Etiquetador morfo-sintáctico para o Português

Ricardo Reis
miguel@ci.uminho.pt
José João Dias de Almeida
jj@di.uminho.pt

1997

Resumo

Neste artigo apresenta-se um etiquetador para Português baseado no Etiquetador Dirigido por Regras de Eric Brill[4], com utilização adicional de um nível de preprocessamento (opcional), de um analisador morfológico (jspell[3]) e de um nível de análise sintáctica (opcional). O analisador morfológico destina-se a facilitar o processo de aprendizagem e de tratamento de palavras desconhecidas, bem como a preparação de um *Corpus* de treino inicial.

O conjunto de etiquetas usado tem como parâmetros Categoria Gramatical, Sub-Categorias (quando existem), Género, Pessoa e Número, com uma cardinalidade próxima de 200. Cada etiqueta tem uma leitura hierárquica das características que a compõem. Este conjunto de etiquetas foi criado de raiz com a intenção de se adaptar bem à Língua Portuguesa.

Para a aprendizagem foi usado um *corpus* de perto de 10000 palavras, extraído do *corpus* "Natura-Público", etiquetado por *bootstrapping*.

1 Introdução

Nesta secção apresentam-se as fases do projecto, pela ordem em que foram executadas. Antes de iniciar o treino para Português, foi necessário estabelecer um conjunto de etiquetas para uso, já que as existentes eram específicas para o Inglês. Em seguida, descreve-se o treino e respectivos refinamentos. Finalmente, introduz-se a fase de análise sintáctica.

1.1 Estabelecimento do Conjunto de Etiquetas

Sendo a Língua Portuguesa de origem latina, tem uma relativa complexidade morfológica. A definição de um conjunto de etiquetas para o Português é uma tarefa complicada e fulcral e corresponde também a um compromisso entre a precisão na descrição e a capacidade de aprendizagem de regras. Decidiu-se seguir alguns princípios funcionais:

- tentar ser o mais preciso possível:
- tentar seguir modelos já usados em experiências com outras línguas;
- definir uma estrutura hierárquica para a etiqueta.

Ficou decidido que a nomenclatura seria de alguma forma hierárquica, isto é, cada campo da etiqueta teria um significado específico e estes campos deveriam ser o mais possível reduzidos. A maior parte destes resultou numa só letra.

Exemplo:
`empatou/VIP3S` – `V`=Verbo, `IP`=Pretérito perfeito do indicativo, `3`= 3.a pessoa, `S`=singular

Podemos assim definir uma etiqueta com a seguinte estrutura:

```
Etiqueta: Categoria Sub-Categorias Género Pessoa Número
Categoria: QUE | SE | D | P | N | J | V | ADV | C | I | & | ?
```

referentes a **que**, **se**, Determinantes, Pronomes ou Preposições, Nomes, adJectivos, Verbos, ADVérbios, Conjunções, Interjeições, Contracções e palavras desconhecidas.

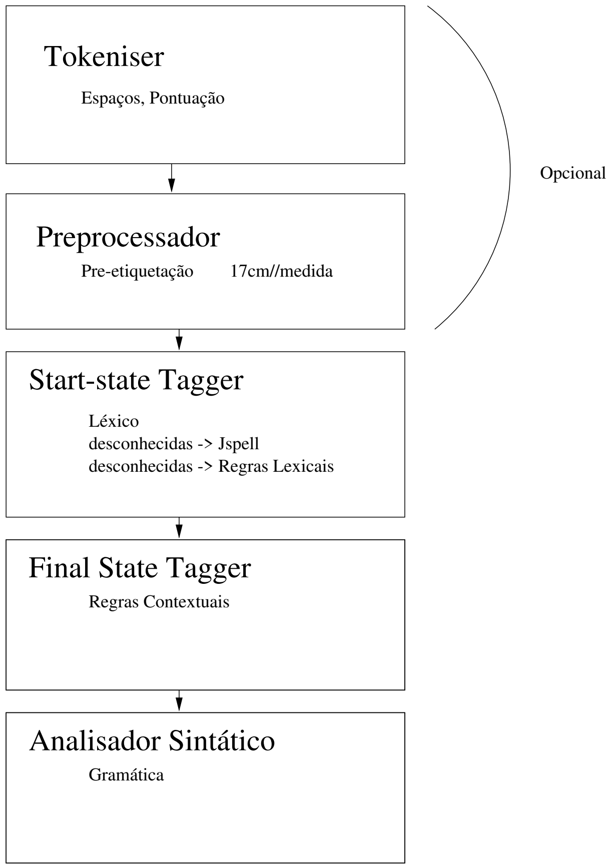
As sub-categorias vão depender de cada categoria, naturalmente, sendo os restantes elementos ortogonalmente flexíveis dentro de espaços sobejamente conhecidos.

