Proposta recebida em 7 de Maio 2018 e aceite para publicação em 16 de Julho 2018.

PLN.pt: Processamento de Linguagem Natural para Português como um Serviço

PLN.pt: Natural Language Processing for Portuguese as a Service

Nuno Ramos Carvalho
United Nations University (UNU-EGOV)
ramos.de.carvalho@unu.edu

Alberto Simões
2Ai — Polytechnic Institute of Cávado and Ave
Barcelos, Portugal
asimoes@ipca.pt

Resumo

As técnicas da área de Processamento de Linguagem Natural (PLN) são cada vez mais utilizadas para enriquecer aplicações nas mais diversas áreas. As ferramentas que implementam ou apoiam o desenvolvimento destas técnicas podem ser complexas de manter e explorar, sendo por vezes necessário conhecimento específico do domínio.

Este artigo introduz o projeto PLN.PT, uma plataforma online que disponibiliza um conjunto de ferramentas para PLN como um serviço *web* (REST API), orientado principalmente para a língua portuguesa.

Palavras chave

PLN, REST, API, serviço web, português

Abstract

Natural Language Processing (NLP) techniques are often used to enrich applications in several areas. Tools that implement or support the development of these techniques can be complex to maintain and exploit, and specific domain knowledge is usually required.

This paper introduces the PLN.PT, an online platform that enables a set of tools for natural language processing as a web-service (REST API), mainly focused on the portuguese language.

Keywords

NLP, REST, API, web service, Portuguese

1 Introdução

Se até há bem pouco tempo a investigação em Processamento de Linguagem Natural (PLN) era essencialmente académica, e poucas eram as áreas onde tal investigação era aplicada comercialmente ou com objetivos comerciais ou industriais, recentemente esta área tem vindo a ter um interesse crescente na indústria. Se áreas como a tradução automática Rychtyckyj (2006) ou técnicas como a indexação de documentos Frakes & Baeza-Yates (1992), eram já populares, cada vez mais surgem novas áreas de interesse, como a compilação automática de resumos sobre notícias e a sua agregação Mani & Maybury (1999) ou a análise de redes sociais para extrapolar a relevância e aceitação dos produtos comercializados Liu (2015).

Atualmente, não existem soluções "*out of the box*," que se possam adquirir e que implementem, como caixa negra, o que se pretende. Este tipo de aplicação é, ainda, desenvolvida caso a caso, e na sua maioria, em parcerias académicas.

A implementação destas soluções requer conhecimento específico de PLN, quer para desenhar a sua arquitetura, quer para escolher e usar as ferramentas relevantes. Se por um lado adquirir o conhecimento científico necessário é complicado, é igualmente trabalhosa a instalação e configuração das ferramentas disponíveis para as diferentes tarefas necessárias.

É neste campo que o PLN.PT pretende atuar, disponibilizando um conjunto de ferramentas de diferentes níveis de complexidade, que permitam a execução de tarefas, desde as tarefas simples de *atomização*, *segmentação* ou *anotação morfossintática* (PoS) até tarefas mais complicadas como a *análise sintática*, a *construção de árvores de dependências* e, no futuro, até mesmo a *classificação de sentimento* ou a *sumarização*.

Na verdade, qualquer aplicação que tire partido de técnicas de PLN necessita de uma pilha de ferramentas específicas de PLN, que usadas em conjunto permitam criar e explorar os recursos necessários para possibilitar as funcionalidades pretendidas. Apesar de haver uma série de pilhas de ferramentas atualmente disponíveis, como sejam o FreeLing Padró & Stanilovsky (2012), o NLTK Loper & Bird (2002), ou o

DOI: 10.21814/lm.10.1.267

OpenNLP Baldridge (2005), estes nem sempre são fáceis de instalar e de manter, já que por vezes surgem conflitos de requisitos, dependências entre bibliotecas, etc., que levam a que seja gasto muito tempo na sua configuração e que, muitas vezes, tem de ser replicada sempre que se pretende implementar uma nova aplicação. Além disso, em alguns casos, é necessário algum conhecimento específico sobre o domínio para tirar o melhor proveito de cada uma das ferramentas.

A abordagem que apresentamos tenta colmatar estas situações e dificuldades, e consiste na criação de uma plataforma central onde toda a informação é processada e os recursos são disponibilizados através de uma série de operações pré-definidas independentes do contexto. Isto permite que outras ferramentas possam usar, de uma forma expedita, estas funcionalidades, bastando para isso a realização de pedidos HTTP e serem capazes de processar resultados na notação JSON.

