A SYSTEM FOR CREATING AND MANIPULATING GENERALIZED WORDCLASS TRANSITION MATRICES FROM LARGE LABELLED TEXT-CORPORA

Wilfried Bloemberg
Institute of Phonetics
University of Nijmegen
P.O. Box 9103
6500 HD Nijmegen
The Netherlands

Michael Kesselheim
Institut für Allgemeine Elektrotechnik und Akustik
Ruhr-Universität Bochum
Universitätsstrasse 150
D-4630 Bochum
West-Germany

ABSTRACT

This paper deals with the training phase of a Markov-type linguistic model that is based on transition probabilities between pairs and triplets of syntactic categories. To determine the optimal level of detail for a set of syntactic classes we developed a system that uses a set-theoretical formalism to define such sets and has some measures to compare and optimize them individually.

In section two we describe the optimization problem (in terms of prediction, information and economy requirements) and our approach to its solution. Section three introduces the system that will assist a linguist in handling the prediction and economy criteria and in the last section we present some sample results that can be achieved with it.

1. INTRODUCTION

The context in which we started developping the system described in this paper is the ESPRIT project #860, 'Linguistic Analysis of the European Languages', which deals with seven European languages.

The main objective of the project is to provide a language independent software environment for dealing with the linguistic phase of a number of applications in the realm of office automation such as high quality, natural sounding text-to-speech conversion for unlimited vocabularies, automatic speech recognition for large vocabularies, and omni-font optical character reading including automatic reading of handwriting.

The decision on what type of linguistic model to be used in the project was made at an early stage. It was decided to aim at a probabilistic positional grammar (a Markov-type grammar) based on transition probabilities of pairs and triplets of syntactic categories. The use of Markov-type models immediately incurs the necessity of defining training texts. We started out with training corpora of approximately 100,000 words of official EEC publications, that were available in all languages of the community. The training consists of building a number of data structures. The first is a lexicon of all words that occur in the text, with their attendant probability of occurrence and all possible wordclasses. The second structure is formed by two and three dimensional matrices describing the transition probabilities between pairs or triplets, respectively, of wordclasses. Clearly, the probabilities specified depend on the choice of syntactic categories along the dimensions. One of the major problems with a Markovian approach is to determine the optimal level of detail of the wordclasses for each dimension. In this paper we will describe a software system that helps linguists in carrying out experiments aimed at finding an 'optimal' system of wordclasses.

2. MARKOW ANALYSIS OF LARGE CORPORA AND WORDCLASS SYSTEMS

The problem of finding a suitable wordclass set for statistical disambiguation of syntactic labelling may be formulated more precisely and formally as follows:
Find a set of wordclass labels (with gross wordclass and complex information) that can label each word of a language and

1. is minimal in the number of labels (economy requirement)

2. provides high predictive power for adjacent wordclasses in a chain. A formal way to do this is by minimizing the average entropy of N-dimensional transition probabilities for subsequent labels in sentences, e.g. reduced to the two-dimensional case, to minimize:

$$E = -\sum_{j}\sum_{i} P(l_j \mid l_i)\log(P(l_j \mid l_i))/n$$

with:

S	summation symbol
n	number of labels in the system
i,j	indices running from 1 to n
P(a|b)	conditional probability of 'a' given 'b'

(prediction requirement)

3. is maximal in the amount of information about each labelled word, e.g. for syntactic analysis or disambiguation of alternative graphemic hypotheses. (information requirement)

To find an exact solution to this problem is difficult - if not impossible, because of

- the dimensionality of the optimization problem (given the large number of wordclasses needed to obtain useful parsing results)

- the difficulty to define a unique starting set of wordclasses for an optimization

- the dependence of a possible finite solution on the analysed corpus

Our approach to this problem is to start from a very detailed hierarchical wordclass system including complex information. The degree of detail can be reduced by means of the notion of "cover symbols" that form partitionings of the original system. Cover symbols and wordclasses not accounted for by cover symbols are called 'labels'. Initially, cover symbols will be created by combining wordclass symbols for related classes - e.g. the classes "verb, 1. person singular indicative present active" and "verb, 1. person singular conjunctive present active" giving a cover symbol "verb, 1. person singular present active". At a later stage other cover symbols can be created by combining and excluding wordclass symbols and already existing cover symbols. In the optimization process different sets of labels are created subsequently and compared by measures related to either of the criteria mentioned.

A user working in the optimization process needs measures to compare the significance of individual labels within a given set and to estimate the usefulness of joining labels into new, more comprehensive cover symbols. As one measure for criterium two we use the entropy directly in a global and diagnostic way. Additionally a number of measures have been defined that are related to entropy and give more specific information on the performance of individual labels.

