Linguistics as Data Mining: Dutch Diminutives

Walter Daelemans
Computational Linguistics,Tilburg University
PO Box 90153, 5000 LE Tilburg, The Netherlands
e-mail: Walter.Daelemans@kub.nl

Peter Berck and Steven Gillis
Center for Dutch Language and Speech, University of Antwerp
Universiteitsplein 1, 2610 Wilrijk, Belgium

Abstract

There are several different ways *data mining* (the automatic induction of knowledge from data) can be applied to the problem of natural language processing. In the past, data mining techniques have mainly been used in linguistic engineering applications to solve knowledge acquisition bottlenecks. In this paper, we show that they can also assist in linguistic theory formation by providing a new tool for the evaluation of linguistic hypotheses, for the extraction of rules from corpora, and for the discovery of useful linguistic categories. Applying Quinlan's C4.5 inductive machine learning method to a particular linguistic task (diminutive formation in Dutch) we show that data mining techniques can be used (i) to test linguistic hypotheses about this process, and (ii) to discover interesting linguistic rules and categories.

1 Introduction

The dominant view about the role of computers in linguistics has been that computer modeling is a useful (or essential) tool for enforcing internal consistency, completeness, and empirical validity of the linguistic theory being modeled. In this paper, we to argue that by using *Data Mining* techniques, the role of computation in linguistics can be significantly broadened.

Data Mining is a branch of computer science concerned with the automatic extraction from data of implicit and previously unknown information which is *nontrivial*, *understandable*, and *useful*, using techniques from Machine Learning and Statistical Pattern Recognition (clustering, rule induction, classification, etc.)(Piatetsky et al. 1991).

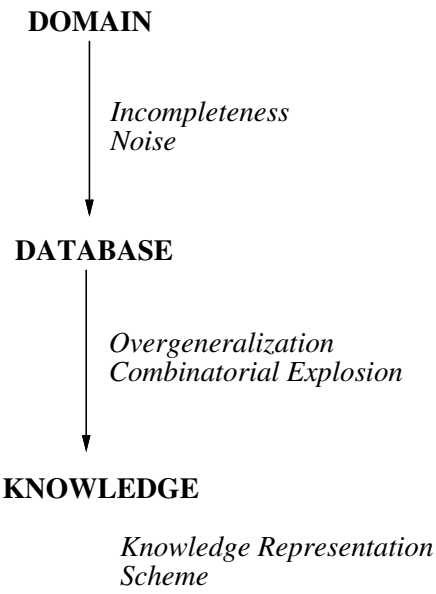


Figure 1: The process of Data Mining.

Data Mining (Figure 1) presupposes a database (or several databases) containing data about a domain. The collected data will most of time be a noisy and incomplete description of the domain. The Machine Learning or Pattern Recognition technique should extract structured knowledge from these data, using some knowledge representation scheme such as if-then rules or decision trees, and is confronted during this process with problems of *overgeneralization* (a learning system should go beyond the data, making inductive leaps which may be incorrect), and *combinatorial explosion* (in general a computationally intractable number of hypotheses can explain the same data). Moreover, it should do so on the basis of incomplete and noisy data about the domain.

In linguistic applications, the domain is constituted by the (hypothesised) regularities (in terms of rules and concepts) governing linguistic behaviour which the linguist wants to discover, the data are corpora of actual language use, and the resulting knowledge is the proposed theory about the regularities, which should explain the corpora.

Data mining techniques can be and have been used in linguistic engineering to solve the *knowledge acquisition bottleneck* in linguistic engineering, i.e. the fact that lexical and grammatical knowledge usually has to be reformulated from scratch whenever a new application has to be built or an existing application ported to a new domain. In addition, we will show in this paper that there are basically two ways in which data mining techniques can be used in linguistic theory formation:

Evaluating hypotheses. In order to evaluate competing theories about a linguistic phenomenon, the following procedure can be applied.

1. Collect a representative corpus of data about the phenomenon.
2. Annotate the corpus according to the requirements of each competing theory or hypothesis (i.e. using the concepts deemed necessary by that theory or hypothesis for the description of the phenomenon).
3. Compute learnability using a simple, neutral, statistical learning algorithm (e.g. k-nearest-neighbour, Baysian learning).[1]

By comparing the performance of the system when trained on these differently annotated corpora, claims about necessity of particular information for the explanation of the phenomenon can be tested.

Discovery of theories. In order to discover new generalizations or concepts, the following procedure can be used.

1. Collect a representative corpus of data.
2. Annotate the corpus with any linguistic information which might be relevant.
3. Extract generalizations and categories using (sophisticated) learning algorithms.

E.g. suppose you want to evaluate different competing theories about sentence accent computation. A corpus could be collected with correct sentence accent markings. Different hypotheses about which type of linguistic knowledge plays a role in sentence accent assignment could be evaluated by encoding the corpus several times using these different knowledge sources (phonological, morphological, syntactic tags, discourse structure etc.). By comparing the learnability of sentence accent using these differently annotated corpora as training material, the hypotheses can be evaluated. Alternatively, a maximally annotated corpus could be used to infer new generalizations about the role of different levels of annotation and their interaction.

