JWN-Br – Uma API Java para a WordNet.Br

Vitor Machado Oliveira[1], **Norton Trevisan Roman**[1]

[1]Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (USP)
São Paulo, SP – Brazil

{vitor.machado.oliveira,norton}@usp.br

Abstract. *This paper introduces JWN-Br – an API for Wordnet.br. Developed in Java, JWN-Br is responsible for (i) providing direct access to Wordnet.br, structuring its data as Java objects; (ii) abstracting away Wordnet.br's internal codification, making its use easier to the programmer; and (iii) giving room for the local augmentation of Wordnet.br's original data set. Distributed under GPL, JWN-Br is available to the community.*

Resumo. *Este artigo apresenta a JWN-Br – uma API para a Wordnet.br. Desenvolvida em Java, a JWN-BR possui as vantagens de (i) fornecer acesso direto à Wordnet.br, estruturando seus dados na forma de objetos Java; (ii) abstrair a codificação interna usada pela Wordnet.br, tornando seu uso mais fácil ao programador; e (iii) possibilitar o incremento local da base de dados original da Wordnet.br. Distribuída sob a GPL, a JWN-Br está disponível para a comunidade.*

1. Introdução

Consistindo basicamente de uma base de dados lexical, a Wordnet [Miller 1995] fornece a seus usuários um conjunto de substantivos, verbos, adjetivos e advérbios em inglês, organizados em grupos de sinônimos, denominados SynSets. Esses grupos, por sua vez, são ligados entre si por relações semânticas. Dentre estas, podemos citar a sinonímia (relação simétrica em que duas palavras possuem o mesmo significado), a antonímia (relação também simétrica, em que duas palavras possuem significado contrário), a hiponímia e a hiperonímia (relações assimétricas que estabelecem uma hierarquia entre palavras)[1]. Naturalmente, por ser organizada em SynSets, a sinonímia constitui a relação básica da Wordnet.

Originalmente desenvolvida na Universidade de Princeton[2], a Wordnet serviu de inspiração para várias iniciativas semelhantes, sempre compatíveis com a original. Estas, por sua vez, foram desenvolvidas para as mais diversas línguas, do Japonês [Bond et al. 2010] ao Português Europeu [Marrafa 2002], passando pelo Árabe [Elkateb et al. 2006] e Alemão [Hamp and Feldweg 1997] . Além destas, merece também destaque a Wordnet.br, desenvolvida para o Português do Brasil [da Silva 2006], que, em sua versão atual, conta com um total de 19.882 SynSets, bem como as relações de sinonímia e antonímia[3], estando as demais ainda em andamento (*cf.* [Scarton and Aluísio 2010]).

[1]Para o conjunto completo das relações consulte [Miller 1995].

[2]http://wordnet.princeton.edu

[3]Estas informações foram obtidas a partir da análise direta da Wordnet.

Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology, pages 153–157,
Fortaleza, CE, Brazil, October 21–23, 2013.

Atualmente, a base construída em Princeton é usada nas mais variadas áreas de pesquisa, indo desde sistemas para recomendação de notícias (*e.g.* [Capelle et al. 2012]) até o reconhecimento automático de inferência em textos (*e.g.* [Pakray et al. 2011]), passando pela classificação de documentos médicos de acordo com o público-alvo pretendido (*e.g.* [Lakiotaki et al. 2013]) e a construção de sistemas capazes de responder questões feitas em língua natural (*e.g.* [Hakimov et al. 2013]). Cada uma dessas interações, por sua vez, exige uma API especializada, que conecte a aplicação à Wordnet. Esta API deve então ser desenvolvida para cada Wordnet específica pois, do contrário, resta ao pesquisador fazer uma tradução da língua em que se trabalha para o Inglês, para assim usar as APIs disponíveis para a Wordnet de Princeton (*e.g.* [Pakray et al. 2011]).