Given a text in which to each word a label has been assigned that is:

1. the basic wordclass, if this has not been defined as belonging to a cover symbol

2. the applicable cover symbol otherwise

and given a 2D-matrix that contains relative frequencies of transitions from any label (wordclass or cover symbol) to any other label in the text, then some useful measures are

- the branching factor for a given label, that tells how many different labels actually followed/preceeded it in an analysed text.

- the variance of the transition probabilities in a row/column of the matrix, that indicates how much the strength of connections from the label to surrounding labels varies as analysed from a text.

- the correlation between different rows/columns of the matrix, that gives information about how similarly the labels behave in a general right/left context, i.e. how much information will be lost by combining two labels into a new cover symbol.

- the relative frequency of a given label, that indicates the relative labelling relevance within a given system.

The measures defined here for a 2D-matrix, can be applied to a 3D-matrix in a similar way, e.g. the correlation between two labels in the same matrix dimension then means correlating the numbers of two planes.

3. EMMA: AN EDITOR FOR MATRICES FROM MARKOV ANALYSIS

In order to assist linguists in their task of designing an optimal set of wordclasses we designed a tool called EMMA: Editor for Matrices from Markov Analysis. The most important design considerations for implementing the system are:

1. it must be a complete system: the user must be able to develop cover symbol sets, analyse matrices and transform them without leaving the system.

2. it must be easy to use for unexperienced users, therefore menu and windowing techniques have been applied. Also extended help is available at every point in the system.

3. it should be a fast tool for experienced users. They can create input command files by themselves or use the logging facility.

EMMA is split into two logical parts, though they are closely related. In the first part a user can create a set of cover symbols. A set-theoretical formalism has been defined for specifying cover symbols in a hierarchical way: recursively sets of labels may be put into lists, then such lists be excluded from other lists to specify the final set of wordclasses contained in a certain cover symbol. (see appendix for notation)

Such symbols can be defined for each dimension (called "scope") of a transition matrix separately, i.e. one can define a specific cover symbol only for e.g. the first position in a transition pair or triple. After a set of cover symbols has been defined a consistency check is made, to ensure that no wordclass symbol belongs to more than one cover symbol.

A set of cover symbol definitions is called a "mapping". A mapping has to be consistent but not necessarily complete, i.e. not every wordclass must belong to some cover symbol. Different sets of mappings can be merged together as long as they stay consistent.

In the second part of the system a user can create and manipulate transition probability matrices with the help of a mapping. Matrices can be created from labelled text: in this case the system will subsume wordclasses in their respective cover symbols and wordclasses not belonging to any cover symbol will extend the matrix. In this way the analysed text is not restricted with respect to the number of wordclasses. A second way to create matrices is from calculation on other matrices. Cover symbols can be defined interactively, and the new matrix belonging to the new mapping can be computed. To handle these matrices a data structure has been designed, based on the sparseness of the matrices. It fulfils two requirements: it is sufficiently fast for retrieval of data in an interactive environment and it can manipulate extremely large matrices (largest so far 750 x 750 x 750).

Different kinds of analyses and manipulations can be done on cover symbols and matrices in addition to the computation of the measures related to entropy. For such purposes the system includes a powerful mechanism to access matrices and related mappings for analysis and editing. One may take a number of labels from a dimension of a matrix, make them a set with a new name and define a submatrix by specifying such sets in the different dimensions. This submatrix may then be accessed selectively by display, statistics, change and quantization procedures.

In the statistics part information on sparseness and the highest and lowest transition probabilities in matrices or submatrices may be gathered. Correlations of transition frequencies between labels may be calculated for a certain numerical range of outcome only. List, change and quantization commands may be specified for a numerical range of frequencies in the submatrix. This ensures that one may access certain "frequency layers" in the matrix, which is an essential operation for viewing very large matrices with only a few percent of the entries non-zero.

If a user eventually finds that the labels in some dimension of a submatrix could be included into a new cover symbol, he/she may specify this directly and the overall matrix together with its mapping will be transfomed into a new smaller one. Different matrices may be merged as long as the related mappings are compatible in an analytic sense: cover symbols in one mapping must be either disjunct from the ones in the other mapping or in subset relation.

4. SOME EXAMPLE RESULTS

The partners within the consortium have just started the development of the optimal wordclass systems. Therefore, in this paper we will restrict ourselves to the presentation of a small number of examples that should convey the flavour of the kind of information that can be derived with the system.

