Gramática formal y lingüística computacional: un panorama

Fernando Carranza
fernandocarranza86@gmail.com

V Jornadas de Jóvenes Lingüistas
Mesa Plenaria:
Lingüística computacional y PLN: ¿qué, cómo y para qué? Una introducción al uso de herramientas computacionales para el análisis de datos lingüísticos
12/11/2021

Lingüística computacional y PLN

Lingüística computacional y PLN: ¿qué, cómo y para qué?

- El término lingüística computacional, de acuerdo con Kay (2005), fue propuesto originalmente por David Hays a mediados de los sesenta en el marco del reporte ALPAC, para evaluar la utilidad de invertir en traducción a máquina (*Machine Translation*).
- Para ese informe se rechazó el término Procesamiento de Lenguaje Natural.
- Con el tiempo, el significado de ambos términos fue cambiando, en ocasiones asimilándose, en ocasiones disimilándose.

Lingüística computacional y PLN

Lingüística computacional y PLN: ¿qué, cómo y para qué?

Definiciones provisorias (y para nada exhaustivas):

- **Lingüística computacional**: Disciplina que desarrolla y estudia las propiedades de diversa clase de modelos formales de representación del conocimiento lingüístico; surge de la interacción de los desarrollos de Inteligencia Artificial y la Lingüística Formal; disciplina que utiliza la computación como herramienta para la indagación lingüística.
- **Procesamiento del lenguaje natural**: Disciplina que desarrolla algoritmos para resolver tareas que involucran lenguaje natural; "pata ingenieril" de la Lingüística computacional.

Lingüística computacional y PLN: ¿qué, cómo y para qué?

El para qué depende de la perspectiva y perfil de quien investiga:

- Lingüista
- Desarrollador

Gramática formal

Gramática formal: la tarea de todo gramático formal consiste en formular un modelo explícito del conocimiento que los hablantes tienen de su lengua, lo que se traduce en el algoritmo encargado de la generación e interpretación de todas las oraciones gramaticales de una lengua y de ninguna de las agramaticales.

Criterios de adecuación epistemológicos

Como disciplina científica, toda teoría lingüística que pretenda llevar adelante la construcción de una gramática formal debe medirse a partir de una serie de criterios de adecuación epistemológica.

Criterios de adecuación epistemológicos

En Carranza (2020) se recogen los siguientes:

- Criterio de adecuación descriptiva
- Criterio de adecuación explicativa
- Criterio de adecuación teórica: i) minimalismo metodológico; ii) unicidad de dominio; iii) homogeneidad; iv) modelización teórica vs. clasificación taxonómica
- Criterio de adecuación evolutiva (ver Fujita 2007)
- Criterio de plausibilidad psicológica (ver Schönefeld 2001)
- **Criterio de adecuación computacional**

Criterio de adecuación computacional

Criterio de adecuación computacional se escinde en dos perspectivas:

1. Métrica de desajuste de complejidad (Ristad 1986)
   - Límite inferior en la jerarquía de lenguajes formales
   - Límite superior en la jerarquía de lenguajes formales
2. Requisito de ser implementada computacionalmente

Métrica de desajuste de complejidad

La preocupación por el límite inferior en la jerarquía de lenguajes formales fue inaugurado por Chomsky (1957) en su argumentación en contra de las gramáticas markovianas como formalismo válido para modelizar el lenguaje natural. Posteriormente, Shieber (1985) demostró que las gramáticas independientes de contexto tampoco alcanzan para dar cuenta del lenguaje natural, y que hay que recurrir a un formalismo superior.

Métrica de desajuste de complejidad

El problema de cuál es el límite inferior en la jerarquía de lenguajes formales es crucial, puesto que va de la mano del mismísimo criterio de adecuación descriptiva.

