos e populares, utilizando expressões regulares. O padrão de procura, foi desenhado para capturar tanto o termo técnico (a negrito), como o termo popular (com a indicação "(pop)"). A expressão em *Regex* utilizada foi a seguinte:

```
pattern = re.compile(
    r'(?P<popular>(?:[^<(]|\((?!(?:pop\)))))*?)\s*\(pop\)\s*,\s
       *<b>(P<tecnico>(?:.|\n)+?)</b>|'
    r'<b>(?P<tecnico2>(?:.|\n)+?)</b>\s*,\s*(?P<popular2
       >(?:[^<(]|\((?!(?:pop\)))))*?)\s*\(pop\)',
    re.DOTALL)
```

Esta expressão regular procura encontrar dois padrões principais que foram detetados na fase da análise do documento:

1. O termo popular seguido de "(pop)", e o termo técnico destacado a negrito dentro das tags <b>.
2. Alternativamente, o termo técnico primeiro, seguido do termo popular.

Foi então criado o dicionário, que armazena os pares de termos encontrados - a chave é o termo técnico, e o valor uma lista de termos populares. Para garantir que os dados ficassem corretamente formatados, também se removeu quebras de linha e excluiu-se vírgulas e espaços extra. A expressão utilizada para substituir múltiplos espaços consecutivos e normalizar as quebras de linha foi:

```
popular = re.sub(r'\s+', '', popular)
popular = popular.replace('\n', '')
popular = popular.strip(',')
```

No caso de múltiplas entradas para o mesmo termo técnico, foram removidas duplicações, garantindo que os termos populares associados fossem exclusivos.

Foi necessário criar uma condição especial para tratar um erro ortográfico específico, no termo "pré-medicação", garantindo que o termo associado estivesse corretamente formatado, e não fosse criada duas entradas na lista de termos populares, quando a descrição era idêntica.

```
if tecnico.lower() == "pre-medicacao":
    popular_terms = [term for term in popular_terms if term.
       endswith('geral')]
```

Por fim, o conteúdo foi ordenado alfabeticamente e exportado para um ficheiro JSON.

3.4 Glossário Temático - Monitoramento e Avaliação

O processamento deste glossário teve início, mais uma vez, com a conversão do ficheiro original, em formato *PDF*, para um formato de texto simples *.txt*, de forma a facilitar a sua manipulação e limpeza automática.

Após a conversão, foi necessária uma etapa de limpeza do ficheiro de texto, que teve como objetivo remover os elementos não necessários, como números de página, cabeçalhos, rodapés, espaços em excesso e quebras de linha desnecessárias. Além da remoção destes

elementos, foram também aplicados marcadores aos conceitos identificados, com o intuito de separar cada entrada no glossário, preparando o conteúdo para fases posteriores de extração e organização automática dos dados.

Após esta fase de limpeza, tornou-se necessário estruturar o conteúdo de forma que cada entrada do glossário pudesse ser identificada com clareza. Para isso, foi utilizado um padrão de expressão regular que reconhece as linhas correspondentes aos conceitos, distinguindo-as das suas respetivas descrições. Este padrão procura por linhas que sigam o formato típico de uma entrada de glossário, isto é, um termo seguido de uma indicação de género gramatical (por exemplo, *fem.* ou *masc.*). Sempre que uma linha cumpre este critério, é-lhe atribuído um marcador personalizado (`###`) para assinalar o início de um novo conceito.

Este marcador facilita não só a leitura do ficheiro, mas também a extração automática dos dados numa etapa posterior. Este processo foi, então, efetuado através da seguinte linha em *Regex*:

```
padrao_conceito = re.compile(r'^([^,]+),\s(fem.|masc.)')
```

Depois de concluída a etapa de limpeza e marcação do glossário no ficheiro *.txt*, o passo seguinte consistiu em transformar este conteúdo num formato de dados estruturados.

