Book Review

Natural Language Processing with Python

Steven Bird, Ewan Klein, and Edward Loper
(University of Melbourne, University of Edinburgh, and BBN Technologies)

Sebastopol, CA: O'Reilly Media, 2009, xx+482 pp; paperbound,
ISBN 978-0-596-51649-9, $44.99; on-line free of charge at `nltk.org/book`

Reviewed by
Michael Elhadad
Ben-Gurion University

This book comes with "batteries included" (a reference to the phrase often used to explain the popularity of the Python programming language). It is the companion book to an impressive open-source software library called the Natural Language Toolkit (NLTK), written in Python. NLTK combines language processing tools (tokenizers, stemmers, taggers, syntactic parsers, semantic analyzers) and standard data sets (corpora and tools to access the corpora in an efficient and uniform manner). Although the book builds on the NLTK library, it covers only a relatively small part of what can be done with it. The combination of the book with NLTK, a growing system of carefully designed, maintained, and documented code libraries, is an extraordinary resource that will dramatically influence the way computational linguistics is taught.

The book attempts to cater to a large audience: It is a textbook on computational linguistics for science and engineering students; it also serves as practical documentation for the NLTK library, and it finally attempts to provide an introduction to programming and algorithm design for humanities students. I have used the book and its earlier on-line versions to teach advanced undergraduate and graduate students in computer science in the past eight years.

The book adopts the following approach:

- It is first a practical approach to computational linguistics. It provides readers with practical skills to solve concrete tasks related to language.
- It is a hands-on programming text: The ultimate goal of the book is to empower students to write programs that manipulate textual data and perform empirical experiments on large corpora. Importantly, NLTK includes a large set of corpora—this is one of the most useful and game-changing contributions of the toolkit.
- It is principled: It exposes the theoretical underpinnings—both computational and linguistic—of the algorithms and techniques that are introduced.
- It attempts to strike a pragmatic balance between theory and applications. The goal is to introduce "just enough theory" to fit in a single semester course for advanced undergraduates, while still leaving room for practical programming and experimentation.
- It aims to make working with language pleasurable.

The book is *not* a reference to computational linguistics and it does *not* provide a comprehensive survey of the theory underlying computational linguistics. The niche for such a comprehensive review textbook in the field remains filled by Jurasky and Martin's *Speech and Language Processing* (2008). What the book does achieve very well is to bring the "fun" in building software tools to perform practical tasks and in exploring large textual corpora.

As a programming book describing practical state-of-the-art techniques, it belongs to the glorious family of Charniak et al.'s *Artificial Intelligence Programming* (1987), Pereira and Shieber's *Prolog and Natural Language Analysis* (1987), and Norvig's mind-expanding *Paradigms of Artificial Programming* (1992). It differs from these books in its scope (CL vs. AI) and the programming language used (Python vs. Lisp or Prolog). Another key difference is in its organization: Whereas the classical books have a strict distinction between chapters covering programming techniques and chapters introducing core algorithms or linguistic concepts, the authors here attempt to systematically blend, in each section, practical programming topics with linguistic and algorithmic topics. This mixed approach works well for me.

As the dates of these older classics indicate (they were published 20 to 25 years ago), this book is important in closing a gap. The transition of the field from a symbolic approach to data-driven/statistical methods in the mid 1990s has transformed what counts as basic education in computational linguistics. Correspondingly, textbooks expanded and introduced new material on probability, information theory, and machine learning. The trend started with Allen's (1995) textbook, which introduced a single chapter on statistical methods. Charniak (1993) and Manning and Schütze (1999) focused uniquely on statistical methods and provided thorough theoretical material—but there was no corresponding focus on programming techniques. Another impediment to teaching was the lack of easy access to large data sets (corpora and lexical resources). This made teaching statistical methods with hands-on exercises challenging. Combining statistical methods for low-level tasks with higher levels (semantic analysis, discourse analysis, pragmatics) within a one-semester course became an acrobatic exercise.

Although deciding on the proper proportion among mathematical foundations, linguistic concepts, low-level programming techniques, advanced algorithmic methods, and methodological principles remains challenging, this book definitely makes the life of computational linguistics students and teachers more comfortable. It is split into five sections: Chapters 1 to 4 are a hand-holding introduction to the scope of "language technologies" and Python programming. Chapters 5 to 7 cover low-level tasks (tagging, sequence labeling, information extraction) and introduce machine learning tools and methods (supervised learning, classifiers, evaluation metrics, error analysis). Chapters 8 and 9 cover parsing. Chapter 10 introduces Montague-like semantic analysis. Chapter 11 describes how to create and manage corpora—a nice addition that feels a bit out of place in the structure of the book. Each chapter ends with a list of 20 to 50 exercises—ranging from clarification questions to mini-programming projects.