A `Categoria` é o único campo obrigatoriamente preenchido. No caso de preposições ou interjeições será mesmo o único a ter valor. A gama de valores (ou mesmo a existência) dos outros campos depende desta, mas os campos de `Género`, `Pessoa` e `Número` são unificados por forma a permitir uma futura *Análise de Concordâncias* que poderá servir tanto de critério de desempate (p.ex. verbos com formas iguais) como de correcção ortográfica.

O conjunto actual de etiquetas encontra-se parcialmente incompleto. Certas categorias não estão suficientemente refinadas, como as conjunções e os advérbios. No caso destes últimos, a sua etiqueta deverá ser mudada para `A` em vez da actual `ADV`.

Um resumo das etiquetas geradas no presente trabalho encontra-se no Apêndice B.

1.2 Arquitectura do Analisador



1.3 Aprendizagem

O etiquetador de base seleccionado foi o Etiquetador Dirigido por Regras de Eric Brill. Este etiquetador tem as seguintes características principais:

É dirigido por regras — isto significa que usa regras para determinar a etiqueta de cada palavra, (por oposição à classe de etiquetadores estocásticos, que são dirigidos por tabelas de probabilidades;)

Faz aprendizagem das regras — extrai de um *corpus* previamente etiquetado a informação morfológica e (proto-)sintáctica necessária para o seu funcionamento, guardando-a sob a forma de regras. Esta fase denomina-se *aprendizagem*.

Usa regras lexicais — As regras lexicais referem-se à grafia das palavras e actuam isoladamente em cada palavra. Servem para determinar a etiqueta de palavras desconhecidas (não presentes no léxico).

Usa regras contextuais simples — As regras contextuais servem para corrigir possíveis erros das anteriores. Para tal, guiam-se pela vizinhança (um contexto reduzido) da palavra em questão, analisando etiquetas ou palavras até uma vizinhança de 4 palavras.

A aprendizagem baseia-se em *Córpora* (previamente) etiquetados. Quanto maior for esta base de treino, melhores, porque mais exactas e abrangentes, serão as regras deduzidas. Os *Córpora* empregues para este fim, em outras línguas, ascendem a, no mínimo, 500 000 palavras.

Infelizmente, ainda não existe tal recurso na língua portuguesa, pelo que a solução é criar um. Isto pode ser conseguido através de *bootstrapping*, um método que consiste em:

1. etiquetar uma pequena porção de texto manualmente
2. usá-la como *corpus* de treino
3. etiquetar mais uma porção de texto usando as regras deduzidas
4. corrigir os erros manualmente
5. voltar ao passo 2

até se atingir um volume de texto suficiente.

Este processo é, apesar de tudo, muito moroso.

