Utilização de técnicas de Machine Learning e PLN para a criação de um Bot

Carlos António Senra Pereira[1]

[1] Mestrado em Matemática e Computação
Departamento de Matemática
Universidade do Minho

12 de Julho de 2018

Conteúdos

Introdução

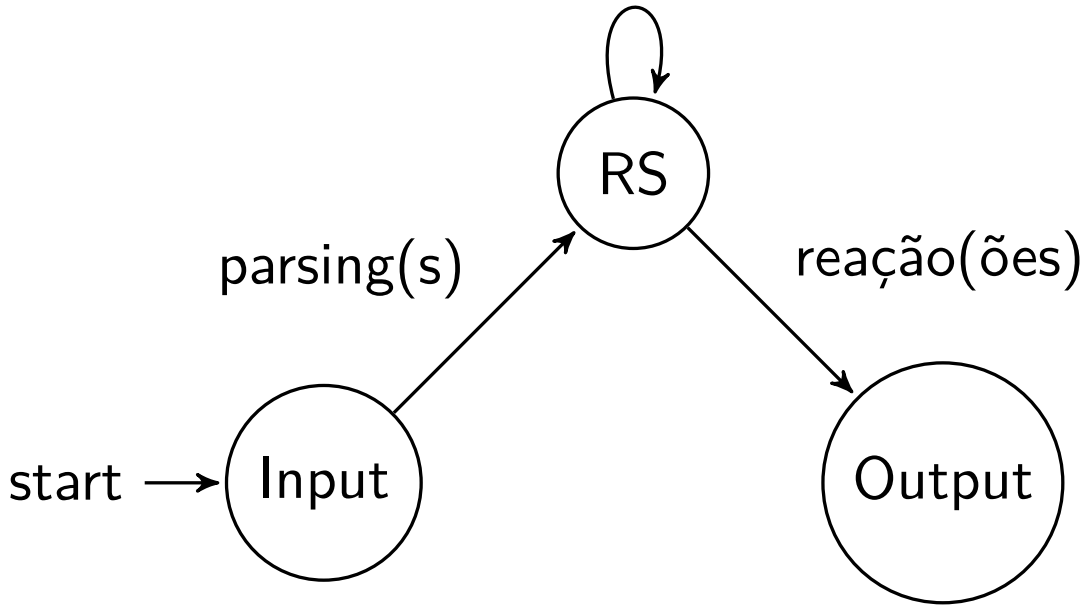
O processamento de Linguagens Naturais é bastante complexo, e ainda mais no ponto de vista de uma máquina determinística.

Por isso, utilizando técnicas de Machine Learning é possível construir modelos capazes de analisar e frases e tomar ações com bastante precisão.

Utilizando Padrões Linguísticos e sistemas de reescrita, podemos criar assim um Bot onde a análise é feita num domínio de pesquisa ou aplicação definido pelo utilizador, através dos Padrões Linguísticos definidos no Modelo do Bot.

Conteúdos

Introdução - Estrutura do Bot



- RS: Representações Semânticas

Conteúdos

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Objetivos

1. Desenvolver uma aplicação de Chat;
2. Utilizar o FreeLing para analisar frases e devolver o PoS e as dependências sintáticas;
3. Criar um modelo linguístico para permitir o uso de padrões linguísticos e sistemas de reescrita;
4. Construir uma base de dados capaz de armazenar conhecimento para a aprendizagem;
5. Escrever uma função para extrair e/ou combinar *features* da análise de uma frase;
6. Definir um modelo de Machine Learning para classificar a ação a tomar após a análise;
7. Definir ontologias para gerar expressões relativas a um certo domínio de pesquisa ou aplicação.

Conteúdos

Modelo

Após a análise do problema em questão, foi criado um modelo de um Bot.

$$Bot ::= Regra^*$$

$$Regra ::= (\quad antecedente : ERL \quad , \quad reação : FR^* \quad)$$

Onde:

1. *FR*: Função de Reação;
2. *ERL*: Expressão Regular Linguística.

Conteúdos

Expressões Regulares Linguísticas

Exemplo

Comecemos com um exemplo prático do nosso modelo:

$$
\begin{aligned}
&\mathit{pergunta\_pessoa} = \{\mathit{exp} : [\\
&\quad (\backslash w^*, \quad \mathit{pronome\_geral}, \quad \mathit{None}), \\
&\quad (\backslash w^*, \quad \mathit{verbo\_geral}, \quad \mathit{None}), \\
&\quad (\backslash w^*, \quad \mathit{nome\_proprio\_geral}, \quad \mathit{name}), \\
&\quad (.^*, \quad \mathit{pergunta}, \quad \mathit{None}) \\
&], \mathit{action} : [\mathit{pergunta\_pessoa}]\}
\end{aligned}
$$

Expressões Regulares Linguísticas

Abstração

Estas são as expressões definidas serão compiladas e interpretadas pelo Bot.

$$
\begin{aligned}
ERL &::= EL^* \\
EL &::= (IT^*, \quad FR^*) \\
IT &::= (pal : ER, \quad tag : ER, \quad catch : palavra) \\
FR &::= \text{Função}
\end{aligned}
$$

Onde:

1. *ER*: Expressão Regular;
2. *pal*: palavra;
3. *tag*: PoS;
4. *catch*: captura.