Este artigo pretende introduzir a plataforma e a sua arquitetura (apresentadas na Secção 2), a API atualmente disponibilizada e exemplos de uso (Secção 3), e as bibliotecas desenvolvidas que permitem a abstração do serviço REST (Secção 4). Finalmente, são apresentadas algumas considerações finais e propostas para trabalho futuro (Secção 5).

2 Plataforma e Arquitetura

A plataforma pode ser vista como um encapsulamento de bibliotecas ou aplicações existentes, e é composta por duas partes:

1. As ferramentas específicas, que implementam as várias funcionalidades disponibilizadas;
2. A transformação dos pedidos REST em parâmetros para as respetivas ferramentas e o tratamento do resultado, transformando-o numa estrutura normalizada em JSON.

2.1 Ferramentas

A plataforma não pretende ser uma implementação de uma ou mais funcionalidades referentes a tarefas de PLN, mas pretende sim disponibilizar o acesso, uniformizado, a um conjunto de ferramentas já existentes.

Para isso, foi necessário escolher um conjunto inicial de funcionalidades a disponibilizar e as ferramentas responsáveis pela sua implementação.

Como foi referido na introdução, o primeiro objetivo do PLN.pt foi o de disponibilizar as ferramentas necessárias para uma pilha de processamento de linguagem natural, que permita o tratamento inicial do texto, com os processos de atomização, segmentação, análise morfológica e análise sintática. Sendo o objetivo a disponibilização para a língua portuguesa, optou-se pela biblioteca FreeLing Padró & Stanilovsky (2012); Simões & Carvalho (2012). Para além de ser de código aberto e gratuito, suporta várias línguas. Também a sua proximidade geográfica e cultural com a língua portuguesa levou a que fosse uma escolha natural.

Embora o FreeLing inclua um analisador morfológico, o carregamento de dados é algo demorado, pelo que, embora seja suficientemente adequado para o processamento de blocos de texto, não é a ferramenta ideal quando se pretende obter as possíveis categorias e propriedades morfológicas de palavras individuais. Nesse sentido, foi também incluído o acesso ao jspell Simões & Almeida (2002), uma ferramenta mantida por um dos autores.

Finalmente, para a disponibilização de um serviço de construção de árvores de dependências, optou-se pelo uso do SyntaxNet Andor et al. (2016). O FreeLing também inclui um módulo de árvores de dependências, mas não disponibiliza um modelo para a língua portuguesa, e num projeto anterior de um dos autores já tinha sido criado um modelo para a SyntaxNet, razão pela qual foi escolhido.

Cada uma destas aplicações é executada, de acordo com uma série de opções bem definidas por omissão, e com os dados necessários, pela componente de interligação do serviço REST.

2.2 Serviço REST

A segunda componente do PLN.pt é então uma aplicação que implementa um serviço REST via HTTP, que implementa a API que permite aceder a todas as ferramentas disponibilizadas de uma forma simples e rápida. Esta aplicação está disponível no GitHub[1], sob uma licença de código aberto, e pode ser executada em qualquer sistema desde que as ferramentas necessárias estejam disponíveis.

De uma forma genérica, esta aplicação executa os seguintes passos para execução de cada pedido:

1. Receber um novo pedido através de um GET ou POST, e que respeite os parâmetros definidos pela API.
2. Utilizar as ferramentas disponíveis para executar a operação e obter um resultado bruto.

[1] https://github.com/nunorc/PLN-PT-api

3. Normalizar o resultado bruto e encapsular o resultado em formato JSON.
4. Devolver o resultado no formato correto para a aplicação cliente.

3 API e Serviços

A API disponibiliza uma série de serviços em que, apesar de serem baseados em ferramentas distintas, os resultados são sempre devolvidos usando uma estrutura coerente.

Esta secção lista os vários serviços, apresentando para cada um o tipo de método HTTP usado (GET ou POST), o endereço do serviço (*endpoint*), o corpo do pedido, sempre que este seja do tipo POST, e o resultado obtido.

3.1 Atomização

Serviço baseado no FreeLing, é responsável por dividir o texto num conjunto de átomos ou *tokens*. O resultado é uma lista de *strings*, em que cada posição corresponde a uma palavra ou a um átomo (pontuação, números, endereços web, etc).