1.4 Redução do esforço de aprendizagem

Por forma a obviar este problema, optou-se por alterar o processo de aprendizagem. Assim, empregou-se um *corpus* de aprendizagem muito mais pequeno (±10 000 palavras), recorrendo-se a um dicionário/analisador morfológico externo em caso de palavras desconhecidas. Deste modo, quando o etiquetador encontra uma palavra desconhecida, isto é, uma palavra que não consta do léxico de base, antes de recorrer às regras lexicais consulta um *guesser*.

A implementação actual abre um *pipe* bidireccional para um programa Perl que invoca o JSpell e traduz a respectiva resposta (Guesser)

1.5 Analisador morfológico JSpell

O analisador morfológico JSPELL baseia-se num dicionário [1] e num conjunto de regras (ficheiro dos afixos).

Pode funcionar de vários modos:

- como programa interactivo com menus e opções
- como interpretador de linha - o utilizador introduz uma palavra e recebe imediatamente informação sobre essa palavra. Este modo pode ser usado em *pipes*.
- como comando de linha
- na forma de uma biblioteca C.

No modo interactivo com menus funciona de forma semelhante a um corrector ortográfico, detectando erros, dando sugestões de correcção e permitindo inserir num dicionário pessoal.

A existência de uma biblioteca C é praticamente indispensável para a construção de aplicações usando o JSpell, tanto em C como noutras linguagens.

O JSpell pode funcionar com várias línguas e vários dicionários. No caso presente o dicionário usado (Português) contem cerca de 40 000 radicais e um ficheiro de cerca de 500 regras morfológicas.

1.6 Pré-processadores

1.6.1 Separação

A separação é uma operação que consiste em separar uma frase nos seus elementos constituintes A necessidade desta operação prende-se com as diferenças entre as normas de elaboração de documentos electrónicos e o formato requerido pelo etiquetador de base.

Esta separação não é tão simples como possa parecer, obrigando a análise de contexto de modo a tentar tratar convenientemente as abreviaturas, sinais de pontuação compostos, etc.

Algumas expressões científicas e/ou matemáticas requerem também tratamento especial, nomeadamente aquelas em que os pontos finais fazem par-

[1] na realidade em 3: dicionário geral, dicionário pessoal e dicionário de sessão

te do formato padrão — vírgulas decimais, pontos de separação de classes numéricas[2] , factoriais, etc.

Na sua versão actual, o separador funciona em cascata, implementando os dois pontos acima apresentados e, em seguida, corrigindo os erros gerados nas situações descritas.

1.6.2 Pré-etiquetação

As situações de erro de separação acima descritas são normalmente referentes a símbolos bem definidos, pelo que pareceu natural aproveitar a ocasião para etiquetá-los logo nesta fase. Este procedimento é possível porque o etiquetador aceita texto pré-etiquetado, sendo estas etiquetas mantidas inalteráveis, isto é imunes às regras de transformação, através de todo o processo. O risco deste *congelamento* de etiquetas é praticamente nulo[3] pois, como já foi mencionado, os casos são bastante típicos.

A vantagem deste procedimento está no relaxamento (ainda que pontual) proporcionado ao etiquetador, sendo então desnecessário acumular abreviaturas no léxico.

Mas as vantagens deste método não se ficam por aqui. De facto, foi imediato constatar que, para além das abreviaturas, havia outra grande classe de lexemas que seria bastante útil excluir das bases de dados fonte: as Expressões Regulares (ERs). Esta classe de lexemas é aberta e torna-se impossível atingir qualquer índice aceitável de cobertura da mesma sem recorrer ao consumo exorbitante de espaço em disco para o Léxico.

Por outro lado, as ERs gozam de algumas propriedades de definição que as tornam representáveis em compreensão de uma forma imediata, compacta e expressiva. Alguns casos de ERs seriam todas aquelas que empregam algarismos no seu formato: números, horas, medidas, resultados desportivos, ...

Outra classe em que se torna necessária a pré-etiquetação é a dos textos anotado; embora mais complicada pelo facto de esse mesmo texto poder assumir funções sintácticas dentro da frase. É esse o caso de fracções de texto anotado em TEX, LATEX, HTML ou SGML.