The chapters all include a mix of code and concepts. Chapter 1 sets the tone. In a few pages, the reader is led into an interactive session in Python, exploring textual corpora, computing insightful statistics about various data sets, extracting collocations, computing a bigram model, and using it to generate random text. The presentation is fun, exciting, and immediately piques the interest of the reader.

Chapter 2 covers one of the most critical contributions of the book. It presents commonly used corpora packaged together with NLTK and Python code to read them. The corpora include the Gutenberg collection, the Brown corpus, a sample of the Penn Treebank, CoNLL shared task collections, SemCor, and lexical resources (WordNet and

Verbnet). The important factor is that these resources are made thoroughly accessible, easily downloaded, and easily queried and explored using an excellent Python programming interface. The NLTK Corpus Reader architecture is a brilliant piece of software that is well exploited in the rest of the book.

Chapter 3 introduces programming techniques to deal with text, Unicode, downloading documents from various sources (URLs, RSS feeds) and excellent practical coverage of regular expressions. It is typical of the book's approach that regular expressions are taught by example and through useful applications, and not through an introduction to automata theory. The chapter ends with an excellent introduction to more advanced topics in sentence and word segmentation, with examples from Chinese. Overall, this chapter is technical but extremely useful as a practical basis.

I find Chapter 4 problematic. It is a chapter fully focused on programming, which introduces some key techniques in Python (generators, higher-order functions) together with basic material (what a function is, parameter passing). In my experience teaching humanities students, the material is not sufficient for non-programmers to become sufficiently proficient and not focused enough to be useful for experienced programmers.

Chapters 5 to 7 introduce the data-driven methodology that has dominated the field in the past 15 years. Chapter 5 covers the task of part-of-speech tagging. The linguistic concepts are clearly explained, the importance of the annotation schema is well illustrated through examples (using a simplified 15-tag tagset and a complex one with 50 or more tags). The chapter incrementally introduces taggers using dictionaries, morphological cues, and contextual information. Students quickly grasp the data-driven methodology: training and testing data, baseline, backoff, cross-validation, error analysis, confusion matrix, precision, recall, evaluation metrics, perplexity. The concepts are introduced through concrete examples and help the student construct and improve a practical tool. Chapter 6 goes deeper into machine learning, with supervised classifiers. The Python code that accompanies this chapter (the classifier interface and feature extractors) is wonderful. The chapter covers a wide range of tasks where the classification method brings excellent results (it reviews POS tagging, document classification, sequence labeling using BIO-tags, and more). The theory behind classifiers is introduced lightly. I was impressed by the clarity of the explanations of the first mathematical concepts that appear in the book—the presentation of the concept of *entropy*, naive Bayes, and maximum entropy classifiers builds strong intuition about the methods. (Although the book does not cover them, NLTK includes excellent code for working with support vector machines and hidden Markov models.) Chapter 7 builds on the tools of the previous two chapters and develops competent chunkers and named-entity recognizers. For a graduate course, the theoretical foundations would be too superficial—and one would want to complement these chapters with theoretical foundations on information theory and statistics. (I find that a few chapters from *All of Statistics* [Wasserman 2010] and from *Probabilistic Graphical Models* [Koller and Friedman 2009] together with Chapter 6 of *Foundations of Statistical NLP* [Manning and Schütze 1999] on estimation methods are useful at this stage to consolidate the mathematical understanding.) Readers come out of this part of the book with an operational understanding of supervised statistical methods, and with a feeling of empowerment: They have built robust software tools, run them on the same data sets big kids use, and measured their accuracy.

The next two chapters (8 and 9) cover syntax and parsing. They start with CFGs and simple parsing algorithms (recursive descent and shift-reduce). CKY-type algorithms are also covered. A short section on dependency parsing appears (Section 8.5), but I found it too short to be useful. A very brief section is devoted to weighted CFGs.

Chapter 9 expands CFGs into feature structures and unification grammars. The authors take this opportunity to tackle more advanced syntax: inversion, unbounded dependency.

The material on parsing is good, but too short. In contrast to the section on tagging and chunking, the book does not conclude with a robust working parser. On the conceptual side, I would have liked to see a more in-depth chapter on syntax—a chapter similar in depth to Chapter 21 of *Paradigms of AI Programming* (Norvig 1992) or the legendary Appendix B of *Language as a Cognitive Process* (Winograd 1983). In my experience, students benefit from a description of clausal arguments, relative clauses, and complex nominal constructs before they can properly gauge the complexity of syntax. On the algorithmic side, there is no coverage of probabilistic CFGs. The material on PCFGs is mature enough, and there is even excellent code in NLTK to perform tree binarization (Chomsky normal form) and node annotation, which makes it possible to build a competent PCFG constituent-based parser. The connection between probabilistic independence and context-freeness is a wonderful story that is missed in the book. Finally, I believe more could have been done with dependency parsing: transition-based parsing with perceptron learning à la MaltParser (Nivre et al. 2007) is also mature enough to be taught and reconstructed in didactic code in an effective manner.