Habitualmente, é útil o uso de segmentação e atomização sobre o mesmo texto, pelo que é possível usar a opção `sentences=1`, obtendo-se uma lista em que cada posição corresponde a uma frase ou segmento, contendo, por sua vez, uma lista dos átomos dessa frase/segmento.

POST	`http://api.pln.pt/tokenizer`
Corpo	`A Maria tem razão.`
Resposta	`["A","Maria","tem","razão","."]`

3.2 Etiquetação Morfossintática

Usa o FreeLing para, além de realizar segmentação e atomização, anotar o texto com *part-of-speech*. O resultado inclui, para cada palavra, o seu lema, etiqueta POS e confiança associada a essa etiquetação.

POST	`http://api.pln.pt/tagger`
Corpo	`A Maria tem razão.`

Resposta

```
[{"pos":"DA0FS0","form":"A","prob":"0.
    675415","lemma":"o"},{"lemma":"maria"
    ,"pos":"NCFS000","form":"Maria","prob
    ":"1"},{"lemma":"ter","form":"tem","
    prob":"0.999287","pos":"VMIP3S0"},{"
    lemma":"razão","pos":"NCFS000","form"
    :"razão","prob":"0.65"},{"lemma":".",
    "pos":"Fp","form":".","prob":"1"}]
```

3.3 Análise de Dependências

Um *parser* de dependências é capaz de analisar a estrutura gramatical de uma frase e gerar uma árvore (ou representação semelhante) das relações binárias entre os elementos léxicos, normalmente chamadas dependências (e.g., sujeito, predicado).

Esta análise é efetuada pelo SyntaxNet, o resultado é normalizado para uma estrutura bem definida, os nomes das relações usados (incluindo os pormenores que cada uma representam) estão disponíveis na coleção do Universal Dependencies[2]. O resultado é uma lista de elementos, em que para cada átomo (*token*) da frase são indicados uma série de propriedades, como por exemplo, a relação de dependência e o átomo pai.

POST	`http://api.pln.pt/dep_parser`
Corpo	`A Maria tem razão.`

Resposta

```
[{"upostag":"DET","deps":"_","head":"2","
    lemma":"_","xpostag":"art||F|S","id":
    "1","feats":"Definite=Def|Gender=Fem|
    Number=Sing|PronType=Art|fPOS=DET++
    art||F|S","form":"A","misc":"_","
    deprel":"det"},{"upostag":"PROPN","
    deps":"_","lemma":"_","head":"3","
    xpostag":"prop|F|S","misc":"_","
    deprel":"nsubj","id":"2","feats":"
    Gender=Fem|Number=Sing|fPOS=PROPN++
    prop|F|S","form":"Maria"},(...)]
```

3.4 Análise Morfológica

A análise morfológica, tal como já foi referido, é realizada pelo jspell. O serviço recebe uma palavra e apresenta uma lista de análises realizadas, incluindo o lema, categoria gramatical, e propriedades de *Part-of-Speech*.

POST	`http://api.pln.pt/word_analysis`
Corpo	`gato`

Resposta

```
[{"rad":"gatinhar","CAT":"v","TR":"i","T"
    :"p","N":"s","P":"1"},
 {"G":"m","N":"s","GR":"dim","CAT":"nc","
    rad":"gato"}]
```

4 Bibliotecas

A API pode ser utilizada a partir de qualquer linguagem de programação, desde que esta seja

[2]Disponível em: http://universaldependencies.org/ (último acesso: 05-05-2018).

capaz de realizar pedidos HTTP. Isto permite que até se possam implementar ferramentas que executem sobre um *browser*, realizando pedidos à API.

De forma a facilitar o desenvolvimento de aplicações em Perl ou em Python, são disponibilizadas bibliotecas para estas linguagens, que encapsulam os pedidos REST, permitindo ao utilizador a manipulação dos resultados sem ter de lidar com o formato JSON.

De seguida faz-se uma pequena demonstração de utilização destas bibliotecas.

4.1 Perl

A biblioteca disponibilizada para Perl, chamada `PLN::PT`[3], implementa um objeto que disponibiliza um método para cada uma das funcionalidades da API, devolvendo os resultados em estruturas de dados nativas da linguagem.

```
use PLN::PT;

my $pln = PLN::PT->new('http://api.pln.pt');
my $text = 'A␣Maria␣tem␣razão.';
my $tokens = $pln->tokenizer($text);

# ['A','Maria','tem',razão','.']
```

Listagem 1: Exemplo de script Perl.

