Designing a System for Swedish Spoken Document Retrieval

Botond Pakucs$^{1,3}$ Björn Gambäck$^{1,2}$

1 Information and Language Engineering
2 Computational Linguistics
3 Centre for Speech Technology
Swedish Institute of Computer Science
University of Helsinki
Royal Institute of Technology
Box 1263
P.O. Box 4
SF-00014 Helsinki, Finland
S-100 44 Stockholm, Sweden

1 Introduction

Information retrieval from spoken documents is analogous to text-based retrieval; however, accessing audio documents causes some extra problems, in particular with respect to document segmentation, choice of indexing features, and robustness. In addition, retrieval of documents in Swedish, like most non-English languages, adds the extra dimension of morphology; also, when analysing spoken Swedish data, prosodic patterns have to be taken into account. In this paper we introduce SIREN, the Swedish Information Retrieval Engine, a very flexible, modular IR system which has been designed with a specific eye towards these issues.

1.2 Indexing Features

In text-based information retrieval methods, the most obvious and frequently used indexing features are the words. This approach is also the dominating one in spoken document retrieval systems. For small domains, it is possible to select a priori small, fixed sets of keywords for both recognition and indexing. This is, however, not feasible if large and diverse domains are covered. In such situations, a possible solution is to use a large-vocabulary word-based speech recognizer to convert the audio data to text and then filter the transcriptions through a language understanding system to reduce recognition errors, as in CMU’s Informedia project (Hauptmann, 1995).

2 System Design

Most spoken document retrieval systems are modifications of existing text-based IR-systems, disregarding the particular problems caused by the new media type. Instead of using existing platforms, we decided to design and implement our own IR-system, in order to manage the problems presented above. We formulated the requirements for a retrieval toolkit as follows:

- A modular design is desirable: making small changes to specific parts of the toolkit should be easy and fast.
- The system must be extensible, since we want to be able to experiment with new
algorithms and new media types.

- Running the toolkit has to be dynamic and flexible and allow for quick adaptations to new requirements at any time by connecting or disconnecting modules.

- Another desirable property is transparency: we might want to examine the data flow under certain conditions.

- Portability is not a major requirement, but a desirable software system property.

We chose the Java environment in order to address the portability issue. The object oriented design also allows us to reuse many of the modules in the query-formation process. By using the built-in Thread class features when implementing the search engine, we can in the future adapt the system to the use of parallel search threads for searching multiple index files. This is useful when several media types are searched at the same time.

The two major components of our IR-toolkit SIREN (the Swedish Information Retrieval Engine) are the indexing tool and the search engine. A third component, an evaluation tool, is planned. The information retrieval solutions and algorithms used are based on well-founded results. We use a common inverted-file structure for storing indexing and posting information. The index-term weighting scheme used is the common $tf \times idf$ formula. The search engine implements the vector-space model and the cosine metric matching function. Thus, the novelty of the approach lies not in the algorithms used, but in the way these algorithms are implemented in the system, and in the system design which allows for fast reimplementation and integration of new modules.

2.1 The modules

The main tools are built up of several small, independent modules, each of them taking care of one well-defined atomic task. The modules are implemented as independent Java classes and are connected together in a pipelined architecture, using the Java built-ins PipedReader and PipedWriter for input and output, respectively. In this way, pipe synchronization is taken care of automatically. The connections between the modules are dynamic, so that modules can be invoked “on the fly”. Modules can even behave in a mutually interchangeable way, if they perform similar tasks.

2.2 The token language

The modules are communicating through a common token stream which consists of simple ASCII text. The modules are operating on the stream by modifying the tokens, or by adding new or deleting old tokens. Each token holds one term and a number of additional attributes connected with it.

References

Falk, J. 1997. Pauses in Synthesized Speech: Automatic Prediction of Silent Intervals in Swedish. MA Thesis, Göteborg University, Dept. of Linguistics.

Hauptmann, A. G. 1995. Speech Recognition in the Informedia™ Digital Library: Uses and Limitations. *Proc. 7th Int. Conf. on Tools with AI*, Washington, DC.

Schäuble, P. and Wechsler, M. 1995. First Experiences with a System for Content Based Retrieval of Information from Speech Recordings. *Proc. IJCAI Workshop on Intelligent Multimedia Information Retrieval*. 