Automatic Extraction of Acronyms from Text

Stuart Yeates
Computer Science, Waikato University, Private Bag 3105, Hamilton, New Zealand

ABSTRACT
A brief introduction to acronyms is given and motivation for extracting them in a digital library environment is discussed. A technique for extracting acronyms is given with an analysis of the results. The technique is found to have a low number of false negatives and a high number of false positives.

Introduction

Digital library research seeks to build tools to enable access of content, while making as few as possible assumptions about the content, since assumptions limit the range of applicability of the tools. Generally, the broader the assumptions the more widely applicable the tools. For example, keyword based indexing [5] is based on communications theory and applies to all natural human textual languages (allowances for differences in character sets and similar localisation issues notwithstanding).

The algorithm described in this paper makes much stronger assumptions about the content. It assumes textual content that contains acronyms, an assumption which is known to hold for modern (post-1940s), western (English), technical and governmental text. These assumptions are limiting, but not necessarily crippling, since there are large, growing collections of such text in active use in libraries around the world which are in need of tool support.

This paper concentrates on the automatic detection and extraction of acronym-definition pairs, which allows the building of acronym lists and appears to be central to acronym based tool-building. Techniques are introduced and evaluated on material from the New Zealand Digital Library (http://www.nzdl.org/) [4]. The program that implements these techniques is called TLA (Three Letter Acronym), because initially many of the interesting acronyms were three letter acronyms, common in computer science literature.

Acronyms

Abbreviations are contractions of words or phrases which are used in place of their full versions, where their meaning is clear from the context in which they appear. Acronyms are a type of abbreviation made up of the initial letters or syllables of other words [2]. Key differences between acronyms and other abbreviations include the lack of symbols such as apostrophe (’) and fullstops (.) in acronyms, more standard construction and the use of capital letters. Can’t and etc. are abbreviations
but not acronyms, in the first case both because of the inclusion of other than initial letters and because of the inverted comma (’) and the in second case because of the use of a (.), both lack of capital letters.

Very commonly used and easily pronounced acronyms may enter the language as normal words, in which case they are capitalised using standard capitalisation rules. Laser (for Light Amplification by Stimulated Emition of Radiation) and radar (for RAdio Detection And Ranging) are such words.

Acronyms are a relatively new linguistic feature in the English language, first appearing in the 1940s and 1950s. Early acronyms include: AmVets for American Veterans’ Association (1947); MASH for Mobile Army Surgical Hospital (1954) and MASER for Microwave Amplification by Stimulated Emition of Radiation (1955) [2]. Historically, fertile sources of acronyms have been organisational (UNESCO, VP, CEO, etc.), military (MASH, SIGINT, NATO, etc.) and scientific (laser, ROM, GIS, etc.) jargons. This suggests that acronym-based tools are going to have little use with pre-1940s text such as the public domain Oxford Text Archive or Project Gutenberg, since most of these texts are (a) literary not technical or governmental and (b) mostly pre-1940s.

Acronyms may be nested, for example SIGINT (Signals Intelligence) is part of JASA (Joint Airborne SIGINT Architecture) or even recursive as in GNU (GNU’s not UNIX). All recursive acronyms appear to be constructed self-consciously and appear to be limited to a few domains. Although mutually recursive acronyms appear possible, they seem unlikely.

Acronym lists are avaliable from a number of sources,¹ but these are static—they list acronyms current in some domain at the time of compilation or officially in use in a domain or organisation. While these may be of use in specific organisational or domain they are unlikely to be useful for an arbitrary piece of text at some point in the future.

Abbreviations such as acronyms are used in places where either readers are familiar with the concepts and entities they stand for or their meanings are clear from the context of the discussion. Unlike other abbreviations, acronyms are usually introduced in a standard format when used for the first time in a text. This form is: ROM (Read Only Memory) or alternatively Read Only Memory (ROM), the latter being preferred when readers are unlikely to be familiar with the concept. Later instances of the acronym can be given simply as ROM. In texts only mentioning an concept tangentially only the definition may be given—Read Only Memory.