A Listagem 1 ilustra a utilização da biblioteca para dividir uma frase em palavras. Começa por criar um objeto `$pln`, instância da classe `PLN::PT`, para depois invocar o método `tokenizer` passando-lhe, como argumento, o texto. O método devolve uma estrutura (referência para lista de palavras) em Perl.

4.2 Python

A biblioteca disponibilizada para Python, chamada `plnpt`, está disponível no GitHub[4].

O funcionamento desta biblioteca é análogo à sua congénere em Perl: um objeto é implementado que disponibiliza uma série de métodos para aceder a cada uma das operações da API.

```
import plnpt

text = 'A␣Maria␣tem␣razão.';
tokens = plnpt.tokenizer(text)

# ['A', 'Maria', 'tem', 'razão', '.']
```

Listagem 2: Exemplo de script Python.

O Exemplo 2 ilustra a utilização da biblioteca de Python, para dividir uma frase em *tokens* (palavras). É chamada o método `tokenizer` da biblioteca, e passado como argumento o texto a processar, e o resultado é uma lista de *tokens* em Python.

5 Conclusão e Trabalho Futuro

Este artigo descreve o desenvolvimento de um serviço REST que disponibiliza uma série de operações comuns usadas em aplicações da área de PLN, especialmente orientadas para o português, através de uma API. As pilhas de ferramentas e *toolkits* necessários para estas operações podem ser complexos de providenciar e manter, esta aproximação elimina completamente este esforço, e permite o desenvolvimento mais rápido e quase independente das aplicações.

A plataforma está disponível em `http://pln.pt` e, apesar de contar ainda com apenas um número limitado de operações, já se mostrou bastante útil em várias aplicações, em áreas diversas. O seu uso reduz, efetivamente não só o esforço de novas aplicações, como deixa de ser necessário conhecimento especializado para a utilização destas ferramentas.

Como trabalho futuro pretende-se a adição de novas ferramentas e bibliotecas, permitindo assim adicionar novas funcionalidades à API. Será, posteriormente, necessário garantir a sobrevivência da aplicação, aplicando restrições de uso, assim que a quantidade de utilizadores assim o exija.

Agradecimentos

Este artigo foi parcialmente desenvolvido no âmbito do projecto "SmartEGOV: Harnessing EGOV for Smart Governance (Foundations, methods, Tools) / NORTE-01-0145-FEDER-000037", cofinanciado pelo Programa Operacional Regional do Norte (NORTE 2020), através do PORTUGAL 2020 e do Fundo Europeu de Desenvolvimento regional (FEDER).

Os autores agradecem ainda ao Mário Peixoto e aos revisores da Linguamática pela revisão e comentários que ajudaram a melhorar o artigo.

Referências

Andor, Daniel, Chris Alberti, David Weiss, Aliaksei Severyn, Alessandro Presta, Kuzman Ganchev, Slav Petrov & Michael Collins. 2016. Globally normalized transition-based neural networks. *CoRR* abs/1603.06042. `http://arxiv.org/abs/1603.06042`.

[3] `https://metacpan.org/release/PLN-PT`
[4] `https://github.com/nunorc/plnpt`

Baldridge, Jason. 2005. The OpenNLP project. http://opennlp.apache.org (Último acesso: 17-10-2017).

Frakes, William Bruce & Ricardo Baeza-Yates. 1992. *Information retrieval: Data structures & algorithms*. Prentice Hall Englewood Cliffs, New Jersey.

Liu, Bing. 2015. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge University Press.

Loper, Edward & Steven Bird. 2002. NLTK: The natural language toolkit. Em *ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, 63–70.

Mani, Inderjeet & Mark T. Maybury. 1999. *Advances in automatic text summarization*, vol. 293. MIT Press.

Padró, Lluís & Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. Em *Language Resources and Evaluation Conference (LREC 2012)*, .

Rychtyckyj, Nestor. 2006. Machine translation for manufacturing: A case study at ford motor company. Em *18th Conference on Innovative Applications of Artificial Intelligence - Volume 2* IAAI'06, 1728–1735.

Simões, Alberto & Nuno Carvalho. 2012. Desenvolvimento de aplicações em Perl com FreeLing 3. *Linguamática* 4(2). 87–92.

Simões, Alberto Manuel & José João Almeida. 2002. `jspell.pm` — um módulo de análise morfológica para uso em processamento de linguagem natural. Em *Actas da Associação Portuguesa de Linguística (APL2001)*, 485–495.