Foi definida a seguinte estrutura para organizar as informações extraídas deste glossário, com base nas diferentes categorias de dados relevantes para cada conceito.

```
{
    "Nome do Conceito": {
        "Traduções": {"es": "tradução", "en": "translation"},
        "Descrição": "Texto descritivo.",
        "Género": "masc",
        "Número": "pl",
        "Sinónimos": ["sinónimo1", "sinónimo2"],
        "Notas": ["Nota 1", "Nota 2"],
        "Sigla": "sigla",
        "Remissivas": ["relacionado1", "relacionado2"]
    }
}
```

De modo a alcançar esta estrutura, foi necessário realizar várias etapas de processamento no conteúdo do glossário. Um dos principais passos foi a separação do texto em blocos que representam os diferentes conceitos. Este processo foi realizado com base no marcador *(###)*, que foi inserido durante a fase de limpeza e marcação. A expressão regular utilizada para separar o conteúdo com base nesse marcador foi a seguinte:

```
conceitos = re.split(r'^###\s', conteudo, flags=re.MULTILINE)
    [1:]
```

De seguida, procedeu-se à extração das informações relevantes de cada conceito:

1. Extração do nome do conceito:

   A primeira linha de cada conceito contém o nome do conceito seguido de informações como o género e o número. A primeira tarefa foi extrair o nome do conceito dessa linha. Para isso, foi usado o seguinte código:

   ```
   first_line = conceito.split('\n')[0].strip()
   name = re.match(r'^([^,]+)', first_line).group(1).strip()
   ```

   A expressão regular captura o nome do conceito, que é tudo o que aparece antes da vírgula na primeira linha. O método *group(1)* garante que apenas o nome do conceito é extraído.

2. Extração do género e número:

   Logo após o nome, encontra-se o género, e em alguns casos, o número (singular ou plural). O género é identificado por `fem.` ou `masc.`, enquanto o número é indicado por `pl.` para plural. Para extrair essas informações, foram utilizadas as seguintes expressões regulares:

   ```
   gender_match = re.search(r',\s(fem|masc)(?:\.\spl\.)?',
      first_line)
   gender = gender_match.group(1) if gender_match else ''
   number = 'pl' if '. pl.' in first_line else ''
   ```

3. Extração dos sinónimos:

   Os sinónimos, se presentes, são identificados e extraídos de uma linha iniciada com *"Sin."*. Assim, para isso, foi utilizada a seguinte expressão *Regex*:

   ```
   sin_match = re.search(r'Sin\.\s([^.;]+(?:;\s[^.;]+)*)',
      conceito)
   if sin_match:
       synonyms = [clean_text(s.strip()) for s in re.split(r'
          ;\s*', sin_match.group(1))]
   ```

   Este código captura os sinónimos e separa-os por ponto e vírgula.

4. Extração de sigla

   Se uma sigla estiver presente, o símbolo ⇒ está presente na frase. Assim, a extração é feita através do seguinte código:

   ```
   acronym = ''
   acronym_match = re.search(r'=>\s([^.,;\n]+)', conceito)
   if acronym_match:
       acronym = clean_text(acronym_match.group(1).strip())
   ```

   Esta expressão, captura a sigla após o símbolo ⇒, considerando que pode ser seguida por uma vírgula, ponto e vírgula ou uma quebra de linha.

5. Extração da descrição

   A descrição do conceito é capturada até os campos *"Notas:"*, *"Em espanhol:"* ou *"Em inglês:"* surgirem, pois estes indicam o início de outras secções. A extração da descrição é feita da seguinte maneira:

```
desc_part = re.split(r'(?:Notas?:|Em␣espanhol:|Em␣ingles:)'
   , conceito, maxsplit=1, flags=re.IGNORECASE)[0]
desc_part = re.sub(r'^[^,]+,.*?\.\s*', '', desc_part, flags
   =re.DOTALL)
```

6. Extração de notas

   As notas são extraídas se houver uma secção chamada *"Notas:"*, que pode incluir diferentes itens numerados com recurso a letras romanas. A extração é feita através da seguinte expressão regular:

```
notes_match = re.search(r'Notas?:\s*(.*?)(?=Em␣espanhol:|Em
   ␣ingles:|$)', conceito, flags=re.DOTALL)
if notes_match:
    notes_text = notes_match.group(1).strip()
    notes = re.split(r'\s*[ivx]+\)\s*', notes_text)
```

