Chapter 2
Natural Language Processing Tools

Justin F. Brunelle
Old Dominion University, USA

Chutima Boonthum-Denecke
Hampton University, USA

ABSTRACT
This chapter discusses a subset of Natural Language Processing (NLP) tools available for researchers and enthusiasts of computer science, computational linguistics, and other fields that may utilize or benefit from Natural Language Processing. Several tools are described in this chapter, along with background, algorithms used in brief, usages, and examples. While this chapter is not comprehensive, it provides an extensive exposure to various NLP tools through examples, and it aims at providing an overview of the resources available, and concentrates mainly on open-source applications. Open-source applications were chosen since they are freely available for download by all users. Commonly, open source software provides the code that makes up the tool, and allows for users to inspect the inner-workings of the tools, or even modify them. By using open source examples, readers of this chapter can extend their investigation of NLP tools beyond the pages of this text by investigating the tools outlined.

INTRODUCTION
Communication is an essential to the existence of modern man. Computers are also an essential aspect of our current information age. As such, it is a natural progression from the development of the modern computer as a computational and modeling tool, to computer artificial intelligence, to the investigation of natural language processing. Natural Language Processing (NLP) comes from the concept of humans communicating with non-human devices. NLP is a technique through which a computer can close the gap between artificial intelligence and pure machine by processing human to human communication messages previously unrecognizable to machines. (Artificial
Natural Language Processing Tools

intelligence is a branch of computer science in which machines are programmed to mimic human learning and decision making. NLP aids artificial intelligence by allowing human languages to be processed by machines, and the machines to learn syntaxes and vocabulary of these languages.

Programmers have developed NLP tools to facilitate the parsing, analyzing, understanding, and generation of human languages by, or with assistance from, computers. These tools range from those that recognize and respond to human voice commands to those that create human-like speech and further to those that can parse and understand written text. This chapter will provide variety types of NLP tools along with background, algorithms, usages, and examples.

This chapter will cover NLP tools in the following categories: parser, understanding, and generation. Finally, speech recognition is briefly discussed.

Natural Language Parser

A natural language parser is a program that understands, or categorizes, the grammatical structure of sentences. Their development was one of the biggest breakthroughs in natural language processing and occurred during the 1990s (MacCartney, 2009). Most parsers have the ability to understand the structure of sentences or word phrases. Phrases are groups of words in a specific order. Sentence structure includes categorizing words as the subject or object of a verb, nouns and noun phrases, or conjunctions. The process of natural language parsing is analogous to sentence diagramming.

The term syntax refers to the ways in which words can fit together to form higher level structures that ultimately lead to higher meaning. These higher structures include, but are not limited to, phrases, clauses sentences and paragraphs. Thus, parsers have evolved to be syntactically driven; meaning they are designed to understand a higher level meaning of words based on their organization into these higher level structures in a sort-of bottom-up fashion. In a way, this is the opposite of pattern matching. With pattern matching, the interpretation of the input is done as a whole, whereas syntactic analyses from parsers are obtained by application of a grammar. Such a grammar determines what word structures are legal, as well as most valuable, in the language being parsed. Pattern matching searches for meaning, while parsers determine meaning through categorization and direct string equivalence comparisons.

Probabilistic parsers “learn” the syntactic knowledge of a language from pre-parsed sentences via machine learning principles. Machine learning is an aspect of computer science in which machines are “trained” using training data. In this case, the training data is the set of pre-parsed sentences. The computer will store the training data as a set of rules. The computer will apply these learned rules when performing its own parsing.

The parsers use acquired knowledge to try to produce the most likely analysis of new sentences (Toutanova and Manning, 2000, 2003). Parsers must be trained, or learn, from multiple examples of pre-parsed structures. Programmers often hard-coded recognizable patterns into a parser to ensure fidelity and reduce complexity. Complexity, in this context, refers to a combination of program runtime and the amount of memory required during code execution. More specifically, complexity refers to the number of pattern comparisons done during execution. In this section, Link Grammar Parser and Part-of-Speech Taggers are discussed.

Link Grammar Parser

The Link Grammar Parser from Carnegie Melon University (2000) is a syntactic parser that assigns to a sentence a syntactic structure that consists of a set of labeled links connecting pairs of words. Valid word use is represented by rules about how each word may be linked to others. A valid sentence is one in which all words are connected with valid connecting rules. Thus a parse is a solution to the problem of finding links to connect all the words