Por tratar-se de um mapeamento a outra língua, no entanto, é possível que essa abordagem acabe por perder alguma informação durante a tradução. Dessa forma, o ideal seria que cada Wordnet possuísse sua própria API de acesso. O trabalho aqui descrito busca preencher essa lacuna para o Português do Brasil, ao apresentar JWN-BR – uma API Java para a Wordnet.br. Construída nos moldes da API JAWS[4] para a Wordnet de Princeton, a JWN-Br possui as vantagens de (i) fornecer acesso direto à Wordnet.br, estruturando seus dados na forma de objetos Java; (ii) abstrair a codificação interna usada pela Wordnet.br, tornando seu uso mais fácil ao programador; e (iii) possibilitar o incremento local, por meio da API, da base de dados original da Wordnet.br. Distribuída sob a GPL, a JWN-Br está disponível para a comunidade no endereço http://www.each.usp.br/norton/resdial/.

O restante desse trabalho é organizado como segue. Na Seção 2 é feito um levantamento das APIs atualmente disponíveis para a Wordnet. A Seção 3, por sua vez, apresenta a descrição da JWN-Br, apontando suas particularidades, enquanto que a Seção 4 descreve os métodos para gerenciamento da API, permitindo sua modificação. Por fim, na Seção 5 são apresentadas as conclusões desse trabalho.

2. Trabalhos Relacionados

Ainda que não possua uma API oficial, a Wordnet de Princeton conta, atualmente, com uma lista de 38 APIs[5] para 22 linguagens diferentes, de R a Java. Outras Wordnets, por outro lado, são bem mais modestas, contando com um número bastante reduzido de APIs, como é o caso da GermaNet[6], com duas APIs (para Java e Perl), e a Wordnet para Hindi[7], com apenas uma API. As demais Wordnets parecem não fornecer API alguma a partir de seus sítios oficiais na *web*, como é o caso com Português Europeu, Holandês, Italiano, Japonês, Polonês, Vietnamita, Tailandês, Árabe, Assamês e Urdu, além, naturalmente, do Português do Brasil[8].

[4] http://lyle.smu.edu/~tspell/jaws

[5] https://wordnet.princeton.edu/wordnet/related-projects/#local

[6] http://www.sfs.uni-tuebingen.de/lsd/tools.shtml

[7] http://www.cfilt.iitb.ac.in/wordnet/webhwn/API_downloaderInfo.php

[8] Respectivamente, http://www.clul.ul.pt/clg/wordnetpt/index.html, http://www2.let.vu.nl/oz/cltl/cornetto/index.html, http://catalog.elra.info/product_info.php?products_id=1110, http://nlpwww.nict.go.jp/wn-ja/index.en.html, http://www.ltc.amu.edu.pl/polnet/, http://vi.asianwordnet.org, http://www.tcllab.org/~wn-th/#download, http://www.globalwordnet.org/AWN, http://www.tezu.ernet.in/~nlp/wordnet.php, http://www.cle.net.pk/urduwordnet/, e https://github.com/WordNet-Br.

Esses números, por sua vez, só vêm demonstrar a importância de se haver uma API dedicada exclusivamente à Wordnet.Br, facilitando assim a construção de sistemas que possam fazer uso desse poderoso recurso. Dessa forma, dentre as APIs presentes para a Wordnet de Princeton, e que foram escritas em Java, optamos por seguir o exemplo da Java API for WordNet Searching (JAWS) quando da construção da JWN-Br, devido à sua simplicidade de uso e configuração, além do fato desta ilustrar de forma mais clara a estrutura da Wordnet.

3. Descrição da API

A Wordnet.Br consiste de um arquivo contendo, em cada linha, as informações necessárias para a construção de um Synset. Estas, por sua vez, incluem a palavra principal do Synset (*headword*) e seu número de identificação, bem como sua função sintática [da Silva 2006]. Em seguida, há um conjunto de sentidos para a palavra que representa o Synset. Para cada sentido apresentado, há tanto uma lista de Synsets sinônimos quanto o Synset antônimo ao analisado.