In the remainder of this paper, we will describe a simpler case study of using an inductive data mining algorithm (C4.5, Quinlan 1993, a state of the art rule induction programme) to test linguistic hypotheses and to discover regularities and categories. The case study concerns allomorphy in Dutch diminutive formation, "one of the more vexed problems of Dutch phonology (...) one of the most spectacular phenomena of modern Dutch morphophonemics" (Trommelen 1983). In this case we will use the same, moderately sophisticated, machine learning method to illustrate both the hypothesis testing and linguistic discovery aspects of data mining.

[1] An additional step would be to compute and compare metrics on each annotated corpus (information entropy, information gain of features etc.), but we will not go into this matter in this paper.

2 Dutch Diminutive Formation

2.1 Allomorphy in Dutch Diminutive Formation

In standard Dutch, the diminutive suffix occurs in five different variants:

Noun	Form	Suffix
huis (house)	huisje	*-je*
man (man)	mannetje	*-etje*
raam (window)	raampje	*-pje*
woning (house)	woninkje	*-kje*
baan (job)	baantje	*-tje*

The frequency distribution of the different categories is given in the folowing table. We distinguish between database frequency (frequency of a suffix in a list of 3900 diminutive forms of nouns we collected) and corpus frequency (frequency of a suffix in the text corpus on which the word list was based).

Suffix	Frequency	Database %	Corpus %
tje	1897	48.7%	50.9%
je	1462	37.5%	30.4%
etje	357	9.7%	10.9%
pje	104	2.7%	4.0%
kje	77	2.0%	3.8%

2.2 Linguistic Analysis of Diminutive Formation

Historically, different analyses of diminutive formation have taken a different view of the rules that govern the choice of the diminutive suffix, and of the linguistic concepts playing a role in these rules (see e.g. Te Winkel 1866, Kruizinga 1915, Cohen 1958, and references in Trommelen 1983).

In Trommelen (1983), it is argued that diminutive formation is a local process, in which concepts such as word stress and morphological structure (proposed in the earlier analyses) do not play a role. The rhyme of the last syllable of the noun is necessary and sufficient to predict the correct allomorph. The rules Trommelen uses are listed here.

$$\text{DIM} \rightarrow \text{etje} \;/\; \text{V}[+\text{sonorant}]\#_$$

$$\text{DIM} \rightarrow \text{pje} \;/\; \left\{ \begin{array}{l} [\,+\text{long}\,] \\ [\,\text{ə}\,] \\ [\,+\text{short}\,] \end{array} \right. \quad \left[\begin{array}{l} \text{C} \\ [\,+\text{sonorant}\,] \end{array} \right] \left. \vphantom{\begin{array}{l} a \\ a \\ a \end{array}} \right\} \;/\text{m}/\#_$$

$$\text{DIM} \rightarrow \text{kje} \;/\; /\text{ŋ}/\#_$$

DIM→je / [+obstruent]#_

DIM→tje / Default rule

The natural classes (or concepts) which are hypothesised in these rules include *obstruents*, *sonorants*, *obstruents*, and the class of *bimoric vowels* (consisting of long vowels, diphthongs and ə).

In summary, we have here a small, relatively easy, linguistic domain for which different competing theories have been proposed, and for which different generalizations (in terms of rules and linguistic categories) have been proposed. What we will show next is how data mining techniques may be used to (i) test competing hypotheses, and (ii) discover generalizations in the data which can then be compared to the generalizations formulated by linguists. But first we will discuss the data mining technique we used.

3 Machine Learning Method

For the experiments, we used C4.5 (Quinlan, 1993). C4.5 is a TDIDT (Top Down Induction of Decision Trees) decision tree learning algorithm which constructs a decision tree on the basis of a set of examples (the training set). This decision tree has tests (feature names) as nodes, and feature values as branches between nodes. The leaf nodes are labeled with a category name and constitute the output of the system. A decision tree constructed on the basis of examples is used after training to assign a class to patterns. To test whether the tree has actually learned the problem, and has not just memorized the items it was trained on, the *generalization* accuracy is measured by testing the learned tree on a part of the dataset not used in training.

The algorithm for the constructing of a C4.5 decision tree can be easily stated. Given a training set T (a collection of examples), and a finite number of classes C_1 ... C_n.

- If T contains one or more cases all belonging to the same class C_j, then the decision tree for T is a leaf node with category C_j.
- If T is empty, a category has to be found on the basis of other information (e.g. domain knowledge). The heuristic used here is that the most frequent class in the initial training set is used.
- If T contains different classes then
  - Choose a test (feature) with a finite number of outcomes (values), and partition T into subsets of examples that have the same outcome for the test chosen. The decision tree consists of a root node containing the test, and a branch for each outcome, each branch leading to a subset of the original set.

– Apply the procedure recursively to subsets created this way.

