PROBLEMS OF INDEXING CLASSES OF NEWS BASED ON THE COMPUTED IMPORTANCE OF WORDS

Echezona, S.C.
Department of Computer Science, University of Nigeria, Nsukka.
E-mail: Echez2003@yahoo.com

ABSTRACT
News is classified. Such classes as sports, politics, news on crime, gossips, business, etc., are common amongst newspapers in Nigeria. Interestingly most readers and patrons of newspapers adopt the rule of the thumb in choosing a suitable newspaper to read/buy. However, most newspapers try to cover all classes but end up in strikingly stress areas. This paper firstly explains the basic steps in generation of Document Indexes then secondly, highlights some of the problems associated with using full automation in classifications. Some of the problems identified are loss of relevance, loss of coverage and partial automation.

Keywords: Classification, Indexing, keywords

INTRODUCTION
Automatic keywords generation has been used in many areas of information indexing and classification, such as, articles, abstracts, captions and books. Every class of news goes with certain keywords that are unique for a given class. Such words as “Domination, National Assembly, Speaker, House of Assembly, Senate, etcetera”, suggest politics even when some can suggest other areas. The method adopted by this paper is to extract relevant titles that belong to various classes of news heuristically and processed them down to keywords (word stems) that will represent these classes using software. Thereafter a user can use any article/caption from any newspaper to match these word stems using the same method that will be discussed.

MATERIALS AND METHODS
To generate word stems for every class of news, newspaper captions/titles/headlines of Nigerian origin are used. Such articles/captions/headlines are heuristically identified, extracted and separated into documents then entered into software which will process them in the following sequence (Doyll and Blankenship; 2002):

- Tokenization
- Noise/common/stop words removal
- Reduction to word stems by removal of suffixes
- Weighting factor attached to the word stems
- Keywords extraction by choosing suitable threshold

Developing a user-friendly software to perform this task within seconds will definitely make things easier and more interesting for the end users. People from all walks of life read newspapers everyday; businessmen looking for business leads, politicians, researchers, students seeking information, etc. Many of these people have to read a large number of newspapers, page after page, perusing tons of seemingly useless information and actually missing the essential due to fatigue and boredom.

Being able to skim through scores of newspapers, reading only headlines and letting the software decide if the story could answer ones questions would exponentially improve the productivity of most newspaper readers. Relevant articles could be sorted out and later read in detail after all the sorting and categorization has been done.

The original ideas of Luhn on which most of automatic text analysis has been built goes on to describe a concrete way of generating document representatives through weighing or classifying keywords are discussed. Luhn’s earlier paper (Edmundson and Wylus, 2007) states: "It is here proposed that the frequency of word occurrence in an article furnishes a useful measurement of word significance. It is further proposed that the relative position within a sentence of words having given value of significance furnishes a useful measurement for determining the significance of sentences. The significant factor of a sentence will therefore be based on a combination of these two measurements." His assumption is that frequency data can be used to extract words and sentences to represent a document.

Let f be the frequency of occurrence of various word types in a given position of text and r their rank order, that is the order of their frequency of occurrence, then a plot relating f and r yields a curve similar to hyperbolic curve below. This is a curve demonstrating Zipf’s Law (as contained in Yu and Salton 2006) which states that the product of the frequency of use of words and the rank order is approximately constant. Zipf (Yu and Salton 2006) verified his law on American newspaper English.
Luhn used it as a null hypothesis to enable him specify two cut-offs, an upper and a lower, thus excluding non-significant words. The words exceeding the upper cut-off were considered to be common and those below the lower cut-off rare and therefore not contributing significantly to the content of the article.

He thus devised a counting technique for finding significant words, by which he meant the ability of words to discriminate content, reached a peak at a rank order position half way between the two cut-offs and from the peak fell off in either direction reducing to almost zero at the cut-off points.

Typical News Classes of Nigerian Newspapers are as follows:
- Sports
- Politics
- Religion
- Entertainment
- Information Technology
- Health
- Crime
- Environment
- Business
- Education
- Government
- Etc.