A função básica da API é fornecer um objeto que implemente a interface Synset, responsável por fornecer métodos para acesso a cada um de seus elementos (a Figura 1 ilustra o diagrama de classes presentes na JWN-Br). Além disso, a API respeita a regra de que um mesmo termo pode estar em dois conjuntos de sinônimos distintos (ou seja, ter mais de um significado). Assim, quando se procura por um termo (com *getSynsets(String s)*, em *DataBase*), a API retorna uma lista dos Synsets que contêm este termo, ou null caso não encontre nenhum. Outras funções de *DataBase* são:

- *exists(String s)*: retorna *true* caso o termo s seja termo de algum Synset;
- *getSynsetByReference(int reference)*: retorna o Synset cujo identificador seja igual ao inteiro passado por parâmetro;
- *dataBaseSize()*: retorna o número de Synsets na Wordnet;
- *getSynsets()*: retorna uma lista com todos os Synsets da Wordnet.Br.

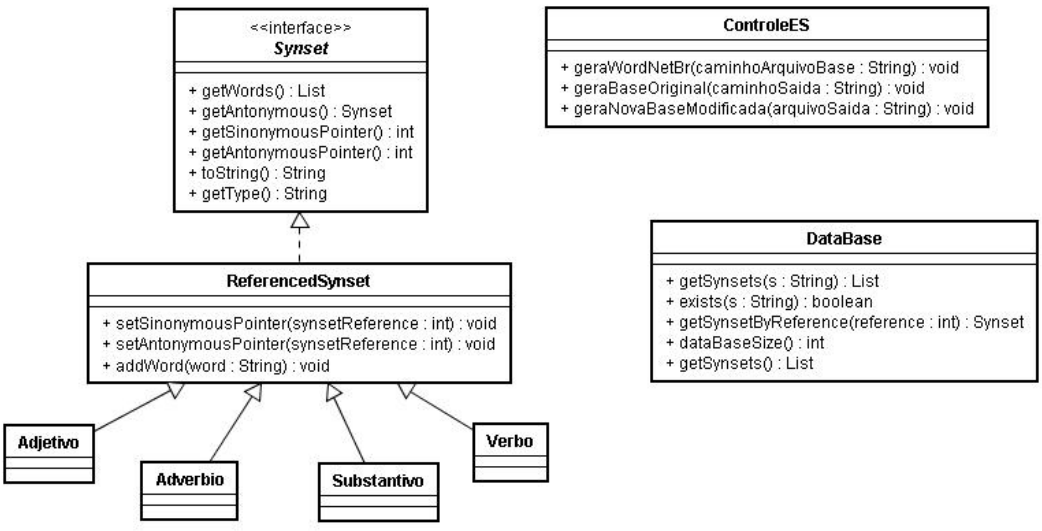

Figura 1. Diagrama de classes da JWN-Br.

O Synset antônimo de um determinado Synset, por sua vez, pode ser recuperado usando-se *getAntonimous()*, definido em *Synset*. Vale notar que, dentro da Wordnet, termos não possuem antônimos, somente Synsets. Isso porque um termo pode ter vários significados e, portanto, constar de vários Synsets. Assim, o antônimo de um termos dependeria muito de seu significado, capturado por seu Synset. Outras funções presentes em *Synset* são:

- *getWords()*, que retorna uma lista dos termos que constituem o Synset;
- *getSinonymousPointer()*, que retorna o identificador para o Synset em uso;
- *getAntonymousPointer()*, que retorna o identificador do Synset antônimo ao Synset em uso;
- *toString()*, correspondendo à representação String do Synset, segundo a estrutura "número_do_Synset tipo_do_Synset termo_1 termo_2 ... termo_n"; e
- *getType()*, que identifica o tipo sintático (String) do Synset[9].

4. Gerenciando a API

Além das funções para uso direto da API, descritas na Seção 3, há também um conjunto de funções destinadas ao seu gerenciamento. Conforme mencionado, devido ao fato da Wordnet.br tratar-se, de fato, de um arquivo texto, a primeira ação a ser tomada pelo programador é carregar o arquivo na memória, com o uso dos métodos *geraBaseOriginal(String caminhoSaida)* e *geraWordNetBr(String caminhoArquivoBase)*, de *ControleES*. Nesse caso, o primeiro lê o arquivo original da Wordnet, transformando-o em um arquivo intermediário, a ser lido pelo segundo método.