Um breve exemplo é apresentado no Apêndice A.

---

[2]Em Portugal, o caracter de separação entre a parte inteira e a parte decimal de um número fraccionário é a vírgula. O ponto, por outro lado, pode ser empregue para separar as classes (unidades, milhares, milhões, ...) embora hoje em dia seja mais usual separá-las com um ligeiro espaço.

[3]Isto é, ainda não nos deparámos com situações de erro decorrentes desta prática ...

2 Analisador Sintáctico

O Analisador Sintáctico constitui o passo seguinte em relação à etiquetação. Este módulo aceita um texto etiquetado e acrescenta-lhe anotação sintáctica, num formato diferente do anterior: agrupa unidades com parêntesis, de modo a representar uma árvore sintáctica.

Várias opções de analisadores sintácticos estão previstos no nosso estudo, desde o uso de ferramentas habituais no processamento de linguagens formais (YACC, PCCTS, etc) passando pelo uso de *Chart Parsing*, generalised LR Parsing (Algoritmo de Tomita) até ao uso de gramáticas de unificação (DCG, LFG, HPSG). Uma solução importante que pretendemos analisar é o *'Parentetisador' de Brill.*

Finalmente, uma vez que o YACC é uma ferramenta bem conhecida da equipa, entendemos ser útil investigar e testar a sua adequação a problemas tipicamente não Gramáticas Independentes do Contexto (GIC), como é o presente.

No presente os resultados referem-se apenas à versão YACC.

2.1 YACC

O YACC foi a ferramenta escolhida para a implementação desta fase. Como é sabido, esta ferramenta apenas implementa GICs. Estas gramáticas apresentam as seguintes vantagens:

simplicidade — a gramática é fácil de construir e de compreender;

rapidez — o autómato gerado é eficiente.

Por outro lado, estas vantagens exibem alguns custos:

- resposta limitada perante situações de **ambiguidade**:

  lexical — a/P *vs.* a/DAFMS; esta dificuldade deve-se à impossibilidade de especificar contexto no lado esquerdo das produções e é contornável à custa da clareza, especificidade e dimensão da gramática, explicitando uma ou outra palavra nas regras, protegendo assim sintacticamente uma ambiguidade lexical;

  sintáctica — acoplamento de frases preposicionais; esta ambiguidade, aliás comum a quase todas as línguas já cobertas por estudos de etiquetação, vê-se agravada no caso da implementação em YACC, visto esta ferramenta não ter uma implementação nativa de *backtracking* que lhe permita *falhar* um caminho e tentar outro; com este mecanismo, seria trivial a produção actual recusar o texto de entrada baseada nesse mesmo texto (ou numa preposição, mais concretamente), passando-se automaticamente à produção seguinte.

- dificuldade em lidar com a **complexidade** de uma gramática extensa e rica em casos particulares; locuções e expressões idiomáticas são dois fenómenos que concorrem para esta categoria de problemas, pois a sua função sintáctica *primária* foi substituída por outra baseada mais em senso comum do que em regras;
- problemas com a **ordem livre** de certas sequências Nome-Adjectivo; este tipo de liberdades linguísticas redunda na explosão das possibilidades, levando quase sempre à replicação do código de tratamento da frase em questão.

2.1.1 'Parentesisador' de Brill

O próprio autor do etiquetador de base disponibiliza um conjunto de ferramentas relacionadas com análise sintáctica de textos previamente etiquetados. Algumas destas soluções pareceram algo atractivas e válidas.

No entanto tem surgido alguns problemas. Por um lado, afiguraram-se-nos pouco analíticas, isto é, o seu algoritmo, talvez devido à escassa documentação nesse sentido, ao contrário do caso do Etiquetador, foi avaliado como algo heurístico, sem prejuízo da avaliação de desempenho do mesmo. Por outro lado, o facto de haver muita dificuldade de ligação ao servidor respectivo para *download* das mesmas acabou por ditar em parte a sua sorte no âmbito do presente projecto.