Some important algorithms developed for this study are as follows: (Echezona’s 2000)

1. Algorithm for Tokenization (close to Pascal language).

   Two recognizable functions are “Expunging from the text special characters” and “from left to right cut the remaining text to strings with blank as delimiter”. List of special characters are punctuation symbols, such as: ‘,’ , ‘;’ , ‘:’ , ‘?’ , ‘.’ , ‘”’ , ‘/’ , ‘(’ , ‘)’ , ‘_’ , etc.

   procedure expunge;
   var writt:string; ch:char;
   while not eoln(text) do
      begin
         read(text,ch);
         if not (ch in skipsett)
            begin
               write(chara, ch)
            end;
      end;

   procedure word;
   var t,v:string;

   begin
while not eoln(comp) do
begin
read(comp,t);
if (t<>' ') then
v:=Copy(t,1,1);
strg:=concat(strg,v)
end
end;

2. Removal of some common words. Some common words encountered are tabulated below: (Chang, 2007)

<table>
<thead>
<tr>
<th>Word</th>
<th>IF</th>
<th>REAL</th>
<th>INTO</th>
</tr>
</thead>
<tbody>
<tr>
<td>STAND</td>
<td>THAT</td>
<td>CONSPICUOUS</td>
<td>A</td>
</tr>
<tr>
<td>RESPIRE</td>
<td>THEN</td>
<td>THE</td>
<td>TAKE</td>
</tr>
<tr>
<td>RESPECTIVE</td>
<td>SPITE</td>
<td>THEIR</td>
<td>OFF</td>
</tr>
<tr>
<td>INSPIRE</td>
<td>THUS</td>
<td>NUMBER</td>
<td>OF</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>ROLL</td>
<td>MAKE</td>
<td>FOR</td>
</tr>
<tr>
<td>AT</td>
<td>IN</td>
<td>IT</td>
<td>REGRET</td>
</tr>
<tr>
<td>TO</td>
<td>RELAX</td>
<td>TICK</td>
<td>WHO</td>
</tr>
<tr>
<td>PULL</td>
<td>PASS</td>
<td>AS</td>
<td>DOWN</td>
</tr>
<tr>
<td>MADE</td>
<td>RAISE</td>
<td>RETURN</td>
<td>HIS</td>
</tr>
<tr>
<td>REQUIRE</td>
<td>WITH</td>
<td>WITHIN</td>
<td>HARD</td>
</tr>
<tr>
<td>WITHOUT</td>
<td>RUSH</td>
<td>OUT</td>
<td>HELP</td>
</tr>
<tr>
<td>LIFT</td>
<td>ESTABLISH</td>
<td>ALL</td>
<td>PART</td>
</tr>
<tr>
<td>DUE</td>
<td>DAY</td>
<td>DEPLORE</td>
<td>NIGHT</td>
</tr>
<tr>
<td>BRIEF</td>
<td>ANY</td>
<td>BEHIND</td>
<td>BUT</td>
</tr>
<tr>
<td>BECOME</td>
<td>AMONG</td>
<td>ALONE</td>
<td>AMONGST</td>
</tr>
<tr>
<td>CAN</td>
<td>ACROSS</td>
<td>ANYTHING</td>
<td>ANYWHERE</td>
</tr>
<tr>
<td>AFTERWARDS</td>
<td>BELOW</td>
<td>BEEN</td>
<td>ALREADY</td>
</tr>
<tr>
<td>TOUGH</td>
<td>GOES</td>
<td>SOFT</td>
<td>BEFORE</td>
</tr>
<tr>
<td>THOUGH</td>
<td>HOW</td>
<td>VIEW</td>
<td>BESIDE</td>
</tr>
<tr>
<td>THOROUGH</td>
<td>CALL</td>
<td>FAIL</td>
<td>AGAIN</td>
</tr>
<tr>
<td>ALTHOUGH</td>
<td>UP</td>
<td>AROUND</td>
<td>IS</td>
</tr>
<tr>
<td>GO</td>
<td>HER</td>
<td>BOTH</td>
<td>ETC.</td>
</tr>
</tbody>
</table>

An algorithm for noise words extractions is as follows:

Procedure noisewords(text, arr:noisewd; n:integer);
var wd: string;
begin
while not eof() do
begin
readln(text wd);
for j:=1 to n do
begin
if strcomp(wd, arr[j]) then
writeln(text," ");
else
write{text,wd};
end
end;
end;

3. Suffix removal algorithm uses common suffices available to compare with the end of each recognized term and chops off the part that match. Simple algorithm is given below. Other checks like morphological transformations existing in English language which may alter the stem of suffixed words; for example the word "absorb" is transformed into "absorption" when the suffix "tion" is added. Similarly "hop" is transformed to "hopping", "relief" becomes "relieving" and so on. Transformational rules can be set up (outside the algorithm below) in order to recode various automatic generated stems following suffix removal. A typical rule might state "remove one of the possible occurrences of b, d, g, l, m, n, p, r, s, t, from the end of the generated term". These rule's algorithm is not included.