Acronyms are not necessarily unique. The “Acronym Finder” web site (http://www.mtnds.com/af/) has 13 definitions for CIA ranging from Central Intelligence Agency and Canadian Institute of Actuaries to Central Idiocy Agency and Chemiluminescence Immunoassay. In normal texts, non-uniqueness doesn’t pose a problem, as usually the meaning is clear from the context of the document. Uniqueness is likely to be an issue if acronyms are extracted from large, broad-

¹See for example http://www.geocities.com/~mlshams/acronym/acr.htm or http://www.ucc.ie/cgi-bin/acronym
Uses for Acronyms in Digital Libraries

There are three main reasons for detecting acronyms in a digital library context.

Helping existing tools work more smoothly. Many tools utilise keyword and similar techniques, however, they are hindered by concepts appearing under different guises (i.e. ROM being called Read Only Memory). Once the majority of acronyms have been recognised it should be possible to overcome these problems.

Building new tools based on gathered acronym data. Thesauruses, search-by-acronym indexes and similar tools could be built once acronyms are identified.

Annotation or decoration of text presented to the user. Conceivably once an acronym has been identified and its definition found, all instances of the acronym in the text (and potentially in other texts in the same domain) can be annotated with the definition (or a link to the definition) of the acronym. The decoration can be explanatory (to explain the meaning of the acronym) or navigational (to help the user navigate to other documents which contain the acronym or from which the acronym was extracted).

The context in which acronyms appear may pose a problem in a broad based digital library, particularly for acronyms which have different definitions in different contexts (see the CIA example above) or which are identical to a normal word—a RAM can be Random Access Memory, or a male sheep!

It may be possible to infer sufficient context from the text itself to determine which of several definitions are likely to be correct, but this possibility is not explored here.

There are three structural possibilities for an acronym definition tool within a digital library, as shown in Figure 1. The first (A) shows acronyms being extracted from a single document, and then the definitions being used to decorate that document. The second (B) shows acronyms being extracted from many documents and collected for later external use. The third (C) shows acronyms being extracted from many documents and collected for use decorating documents (presumably documents from the group that the acronyms have been extracted from).

Taghva [3] gives the following definitions for measuring acronym extraction:

\[
\text{recall} = \frac{\text{# of correct acronym definitions found}}{\text{total # of acronym definitions in document}}
\]

\[
\text{precision} = \frac{\text{# of correct acronym definitions found}}{\text{total # of acronym definitions found}}
\]
Figure 1: Three structural possibilities for an acronym definition tool within a digital library (a) use of acronyms to decorate the same document (b) compilation of acronyms for external use (c) decoration of documents from compiled acronyms
Extraction Techniques

Figure 2 shows the overall design of TLA, our acronym extractor. The lexical analyser takes a stream of raw text from which it selects candidate acronyms and their definitions. These are then fed to a heuristic checker which applies a number of rules in order to discard false matches in the lexical analyser. The resulting acronyms are then smoothed (sorted and duplicates removed) by a stage known as the smoother.

The primary motive from separating the lexical analyser from the heuristic checker is to allow the heuristics to be changed quickly and easily without changing the basic detector.

The lexical analyser performs two tasks. The first is to remove all non-alphabetic characters and break the text into chunks based on the occurrence of (, ) and . characters, all of which indicate the end of one chunk and the start of another. For example the text:

\[ \text{Ab cde (fgh ijk) lmn o p. Qrs} \]

is broken into:

\[ \text{Ab cde | fgh ijk | lmn o p | Qrs} \]

Each word in each chunk is then considered to determine whether it is a candidate acronym. It is compared with preceding chunk and following chunk, looking for a matching definition. Thus the following pairs are checked:

\[
\begin{align*}
\text{Ab} & \quad \text{fgh ijk} \\
\text{cde} & \quad \text{fgh ijk} \\
\text{fgh} & \quad \text{Ab cde} \\
\text{ijk} & \quad \text{Ab cde} \\
\text{fgh} & \quad \text{lmn o p} \\
\text{ijk} & \quad \text{lmn o p} \\
\ldots & \quad \ldots
\end{align*}
\]

If a definition is found that is a potential match, the acronym-definition pair becomes a candidate acronym, and is passed to the heuristic checker. The lexical analyser uses the algorithm outlined in figure 3 when looking for definitions. It grossly under-estimates the requirements for an acronym-definition pair, meaning that \( \geq 95\% \) of the candidate acronyms are rejected by the heuristic checker.
Heuristics