2.2 Problemas

O maior problema foi sempre a falta de *Corpus* etiquetado para treino do etiquetador. Esta lacuna é tanto mais grave quanto se considerar que, por um lado, a especialidade desta equipa não é Linguística mas Ciências da Computação, e que, por outro, existem *Córpora* etiquetados em Portugal, mas infelizmente, não se encontram, por motivos pouco claros, disponíveis. Assim, a magra produção neste domínio é manifestamente insuficiente para o fim em vista e não estará canonicamente correcta, apesar das sucessivas filtragens a que tem sido submetida. Como consequência, as regras geradas pelo etiquetador são pobres em duas dimensões ortogonais: cobrem apenas uma parte reduzida do espectro sintáctico da Língua Portuguesa e, em cada categoria, não cobrem todas as flexões parciais. A solução possível, neste momento, é a codificação manual das regras em falta.

Outro aspecto fundamental é de que a gramática não é aprendida mas codificada, isto é, fixada na fase de programação, tornando-se cada versão do programa um cristal sintáctico insensível a quaisquer mudanças ou erros de princípio. A este problema acresce o da falta de cobertura devida ao estado incipiente da gramática já codificada.

O já focado problema da falta de *backtracking* é um obstáculo importante, com consequências directas por exemplo no acoplamento de frases preposicionais.

2.3 Resultados

Nesta sempre ingrata secção expomos os resultados obtidos com o emprego do analisador. As ferramentas de análise estatística empregues foram aquelas que vinham no pacote do Etiquetador Dirigido por Regras de Eric Brill.

Em relação ao acerto lexical, este cifrou-se na vizinhança dos 96%. Este acerto é função da qualidade do Etiquetador de base (e respectivo paradigma), da qualidade do texto de treino e da qualidade das ferramentas de análise morfológica acrescentadas ao referido etiquetador (neste caso, o JSpell).

Quanto ao acerto sintáctico, este é mais difícil de quantificar, pois a cobertura é ainda reduzida. No entanto, nas frases cobertas, este acerto é muito elevado pois o autómato gerado pelo YACC é não ambíguo. A verdadeira dificuldade de acerto no aspecto sintáctico reside na elaboração da gramática e não no seu funcionamento.

2.4 Velocidade

Em seguida apresentam-se alguns valores resultantes de medições efectuadas várias vezes sobre o mesmo texto, por forma a obter valores médios consoante a carga da máquina hospedeira:

- lexical: $>$ 200 palavras/segundo
- sintáctica: 25 frases/segundo $\mapsto$ 250 palavras/segundo;

3 Trabalho Futuro

3.1 Problemas Pendentes

Alguns aspectos deste Analisador deverão ser mudados num futuro próximo. Por exemplo, algumas etiquetas de super-categorização, como "adjectivos ou nomes comuns"(X__) foram introduzidas pelo *Guesser*, no intuito de serem resolvidas pelas regras contextuais, uma vez que não se encontram no *Corpus* de treino. Algumas não o foram devido à insuficiência daquelas, sendo passadas à fase sintáctica, onde não estão previstas, e gerando assim erros sintácticos.

As contracções estão todas aglutinadas numa só categoria, quando na realidade a existência dessa mesma categoria é questionável face a trabalhos anteriores que elegem a alternativa de as separar em palavras constituintes.

Este caso está relacionado com o dos clíticos verbais, que estão a ser ignorados. Em ambos os casos o ideal seria dividir cada palavra composta (contracção, conjugação pronominal, etc.) nos lexemas constituintes, etiquetando-os individualmente. Felizmente, estamos apenas a ignorar informação disponibilizada pelo JSpell, pelo que a qualquer momento podemos incluí-la no sistema.