Métrica de desajuste de complejidad

El problema del límite superior resulta más esquivo. ¿Es necesario que las teorías gramaticales atiendan este problema? Existen distintas respuestas a esta pregunta:

- No es necesario, existen otros problemas más importantes: gramática minimalista (Chomsky 1995 y trabajos posteriores bajo su influencia)
- El formalismo mismo tiene que estar restringido para no superar este límite: gramáticas categoriales (Bar-Hillel 1953, Moortgat 1988, Steedman y Baldridge 2011), Tree Adjoining Grammar (Kroch y Joshi 1985)
- La teoría, pero no el formalismo, tiene que estar restringido para no superar este límite: HPSG (ver Bender y Emerson En prensa)

Implementación computacional

Implementar computacionalmente en la práctica la gramática, por supuesto, no es un requisito para una teoría lingüística. Sin embargo, es una práctica que puede reportar grandes beneficios:

> *Una de las ventajas de programar fenómenos lingüísticos es que cualquier rasgo relevante de la organización lingüística debe hacerse explícito y entonces debe ser estudiado con vistas a mejorar tanto la arquitectura global como la formulación local de las reglas y principios involucrados. Por esta razón, estoy convencido de que la programación computacional de esta clase se convertirá (y, en realidad, continuará convirtiéndose) en una herramienta esencial para la lingüística teórica*
>
> *Dik 1992:30*

Implementación computacional

- Prueba de suficiente explicitud
- Depuración de errores
- Testeo de hipótesis lingüísticas

Implementación computacional

Algunas implementaciones computacionales de teorías lingüísticas disponibles:

- Gramáticas implementadas en ALE (ya no tienen soporte)
- Gramáticas minimalistas de Stabler (ver Stabler 2011, Collins y Stabler 2016)
- Pydelphin (ver Bender y Emerson en prensa)
- CCG en NLTK (ver Baldridge 2002)

Una alternativa

Una alternativa a la implementación computacional es el uso de apéndices que recojan el modelo propuesto (*e.g.* Gazdar *et al.* 1985, Pollard y Sag 1994, ver argumentación en Borsley y Börjars 2011 y Carranza 2020).

- El PLN involucra una gran variedad de tareas: reconocimiento de habla, síntesis de habla, etiquetamiento de clase de palabra, descomposición morfológica, análisis sintáctico (parcial o total), comprensión del lenguaje natural (*Natural Language Understanding* o NLU), extracción de entidades, extracción de hechos, generación del lenguaje natural, asistencia en redacción, traducción automática (*Machine Translation*), etc.
- Algunas tareas y/o algunos métodos específicos para resolver esas tareas involucran mayor conocimiento lingüístico que otras. Pero en todos los casos, es importante contar con conocimiento específicamente lingüístico. Por ejemplo, Bender y Koller (2020) observan que el malentendido de la distinción entre forma y significado llevan a muchos autores a creer que modelos entrenados a partir de grandes corpus "entienden" el lenguaje.

Lingüística computacional y PLN

Lingüística computacional y PLN: ¿qué, cómo y para qué?

Lingüística computacional y PLN

Lingüística computacional y PLN: ¿qué, cómo y para qué?

- Cómo prepararse
- Cómo insertarse

Cómo prepararse

Tentativamente, podemos establecer esta lista de recursos que alguien interesado en adentrarse en lingüística computacional debería aprender a manejar:

1. **Conocimientos teóricos**: lógica y matemática (funciones, teoría de conjuntos, lógica proposicional y lógica de predicados; ver Gamut 2002, 1991, Partee *et al.* 2012); rudimentos de teoría de los lenguajes (Hopcroft *et al.* 2006); Estadística y teoría de la probabilidad (Manning y Schutze 1999, Gries 2013)
2. **Conocimientos operativos**: Algún lenguaje de programación, preferentemente Python o R; escritura de expresiones regulares; manejo de repositorios (git); uso de consola y bash; uso de foros (*e.g.* stackoverflow)

Cómo insertarse

- Ver `https://barracudanlp.github.io/#`
- Prestar atención a las actividades de `https://gesel.github.io/`
- Grupo de Telegram (`t.me/difusion_nlp`)
- Animarse a presentarse a búsquedas laborales.

Bibliografía I

Baldridge, J. (2002). Lexically specified derivational control in combinatory categorial grammar.

Bar-Hillel, Y. (1953). A Quasi-Arithmetical Notation for Syntactic Description. *Language*, 29(1):47–58.