Procedure remsuf;
var t,l:integer;
s,c,strr,suf:string;

begin
  reset(cf,'retainer'); \* New file of word stem. Reset positions the pointer to the first record*
  rewrite(cf,'wordfile'); \* Tokenized words requiring suffix removal*
  reset(suff,'suffix.text'); \* File of suffixes *
  while not eof(cf) do
    begin
      readln(cf,strr);
      while not eof(suff) do
        begin
          read(suff,suf); l:=0; t:=length(suf);
          for j:=1 to length(suf) do
            begin
              s:=copy(suf,t-j+1,1);
              c:=copy(strr,length(strr)-j+1,1);
              if(s=c) then l:=l+1 end;
            end;
          if(l = t) then begin chara:=copy(strr,1,length(chara)-t);
            writeln(cff,chara) end;
        end;
    rewrite(cf,'retainer');
  end;
  rewrite(cf,'retainer');
  while not(eof(cf) and eof(cff)) do
    begin
      readln(cff,chara);
      writeln(cf,chara);
    end;
end;

Some suffices in English are tabulated below: (Chang, 200]

<table>
<thead>
<tr>
<th>S</th>
<th>ISM</th>
<th>AL</th>
<th>LY</th>
<th>LLY</th>
</tr>
</thead>
<tbody>
<tr>
<td>IVENESS</td>
<td>IVE</td>
<td>NESS</td>
<td>D</td>
<td>MENT</td>
</tr>
<tr>
<td>OUS</td>
<td>CEOUS</td>
<td>ACEOUS</td>
<td>IES</td>
<td>ALIC</td>
</tr>
<tr>
<td>ER</td>
<td>UOUS</td>
<td>ABILITIES</td>
<td>ACIDEOUS</td>
<td>AIC</td>
</tr>
<tr>
<td>ABILITY</td>
<td>ACIDEOSLY</td>
<td>AICAL</td>
<td>ACIES</td>
<td>ABLE</td>
</tr>
<tr>
<td>AICALLY</td>
<td>ACEOUSNESS</td>
<td>ABLED</td>
<td>AICISM</td>
<td>AICS</td>
</tr>
<tr>
<td>ACTIES</td>
<td>ABLENES</td>
<td>ACITIES</td>
<td>AICISMS</td>
<td>AL</td>
</tr>
<tr>
<td>ALISATION</td>
<td>ABLINFUL</td>
<td>ABLER</td>
<td>ABLING</td>
<td>AE</td>
</tr>
<tr>
<td>ALISATIONALLY</td>
<td>ALISATIONAL</td>
<td>ABLY</td>
<td>AGER</td>
<td>ACY</td>
</tr>
<tr>
<td>ACEOUSLY</td>
<td>ALISEDLY</td>
<td>ACITY</td>
<td>ACISE</td>
<td>AGE</td>
</tr>
<tr>
<td>ACEOUSNESSES</td>
<td>AGINGFUL</td>
<td>ALISER</td>
<td>ALISED</td>
<td>AGES</td>
</tr>
<tr>
<td>AGED</td>
<td>AGING</td>
<td>ALISER</td>
<td>ALISED</td>
<td>AGES</td>
</tr>
</tbody>
</table>

4. Finally, weighting by calculating frequencies within and outside documents and applying selected weighting formula, the final indexes results which can now represent content of the body of classes. Typical example includes: To find each term’s weight using Inverse Document Frequency Weight – INVDFWT.

\[ \text{WEIGHT}_i = \text{FREQ}_k \times (\ln(n) - \ln(\text{DOCFREQ}_k) + 1) \]

Where \( \text{FREQ}_k \) is the frequency of the term \( k \) in document \( i \), \( n \) is the number of documents, and, \( \text{DOCFREQ}_k \) is the number of documents the unique term appeared.

Associated Problems
This work was done at two periods covering a decade; the following problems were seen to be associated with the results, (that is, the index terms generated). These problems are:

- Loss of relevance
- Loss of coverage
- Partial automation

Loss of Relevance
This refers to the inability of index terms to still be relevant over time. The researcher has successfully carried out this indexing twice within a period of a decade, and has observed that most of the index terms generated tends to lose relevance with time.
This might be because no area is static. New syllables continue to be generated while others are dropped. This is most prominent when names of persons are used as part of the content identifiers. It is obvious that new entrants are made into the field of discuss, and obsolete ones are seldom referred to. For instance, if the name like “Patricia Etteh” that has been foremost in most newsprints in the recent past is used to represent part of the corpus of Nigerian politics will lose relevance in say next decade.

**Loss of Coverage**

Most classes of news command vast syllables. And so, to cover or index using abstracts/captions/titles/headlines extracted from say newspapers may not finally explore all possibilities. This may bias the result of the search and hence lead to loss of coverage. An attempt to cover all may mean expanding the coverage of the input data. Meaning; sampling more newspapers for over almost endless periods.

**Partial Automation**

Some steps of computation of index terms are painfully manual, not completely automatic. The

**REFERENCES**