Nesse momento, é criado um objeto *DataBase* (presente na classe *ControleES*), dando ao programador acesso aos objetos *Synset* da base. A interface *Synset*, por sua vez, implementada por *ReferencedSynset*, apresenta métodos para modificação de um Synset específico. São estes: *setSinonymousPointer(int SynsetReference)*, que permite a atribuição de um identificador ao Synset (modificando o atual ou então atribuindo um novo, caso o Synset tenha sido criado pelo programador); *setAntonymousPointer(int SynsetReference)*, que permite a atribuição de um identificador ao Synset antônimo àquele sob consideração; e *addWord(String word)*, que permite a adição de um novo termo ao Synset.

Por fim, tendo modificado a base de dados original da Wordnet, o usuário pode usar o método *geraNovaBaseModificada(String arquivoSaida)*, de *ControleES*, para armazenar a nova base em um arquivo separado, caso deseje, podendo então trabalhar tanto com objetos da Wordnet original quanto da modificada por ele.

5. Conclusão

Nesse artigo apresentamos a JWN-Br – uma API java para a Wordnet.br. Única do gênero, até onde pudemos averiguar, a JWN-Br tem como principais características: (i) o oferecimento de acesso direto, para programas em Java, à Wordnet.br; (ii) a abstração de sua codificação interna; e (iii) a possibilidade de incremento, via API, da base de dados da Wordnet.br. Distribuída sob a GPL, a JWN-Br está disponível para a comunidade no endereço http://www.each.usp.br/norton/resdial/.

[9] Vale lembrar que todos os termos de um Synset são do mesmo tipo.

Referências

Bond, F., Isahara, H., Uchimoto, K., Kuribayashi, T., and Kanzaki, K. (2010). Japanese wordnet 1.0. In *Proceedings of the 16th Annual Meeting of The Association for Natural Language Processing*, page A5–3, Tokyo, Japan.

Capelle, M., Frasincar, F., Moerland, M., and Hogenboom, F. (2012). Semantics-based news recommendation. In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics (WIMS '12)*, Craiova, Romania.

da Silva, B. C. D. (2006). Wordnet.br: An exercise of human language technology research. In *Proceedings of the Third International WordNet Conference (GWC 2006)*, pages 301–303, South Jeju Island, Korea.

Elkateb, S., Black, W., Rodríguez, H., Alkhalifa, M., abd Adam Pease, P. V., and Fellbaum, C. (2006). Building a wordnet for arabic. In *Proceedings of The fifth international conference on Language Resources and Evaluation (LREC 2006)*, pages 29–34, Genoa-Italy.

Hakimov, S., Tunc, H., Akimaliev, M., and Dogdu, E. (2013). Semantic question answering system over linked data using relational patterns. In *Proceedings of the Joint EDBT/ICDT 2013 Workshops*, pages 83–88, Genoa, Italy.

Hamp, B. and Feldweg, H. (1997). Germanet – a lexical-semantic net for german. In *Proceedings of ACL workshop Automatic Information Extraction and Building of Lexical Semantic Resources for NLP Applications*, Madrid, Spain.

Lakiotaki, K., Hliaoutakis, A., Koutsos, S., and Petrakis, E. G. (2013). Towards personalized medical document classification by leveraging umls semantic network. In *Proceedings of the Second International Conference on Health Information Science (HIS 2013)*, pages 93–104, London, UK.

Marrafa, P. (2002). The portuguese wordnet: General architecture and semantic internal relations. *DELTA*, 18:131–146. ISSN 0102-4450.

Miller, G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41.

Pakray, P., Neogi, S., Bandyopadhyay, S., and Gelbukh, A. (2011). A textual entailment system using web based machine translation system. In *Proceedings of the 9th NII Test Collection for Information Retrieval Workshop Meeting (NTCIR-9)*, Tokyo, Japan.

Scarton, C. E. and Aluísio, S. M. (2010). Análise da inteligibilidade de textos via ferramentas de processamento de língua natural: adaptando as métricas do coh-metrix para o português. *Linguamática*, 2(1):45–61.