The data in the examples are derived from an office text in German (80,000 words) and the same text in Dutch (100,000 words) labelled with the ESPRIT-wordclass system (ca. 250 wordclasses for German and 104 for Dutch were actually used). The symbols used in the examples can be interpreted as:

'P':	preposition,	'D':	determiner,
'N':	noun,	'A':	adjective,
'C':	conjunction,	'B':	adverb,
'M02':	date		

'#': the subclass cannot be specified for the wordclass in question

'.': the subclass is specifiable, but has not been specified

Example 1:

If a user works on a 3D-matrix with the matrix editor and considers inclusion of all conjunctions into one cover symbol in the first scope, but wants to leave the most frequent labels out, he/she will look e.g. at a part of the matrix by a command

```
DISPLAY C---------;;
```

which will give a display of only those parts of the matrix where a conjunction stands in the first position of the Markov chain.

Let us assume that the most frequent labels are C00#######, C02..##### and all labels C01 but without C01..#####, then he/she could define the cover symbol 'Z_CON' for scope 1 in the following way:

```
Z_CON =   C--------- ! _ZCEX;
_ZCEX     ( C00#######, C02#######,
          C01------- ! C01#####);
```

with: '()' the list operator
'!' the exception operator
'_ZCEX' a local name

With the help of this new cover symbol we can transform the matrix accordingly.

Example 2:

```
Listing of two most frequent wordclass
triples within German corpus
--------------------------------------

D00##N.F## A00.....## N00..S.F##    660
P00######## D00##N.F## N00..S.F##  1310
```

This is the well-known determiner-adjective-noun phrase and the preposition-determiner-noun phrase. The numbers indicate the frequency with which the triples occur in the training text.

Example 3: Statistics

```
Some symbols in first position of a chain
-----------------------------------------

symbol     scope relfreq  branching  stddev
                            factor

A17.....##   1  0.00006     0/1      0.030612
B09########  1  0.00399     0/28     0.238650
C00########  1  0.02771     0/105    1.298851
D01##S.M##   1  0.00260     0/17     0.348807
```

The very low standard deviation of the label A17.....## casts considerable doubt upon its significance; it will probably be included into a cover symbol. The label C00#######, on the other hand, will probably deserve to be given a class of its own.

Example 4:

```
Correlations between symbols in scope 1
---------------------------------------

V0001T..##      V0043T..##        0.000
V00.0...##      V29.0...##        0.838
M02########     B02########       0.908
```

The labels M02####### and B02####### have a high correlation and are therefore candidates to be put into the same cover symbol. But before doing this one has to determine the significance of such an operation by checking the standard deviation, branching factor and the relative frequency. Also the third criterium as defined in section two has to be taken into account.

Example 5:

```
Entropy of symbols in scope 1 derived
from the Dutch corpus
-------------------------------------

ZVERB                2.675
ZNOUN                2.371
ZADJEC               1.830
ZADVER               2.609
ZPRONO               1.799
ZPREP                1.870
ZCONJ                2.481
ZMISCE               2.564
```

This table has been derived from the Dutch corpus after definition of cover symbols for the main word classes. The entropies of these cover symbols are low compared to the maximum we encountered. Certainly this set of cover symbols is too small to fulfill the information requirement for e.g.

disambiguation of alternative graphemic forms.

APPENDIX: SYNTAX OF COVER SYMBOL SET DEFINITIONS

The grammar is in BN-form, where:

'{' means optionality,
'|' alternative,
'<' and '>' nonterminal,
informal descriptions are between double quotes.

<Definition> = <CS-notation> '=' < CS > ';' |
<CSA-notation>'=' < CS > ';'

<CS> = <Symbol list> {'!' <Symbol list>}
<Symbol list> = <Prim> | '(' <Primlist> ')'
<Primlist> = <Prim> | <Primlist> ',' <Primlist>
<Prim> = <CS> | <WCL-notation> |
<CSA-notation> |
<CS-constraint>

<CSA-notation> = '_'<CS_notation>
<CS-notation> = "valid cover symbol notation"
<WCL-notation> = "valid wordclass symbol notation"
<CS-constraint> = "constraint use of CS-notation"

with the following constraints:

- definitions are not allowed to be directly or indirectly recursive.

- cover symbols used in the map can only be excluded from other cover symbols (not included, otherwise the mapping would be inconsistent). This gives the constraint use of cover symbol notations within a cover symbol definition. E.g. in an expression Z1 = <exp1>!(<exp2>!<exp3>), the cover symbol set becomes inconsistent, if another cover symbol Z2 occurs included in <exp1> or <exp3>.

- cover symbols occurring on the right side of a definition must be defined in the same file.

In order to support order in the cover symbol definitions cover symbols that are to be included into other cover symbols (i.e. they have only auxiliary function, but will not occur in a map) are notated differently from cover symbols, that will occur in a map: Auxiliaries have a name preceeded by a '_'.

Additional notations are used in a textual definition to specify the scope for subsequently defined cover symbols.

Cover symbol definition files may include other cover symbol definition files by a C-like "#include" command.

INFORMATION FLOW IN THE EMMA MARKOW ANALYSIS SYSTEM