Once candidate acronyms have been found they are passed through a number of heuristics, any one of which may fail the acronym. The heuristics are loosely based on the definition of acronyms given above. Specifically:

- Acronyms are shorter than their definitions
- Acronyms contain initials of most of the words in their definitions
- Acronyms are given in upper case
- Shorter acronyms tend to have longer words in their definition
- Longer acronyms tend to have more stop words

The order and strictness with which these heuristics are applied is a matter of tuning, which is context dependant. In situations where acronyms are being gathered from thousands of documents (i.e. the first two given uses of acronyms), even a low error rate (for example one per document) or high precision, is going to lead to thousands of errors in the acronym dictionary. Applications from the third group (decorating documents with acronyms from with the document) are not going to accumulate errors in the same manner, so precision is not so crucial.

Choosing which heuristics to enforce and the level of strictness appears to be non-trivial, and may vary from domain to domain, organisation to organisation with acronymising habits. This may be a suitable use for Machine Learning (ML) techniques [1]. Given a table of real-life acronyms and the results of the acronyms on them, it should be possible to tune for a given recall or precision.

Stop words are common words such as the, of or a which are deliberately overlooked in text processing.
Previous Work

The only significant previous work in this area known is Taghva [3], which describes a method of extracting algorithms in an OCR (Optical Character Recognition) environment called AFP (Acronym Finding Program).

There are five main differences between the work by Taghva and that described here. These reflect the different approaches to the topic:

1. AFP only considers the first letter of each word when searching for acronyms. Long acronyms containing characters other than the first letter are sometimes matched successfully, but the matches are ‘errors.’ TLA considers the first three letters in each word. This enables TLA to match acronyms such as AmVets.

2. AFP uses probabilistic techniques to determine matches. A probability is computed that a given definition matches a candidate acronym and if this probability is higher than a certain threshold, then it is accepted. TLA uses a set of heuristics each of which can reject any candidate acronym in a boolean fashion.

3. AFP candidate acronyms are all upper case; upper case sentences are ignored. TLA is independent of case (but heuristics have case information available to them).

4. AFP parses words with embedded punctuation as single words, whereas TLA parses them as separate words. This allows matching of U.S. Geographic Service (USGS), but may prevent matching of other acronyms.

5. AFP accepts some errors. This enables matches for acronyms which TLA misses. For example DBMS (DataBase Management System), which TLA misses because the “B” is the middle of the “DataBase”.

The first three differences appear to indicate that TLA is more general than AFP, that is they increase the number of acronyms that fall within TLA’s definition of an acronym compared to that of AFP. It is unclear whether the fourth difference is likely to increase or decrease the generality of TLA. The fifth difference indicates the AFP is more general than TLA.

AFP reports precision and recall rates as high as 98%, this is far higher that TLA has so far achieved. In part the difference in recall and precision can be attributed to the increased generality. It may also be party due to the fact that TLA is still in development.

Discussion

The best results achieved by TLA (for a sample of ten computer science technical reports) were recall of $\frac{22}{52} = 91\%$ and precision of $\frac{32}{52} = 68\%$. AFP reported results as recall of 93% and precision of 98%, clearly better.
There are however three planned improvements for TLA which should increase its accuracy. The first is selection of only a single acronym definition for each acronym. If only the longest definition for each acronym is accepted from the 10 sample documents, this increases precision to $\frac{22}{25} = 88\%$ without effecting recall, a clear improvement.

The second improvement would be to use ML to select which combination of heuristics best match the acronyms in a collection of text documents and the desired recall/precision balance. This would enable automatic or semi-automatic tuning for different types of collections and different uses.

The third improvement is the requirement that acronyms appear in multiple documents before they are accepted. This prevents acronyms which appear only in a single document (such as ephemeral names of proof-of-concept implementations) leaking into global glossaries and similar entities. There is also the related issue of acronyms that are defined multiple times within a document. This is generally not considered good writing style and multiple definitions of a candidate acronym may indicate that the candidate is not an acronym.

As mentioned above, any thorough comparison of AFP and TLA requires that they be run on the same set of documents. The trial documents used by AFP were markedly different, with an acronym definition density of $\frac{463}{17} = 27$ per document, compared to that of $\frac{22}{10} = 2$ for the TLA documents. This is possibly due to the fact that the AFP documents are governmental documents, whereas the TLA documents are computer science technical reports.

References