7. Extração das traduções

   Existem dois tipos de traduções neste glossário, que são identificadas através das secções *"Em inglês:"* e *"Em espanhol:"*.

```
translations = {'es': '', 'en': ''}
es_match = re.search(r'Em␣espanhol:\s*(.*?)(?:\n|$)',
   conceito)
if es_match:
    translations['es'] = clean_text(es_match.group(1).strip
       ().rstrip(';'))

en_match = re.search(r'Em␣ingles:\s*(.*?)(?:\n|$)',
   conceito)
if en_match:
    translations['en'] = clean_text(en_match.group(1).strip
       ().rstrip(';'))
```

   Estas expressões regulares capturam as traduções, extraindo o conteúdo após as palavras *"Em espanhol:"*e *"Em inglês:"*.

8. Extração das remissivas

   Por fim, as remissivas, que indicam outras entradas relacionadas no glossário, são capturadas por duas expressões regulares. A primeira regista as remissivas com *"Ver sin."*, e a segunda captura as remissivas com *"Ver"*:

```
remissivas = []
concept_one_line = conceito.replace('\n', '␣')
ver_sin_match = re.search(r'Ver␣sin\.\s*([^.;]+(?:;\s[^.;]+)*)'
    , concept_one_line)
if ver_sin_match:
    remissivas.extend([clean_text(s.strip()) for s in re.split(
        r';\s*', ver_sin_match.group(1))])

ver_match = re.search(r'Ver\s+(?!sin\.)([^.;]+(?:;\s[^.;]+)*)',
    concept_one_line)
if ver_match:
    remissivas.extend([clean_text(s.strip()) for s in re.split(
        r';\s*', ver_match.group(1))])
```

4 Estrutura de dados final

De modo a consolidar a informação proveniente dos dicionários, foi necessário implementar um processo de fusão de dados, assegurando que nenhuma informação se perdesse no caso de existirem termos repetidos entre diferentes fontes.

A estrutura final foi definida como um único ficheiro JSON, onde cada chave representa um termo e o respetivo valor agrega toda a informação disponível nos vários dicionários sobre cada termo. Para isso, foi necessária uma função de fusão capaz de lidar com os diferentes tipos de dados associados a cada termo: strings, listas ou dicionários. A função desenvolvida, *merge*, é chamada quando um termo aparece em mais do que um dicionário e recebe como parâmetros os valores associados a esse termo. Se ambos os valores forem dicionários, compara cada chave interna e, caso a chave esteja nos dois, chama recursivamente a função *merge* para fundir os valores associados, caso a chave só esteja num deles, mantém esse valor. Se ambos os valores forem listas, junta as listas removendo os duplicados. Por outro lado se um for lista e outro não, junta o elemento à lista. Por fim, se forem de tipos diferentes, junta ambos numa lista, sem duplicar.

5 Conclusão

O desenvolvimento deste projeto permitiu aplicar de forma prática os conhecimentos adquiridos na unidade curricular de Processamento de Linguagem Natural, com foco na extração estruturada de informação a partir de documentos em formato PDF.

Foram processados quatro documentos, incluindo os dois de caráter obrigatório, recorrendo a técnicas de conversão, limpeza e *parsing* manual dos dados, com o auxílio de expressões regulares. A informação considerada relevante, como termos, definições, traduções, siglas e sinónimos, foi cuidadosamente extraída e organizada num único ficheiro JSON, com uma estrutura pensada para facilitar a reutilização em trabalhos futuros.

Apesar de o sistema desenvolvido se ter demonstrado eficaz na tarefa de extração de informação de dicionários médicos, em trabalhos futuros seria de interesse investir na automatização de algumas etapas e na robustez da deteção de padrões textuais. Ainda assim, este projeto evidenciou a importância da organização e da estruturação da informação em aplicações biomédicas, tornando o conhecimento mais acessível.