Chapter 10 is an introduction to computational semantics. It adopts the didactic approach of Blackburn and Bos (2005) and covers first-order logic, lambda calculus, Montague-like compositional analysis, and model-based inferencing. The chapter extends up to Discourse Representation Theory (DRT). As usual, the presentation is backed up by impressively readable code and concrete examples. This is a very dense chapter—with adequate theoretical material. It could have been connected to the material on parsing, by combining a robust parser with the semantic analysis machinery. This would have had the benefit of creating more cohesion and illustrating the benefits of syntactic analysis for higher-level tasks.

Chapter 11 is an interesting addition on managing and constructing corpora. The skills required for collecting and annotating textual material are complex, and the chapter is a unique and welcome extension to the traditional scope of CL textbooks.

Overall this book is an excellent practical introduction to modern computational linguistics. As a textbook for graduate courses, it should be complemented by theoretical material from other sources, but the introduction the authors give is never too simplistic. The authors provide remarkably clear explanations on complex topics, together with concrete applications.

The book builds on high-quality code and makes significant corpora accessible. Although I still use Lisp in class to present algorithms in the most concise manner, I am happy to see how effective Python turns out to be as the main tool to convey practical CL in an exciting, interactive, modern manner. Python is a good choice for this book: It is easy to learn, open-source, portable across platforms, interactive (the authors do a brilliant job of exploiting the exploratory style that only interpreters can provide in interspersing the book with short code snippets to make complex topics alive), and it supports Unicode, libraries for graph drawing and layout, and graphical user interfaces. This allows the authors to develop interactive visualization tools that vividly demonstrate the workings of complex algorithms. The authors exploit everything this software development platform has to deliver in an extremely convincing manner.

The decision of which material to include in the book is in general well founded. The authors manage to cover a range of issues from word segmentation, tagging, chunking, parsing, to semantic analysis, and even briefly reach the world of discourse. I look forward to an expanded edition of the book that would cover probabilistic parsing, text

generation, summarization, and lexical semantics. I would also have liked to see some coverage of unsupervised and semi-supervised learning methods.

For instructors, students, and researchers, this book, together with the excellent NLTK library, is an important milestone. No one should learn computational linguistics without it.

References

Allen, James, 1995. *Natural Language Understanding*. Benjamin/Cummings, Menlo Park, CA, 2nd edition edition.

Blackburn, Patrick, and Johan Bos. 2005. *Representation and Inference for Natural Language: A First Course in Computational Semantics*. CSLI Publications, Stanford, CA.

Charniak, Eugene. 1993. *Statistical Language Learning*. The MIT Press, Cambridge, MA.

Charniak, Eugene, Christopher K. Riesbeck, Drew V. McDermott, and James R. Meehan. 1987. *Artificial Intelligence Programming*. Lawrence Erlbaum Associates, Hillsdale, NJ, 2nd edition edition.

Jurafsky, Daniel and James H. Martin. 2008. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall, Upper Saddle River, NJ, 2nd edition edition.

Koller, Daphne and Nir Friedman. 2009. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press, Cambridge, MA.

Manning, Christopher D. and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, MA.

Nivre, Joakim, Johan Hall, Jens Nilsson, Atanas Chanev, Gülşen Eryiğit, Sandra Kübler, Svetoslav Marinov, and Erwin Marsi. 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95–135.

Norvig, Peter. 1992. *Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp*. Morgan Kaufmann, San Francisco, CA.

Pereira, Fernando C. and Stuart M. Shieber. 1987. *Prolog and Natural-Language Analysis*. CSLI Publications, Stanford, CA.

Wasserman, Larry. 2010. *All of Statistics: A Concise Course in Statistical Inference*. Springer, New York, NY.

Winograd, Terry. 1983. *Language as a Cognitive Process*. Addison-Wesley, Reading, MA.

Michael Elhadad is an associate professor at Ben-Gurion University, Israel. He has been teaching computational linguistics for 15 years. His research focuses on computational models of Modern Hebrew, text summarization, and text generation. His address is Department of Computer Science, Ben-Gurion University, Beer Sheva, 84105, Israel; e-mail: elhadad@cs.bgu.ac.il.