Using language models for generic entity extraction

Ian H. Witten, Zane Bray, Malika Mahoui, W.J. Teahan
Computer Science
University of Waikato
Hamilton, New Zealand
ihw@cs.waikato.ac.nz

Abstract

This paper describes the use of statistical language modeling techniques, such as are commonly used for text compression, to extract meaningful, low-level, information about the location of semantic tokens, or “entities,” in text. We begin by marking up several different token types in training documents—for example, people’s names, dates and time periods, phone numbers, and sums of money. We form a language model for each token type and examine how accurately it identifies new tokens. We then apply a search algorithm to insert token boundaries in a way that maximizes compression of the entire test document. The technique can be applied to hierarchically-defined tokens, leading to a kind of “soft parsing” that will, we believe, be able to identify structured items such as references and tables in html or plain text, based on nothing more than a few marked-up examples in training documents.

1. INTRODUCTION

Text mining is about looking for patterns in text, and may be defined as the process of analyzing text to extract information that is useful for particular purposes. Compared with the kind of data stored in databases, text is unstructured, amorphous, and difficult to deal with. Nevertheless, in modern Western culture, text is the most common vehicle for the formal exchange of information. The motivation for trying to extract information from it is compelling—even if success is only partial.

Text mining is possible because you do not have to understand text in order to extract useful information from it. Here are four examples. First, if only names could be identified, links could be inserted automatically to other places that mention the same name—links that are “dynamically evaluated” by calling upon a search engine to bind them at click time. Second, actions can be associated with different types of data, using either explicit programming or programming-by-demonstration techniques. A day/time specification appearing anywhere within one’s email could be associated with diary actions such as updating a personal organizer or creating an automatic reminder, and each mention of a day/time in the text could raise a popup menu of calendar-based actions. Third, text could be mined for data in tabular format, allowing databases to be created from formatted tables such as stock-market information on Web pages. Fourth, an agent could monitor incoming newswire stories for company names and collect documents that mention them—an automated press clipping service.

In all these examples, the key problem is to recognize different types of target fragments, which we will call tokens or “entities”. This is really a kind of language recognition problem: we have a text made up of different sublanguages (for personal names, company names, dates, table entries, and so on) and seek to determine which parts are expressed in which language.

The information extraction research community (of which we were, until recently, unaware) has studied these tasks and reported results at annual Message Understanding Conferences (MUC). For example, “named entities” are defined as proper names and quantities of interest, including personal, organization, and location names, as well as dates, times, percentages, and monetary amounts (Chinchor, 1999).

The standard approach to this problem is manual: tokenizers and grammars are hand-designed for the particular data being extracted. Looking at current commercial state-of-the-art text mining software, for example, IBM’s *Intelligent Miner for Text* (Tkach, 1997) uses specific recognition modules carefully programmed for the different data types, while Apple’s *data detectors* (Nardi *et al.*, 1998) uses language grammars. The *Text Tokenization Tool* of Grover *et al.* (1999) is another example, and a demonstration version is available on the Web. The challenge for machine learning is to use