Desta divisão surgiria um problema partilhado também pelas locuções e nomes próprios compostos: a perda do significado conjunto das partículas constituintes.

3.2 Solução

Na secção anterior vimos como cada problema podia ser parcialmente resolvido, à custa do levantamento de outro mais profundo. De facto, todos eles iam desembocar num só: a atribuição de significado a expressões compostas por vários lexemas.

Lembre-se que o mecanismo de etiquetação é, neste aspecto, manifestamente insuficiente, pois só atribui significado a um lexema de cada vez, por muito que consulte uma vizinhança (restrita) deste antes de decidir. Mesmo a nível sintáctico, a forma da etiqueta é uma sequência de letras (cf B) separadas da palavra por uma barra (<palavra>/<etiqueta>), não havendo assim qualquer mecanismo de agrupamento ao nível da etiquetação.

A forma actual torna-se então insuficiente para resolver os problemas levantados, que requerem a possibilidade de definir agrupamentos. A solução passa por substituir a forma de barras por uma de **etiquetação estrutural**, em árvore (**parêntesis tipado**). Desta maneira, a nova forma seria algo como (<expressão>)/<etiqueta>, em que expressão seria qualquer coisa desde uma palavra simples (não etiquetada) até um conjunto de palavras (já etiquetadas individualmente), sendo `etiqueta`, neste último caso, uma característica qualquer comum a este conjunto de palavras, por exemplo, um identificador de verbo com clítico, uma etiqueta de locução, ou até mesmo uma marca de sintagma.

Desta maneira, dá-se uma junção da noção de estrutura entre palavras e sintagmas, passando todos estes componentes a constituir numa só estrutura hierárquica. Esta evolução é fundamental pois permite uma grande generalização no suporte formal de qualquer etiqueta, bem como no tratamento (localização, identificação e interpretação) da mesma. No que se refere à apresentação (textual ou gráfica) da árvore assim representada, as funções serão exactamente as mesmas para a apresentação de qualquer sub-árvore, seja ela uma palavra (e respectiva etiqueta), seja uma frase completa.

Para este tipo de etiquetação será usado o formato SGML ou do seu subconjunto, o XML. A adopção de um destes formatos permite também o uso de ferramentas a eles associadas e permitir uma maior troca de informação no âmbito do desenvolvimento da vertente de tradução automática.

O grande obstáculo a vencer é o do formato interno do etiquetador de base. Este foi desenvolvido sobre o modelo `<palavra>/<etiqueta>`, pelo que o seu *output* obedece a este formato. Mais complicado se torna este aspecto se considerarmos que o etiquetador é constituído por dois programas separados, articulados em *pipe*. Se por um lado se torna trivial alterar o *output* do primeiro programa (`start-state-tagger`), já será mais complicado alterar o reconhecimento do *input* do segundo (`final-state-tagger`). Para já, a solução preconizada passa por um conversor de formatos, pois, sendo o modelo de parêntesis tipado um *superset* do modelo de barras, a conversão resume-se a uma simples e regular tradução.

Referências

[1] J.J. Almeida. Nllex – a tool to generate lexical analysers for natural language. *SEPLN - Procesamento del lenguaje Natural*, (19):81–90, Sep 1996.

[2] J.J. Almeida and Ulisses Pinto. Jspell – um módulo para análise léxica genérica de linguagem natural. In *Actas do Congresso da Associação Portuguesa de Linguística, Évora*, 1994.

[3] J.J. Almeida and Ulisses Pinto. Manual de utilizador do JSpell. Manual, Universidade do Minho, July 1994.

[4] E. Brill. A simple rule-based part of speech tagger. In *Third Conference on Applied Natural Language Processing*, 1992. ACL, Trento, Italy.

[5] E. Brill. *A Corpus-Based Approach to Language Learning.* PhD thesis, 1993.