Conteúdos

Função Verifica

No momento em que o utilizador introduz uma frase, o Bot terá de verificar se a frase introduzida tem correspondência. É nesse momento que a função verifica atua.

$$verifica(frase, EL) \mapsto (id \hookrightarrow val) \quad | \quad \perp$$

Esta função aceita uma frase e uma expressão linguística, e devolve um de dois possíveis resultados:

- um dicionário com o id e valor das expressões que se captaram;
- ou vazio quando a expressão não corresponde à frase.

Conteúdos

Função Executa

Este é o esqueleto de funcionamento de um *Bot*.

$$
\begin{aligned}
&executa(Bot, str): \\
&\quad v \leftarrow [verifica(str, ant) \quad for(ant, rea \in Bot)] \\
&\quad f \leftarrow sortear(v) \\
&\quad f(v)
\end{aligned}
$$

Conteúdos

Implementação

Exemplo

Um exemplo de uma expressão deste modelo é:

$$.^*/\$adverbio\_geral \quad .^*/\$verbo\_geral \quad .^*/\$pronome\_geral? \quad (name = .^*)/\$nome\_proprio\_geral \quad .^*/Fit?$$

Que corresponde a:

Onde fica o Porto?

Implementação

Explicação

Explicação:

- $.*/.*$: podemos ter qualquer coisa de um tipo qualquer;
- $.*/\$verbo$: podemos ter qualquer coisa de um tipo chamado *verbo*;
- $(lugar = .*)/\$cidades$: podemos ter qualquer coisa, que vai ser guardada numa variável chamada de *lugar*, captura de variáveis, do tipo cidades;
- $.*/.*?$: podemos ter ou nao qualquer coisa de um tipo qualquer.

Conteúdos

Implementação

Gramática

Após uma análise do modelo linguístico, foi criado uma gramática para a sua implementação.

$$
\begin{aligned}
ERL &::= EL^* \\
EL &::= ER \quad \text{“/”} \quad IT \quad ?^* \\
ER &::= IT \quad | \quad (\backslash w = IT) \\
IT &::= \$\backslash w \quad | \quad \backslash w
\end{aligned}
$$

Implementação

Gramática

Onde,

$$
\begin{aligned}
ER &: \text{Expressão Regular} \\
IT &: \text{Item} \\
\backslash w &: [a-zA-Z0-9]^* \\
ERL &: \text{Expressão Regular Linguística} \\
.^* &: \backslash w^*
\end{aligned}
$$

Conteúdos

FreeLing

Descrição

- Biblioteca escrita em C++ com wrappers em Java e Python;
- Analisa frases morfossintáticamente, através de um classificador treinado pelo *Bosque* e *Label-Lex*;
- Analisador bastante preciso.

FreeLing

Servidor/Cliente

Devido ao facto do modelo de análise do *FreeLing* demorar bastante tempo a ser construído, utiliza-se um servidor já incluído no FreeLing que devolve em formato JSON a análise.

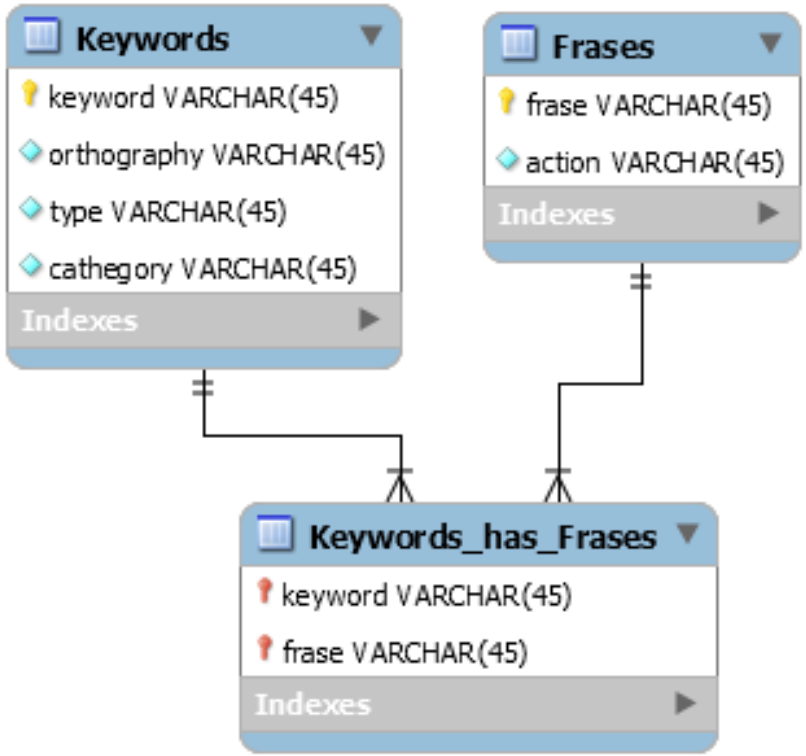
Neste caso devolve o PoS e a árvore de dependências sintáticas.

Conteúdos

Base de Dados de Conhecimento

Base de dados:

- Palavras-chave:
  tipo, categoria e ortografia
  de palavras-chave
  consideradas importante;
- Frases: histórico
  de frases processadas.



Conteúdos

Classificador de Respostas

- Após a análise do FreeLing, obtemos o PoS da frase de entrada. Depois, segue uma verificação ao modelo linguístico para ver se é detetado algum padrão.
- De seguida, passámos para a função que extrai as *features* da frase.
- Estas features são uma combinação da verificação de padrões do modelo linguístico, bem como a comparação com outras frases já analisadas e também a presença de certas palavras-chave na frase.