Bender, E. M. y Emerson, G. (En prensa). Computational linguistics and grammar engineering. En Müller, S., Abeillé, A., Borsley, R. D., y Koenig, J.-P., editores, *Head-Driven Phrase Structure Grammar: The handbook*.

Bender, E. M. y Koller, A. (2020). Climbing towards nlu: On meaning, form, and understanding in the age of data. En *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 5185–5198.

Borsley, R. y Börjars, K. (2011). Introduction. En *Non-Transformational Syntax*, pp. 1–8. Wiley Blackwell, Massachusetts.

Bibliografía II

Carranza, F. (2020). *El problema de las alternancias argumentales en la teoría lingüística: el caso de la diátesis pasiva*. Tesis doctoral, Universidad de Buenos Aires.

Chomsky, N. (1957). *Syntactic structures*. Mouton, The Hague.

Chomsky, N. (1995). *The minimalist program*. MIT Press, Cambridge, Massachusetts.

Collins, C. y Stabler, E. (2016). A formalization of minimalist syntax. *Syntax*, 19(1):43–78.

Dik, S. C. (1992). *Functional Grammar in Prolog: An Integrated Implementation for English, French, and Dutch*. Mouton de Gruyter, Berlin.

Fujita, K. (2007). Facing the logical problem of language evolution (I. jenkins, variation and universals in biolinguistics). *English Linguistics*, 24(1):78–108.

Bibliografía III

Gamut, L. T. F. ([1982] 2002). *Lógica, lenguaje y significado. Vol. 1: Introduccón a la lógica*. Eudeba, Buenos Aires.

Gamut, L. T. F. (1982/1991). *Lógica, Lenguaje y Significado. Vol. 2: Lógica Intensional y Gramática Lógica*. Eudeba, Buenos Aires. 2009.

Gazdar, G., Klein, E., Pullum, G., y Sag, I. (1985). *Generalized Phrase Structure Grammar*. Harvard University Press, Cambridge, Massachusetts.

Gries, S. T. (2013). Descriptive Statistics. En *Statistics for Linguistics with R: A Practical Introduction*, pp. 102–156. De Gruyter Mouton.

Hopcroft, J. E., Motwani, R., y Ullman, J. D. (2006). *Automata theory, languages, and computation*. Addison-Wesley, Boston, Massachusetts.

Kay, M. (2005). Acl lifetime achievement award: A life of language. *Computational Linguistics*, 31(4):425–438.

Bibliografía IV

Kroch, A. y Joshi, A. (1985). The linguistic relevance of Tree Adjoining Grammar. Manuscrito. University of Pennsylvania.

Manning, C. y Schutze, H. (1999). *Foundations of statistical natural language processing*. MIT press, Massachusetts.

Moortgat, M. (1988). *Categorial Investigations: logical and linguistics aspects of the Lambek calculus*. Foris, Dordrecht.

Partee, B., Meulen, A., y Wall, R. (2012). *Mathematical methods in linguistics*. Kluwer Academics, Dordrecht.

Pollard, C. y Sag, I. A. (1994). *Head-driven phrase structure grammar*. University of Chicago Press, Chicago, London.

Ristad, E. S. (1986). Computational complexity of current gpsg theory. En *Proceedings of the 24th annual meeting on Association for Computational Linguistics*, pp. 30–39. Association for Computational Linguistics.

Bibliografía V

Schönefeld, D. (2001). *Where Lexicon and Syntax meet*. Mouton de Gruyter, Berlin, New York.

Shieber, S. (1985). Evidence against the context-freeness of natural language. (8):333–343.

Stabler, E. P. (2011). Computational perspectives on minimalism. En Boeckx, C., editor, *Oxford handbook of linguistic minimalism*, pp. 617–643. Oxford.

Steedman, M. y Baldridge, J. (2011). Combinatory categorial grammar. En Borsley, R. y Börjars, K., editores, *Non-Transformational Syntax: Formal and Explicit Models of Grammar*, pp. 181–224. Wiley-Blackwell, Massacushetts.