[6] E. Brill. Some advances in rule-based part of speech tagging. In *Twelfth national conference on artificial intelligence (AAAI-94)*, 1994. Seattle, Wa.

[7] Ulisses Pinto and J.J. Almeida. Tratamento automático de termos compostos. In *Actas do XI Congresso da Associação Portuguesa de Linguística, Lisboa 1995*, volume 2, 1996.

A Exemplo de Pré-etiquetação

Texto fonte:

```
Às 18h00m do dia 7, o Salgueiros empatou 2-2 com o
Benfica mantendo a tradição.
```

Texto etiquetado:

```
Às/\&FP 18h00m//HORA do/\&MS dia/NCMS 7//DNCNP ,/,
o/DADMS Salgueiros/NCMP empatou/VIP3S 2-2//RESUL
com/P o/DADMS Benfica/NPMS mantendo/VG a/DADFS
tradição/NCFS ./.
```

com o seguinte preprocessador:

```
...
[0-9]+          {ECHO;printf("//DNCNP");}
[0-9]+h[0-9]+m {ECHO;printf("//HORA");}
[0-9]+-[0-9]+  {ECHO;printf("//RESUL");}
...
```

B Etiquetas usadas

Flexões comuns a muitos tipos:

género — M - masculino; F - feminino;

número — S - singular; P - plural;

pessoa — 1, 2, 3 - 1^a, 2^a e 3^a;

Etiquetas:

Determinantes — podem ser artigos ou numerais; flectem em tipo, género e número; Etiqueta: Dcategoriatipogéneronúmero

- **categoria** — DA - artigo; DN - numeral;
- **tipo-A** — DA: D - definido; I - indefinido;
- **tipo-N** — DN: C - cardinal; O - ordinal;

Pronomes — podem ser pessoais, relativos, possessivos, demonstrativos, interrogativos ou indefinidos; flectem em caso, género, pessoa e número; Etiqueta: Pcategoriacasogéneropessoanúmero;

- **categoria** — PS - pessoal; PR - relativo; PP - possessivo; PD - demonstrativo; PI - interrogativo; PF - indefinido;
- **caso** — N - nominativo; A - acusativo; G - genitivo; D - dativo;

Nomes — podem ser comuns ou próprios[4]; flectem em género e número; Etiqueta: Ncategoriagéneronúmero;

- **categoria** — NC - comum; NP - próprio;

Adjectivos — flectem em género e número; Etiqueta: Jgéneronúmero;

Nomes ou Adjectivos — flectem em género e número; esta categoria é provisória, devendo as regras contextuais resolvê-la em NC ou J; Etiqueta: Xgéneronúmero;

Verbos — flectem em tempo, modo, género (particípio passado); pessoa e número; esta flexão não é total, pelo que agrupámos as combinações de tempo e modo numa composta, designada categoria; Etiqueta: Vcategoriagéneropessoanúmero

- **categoria** — formas nominais: N - infinitivo impessoal; PP - Particípio Passado; G - Gerúndio; modo Indicativo: IH - presente (Hoje); IP - pretérito Perfeito; II - pretérito Imperfeito; IM - pretérito Mais-que-perfeito; IF - Futuro; modo conjuntivo (Se): SH - presente (Hoje); SI - pretérito Imperfeito; PI - Futuro (ver Infinitivo Pessoal ou Presente); modo iMperativo: MH - presente (Hoje); modo Condicional: CH - presente (Hoje); modo Infinitivo (Pessoal ou Presente): PI

[4] Outras categorias, como colectivos, são integradas nos comuns

Preposições — fixas; Etiqueta: P

Advérbios — fixos; Etiqueta: ADV

Conjunções — fixas; Etiqueta: C

Interjeições — fixas; Etiqueta: I

Contracções — flectem em género e número; Etiqueta: &génerонúmero

Casos Particulares — as palavras *que* e *se* têm etiquetas iguais a elas próprias: QUE e SE.