In this algorithm, it is not specified which test to choose to split a node into subtrees at some point. Taking one at random will usually result in large decision trees with poor generalization performance, as uninformative tests may be chosen. Considering all possible trees consistent with the data is computationally intractable, so a reliable heuristic test selection method has to be found. The method used in C4.5 is based on the concept of *mutual information* (or *information gain*). Whenever a test has to be selected, the feature is chosen with the highest information gain. This is the feature that reduces the information entropy of the training (sub)set on average most, when its value would be known. For the computation of information gain, see Quinlan (1993).

Decision trees can be easily and automatically transformed into sets of if-then rules (production rules), which are in general easier to understand by domain experts (linguists in our case). The C4.5 algorithm also contains a *value grouping* method which, on the basis of statistical information, collapses different values for a feature into the same category. That way, more concise decision trees and rules can be produced (instead of several different branches or rule conditions for each value, only one branch or condition has to be defined, making reference to a class of values).

4 Experimental Set-up

4.1 Corpus

We collected a corpus of about 3900 Dutch nouns from the CELEX database. For each noun, the following information was kept.

1. The phoneme transcription describing the syllable structure (in terms of onset, nucleus, and coda) of the last three syllables of the word.

2. For each of these three last syllables the presence or absence of stress.

3. The corresponding diminutive allomorph, abbreviated to E (-etje), T (-tje), J (-je), K (-kje), and P (-pje). This is the 'category' of the word to be learned by the learner.

Some examples are given below (the word and its gloss are provided for convenience and were not used in the experiments).

```
- b i = - z @ = + m A nt J biezenmandje (basket)
= = = = = = = = + b I x E big (pig)
= = = = + b K = - b a n T bijbaan (job on the side)
= = = = + b K = - b @ l T bijbel (bible)
```

Figure 2: Learning curves of C4.5 for each allomorph

4.2 Experimental Method

The experimental set-up used in all experiments consisted of a ten-fold cross-validation experiment (Weiss & Kulikowski 1991). In this set-up, the database is partitioned ten times, each with a different 10% of the dataset as the test part, and the remaining 90% as training part. For each of the ten simulations in our experiments, the test part was used to test generalisation performance. The success rate of an algorithm is obtained by calculating the average accuracy (number of test pattern categories correctly predicted) over the ten test sets in the ten-fold cross-validation experiment.

4.3 Learnability

The experiments show that the diminutive formation problem is learnable in a data-oriented way (i.e. by extraction of regularities from examples, without any a priori knowledge about the domain). The average error rate of 1.6% should be compared to baseline performance measures based on (intelligent) guessing. This baseline would be an error rate of about 60% for this problem. By observing the generalization performance of the system with different amounts of examples, we can get an insight into the relative learnability of different allomorphs.

Figure 2 shows the learning curves of C4.5 for the different allomorphs. A learning curve expresses the generalization accuracy of a system for increasing sizes of the training set.

The figure shows that whereas *-tje, -kje, -pje* and *-je* have roughly expo-

nential learning curves, the learning curve of *etje* is roughly linear.

So far, we have provided a rationale for using data mining techniques in linguistic research, we have selected a domain (Dutch diminutive formation) and a data mining method (the C4.5 machine learning method), and have described the corpus of examples that (in data mining terminology) plays the role of the database describing the domain, and the experimental method. We have also shown that the problem is indeed learnable by the learning method selected. In the next two sections, we will describe the results of the experiments; first on the task of comparing conflicting hypotheses (section 5), then on discovering linguistic generalizations (section 6).

5 Linguistic Hypothesis Testing

On the basis of the analysis of Dutch diminutive formation by Trommelen (1983), discussed briefly in section 2, the following hypotheses (among others) can be formulated.

1. Only information about the last syllable is relevant in predicting the correct allomorph.
2. Information about the onset of the last syllable is irrelevant in predicting the correct allomorph.
3. Stress is irrelevant in predicting the correct allomorph.

In other words, information about the rhyme of the last syllable of a noun is necessary and sufficient to predict the correct allomorph of the diminutive suffix. To test these hypotheses, we performed four experiments, training and testing the C4.5 machine learning algorithm each time with a different corpus. These corpora contained the following information.

1. All information (stress, onset, nucleus, coda) about the three last syllables (3-SYLL corpus).
2. All information about the last syllable (SONC corpus).
3. Information about the last syllable without stress (ONC corpus).
4. Information about the last syllable without stress and onset (NC corpus).

5.1 Results

The following table lists the learnability results[2]. The generalization error is given for each allomorph for the four different training corpora:

[2] It should be noted that CELEX contains a number of coding errors, so that some of the 'wrong' allomorphs predicted by the data mining system were actually correct, we did not correct for this.

	Errors and Error percentages							
Suffix	3 SYLL		SONC		ONC		NC	
Total	61	1.6	79	2.0	80	2.0	77	2.0
-tje	13	0.69	13	0.69	14	0.74	14	0.74
-je	16	1.09	15	1.03	16	1.09	14	0.96
-etje	26	7.28	49	13.73	48	13.45	44	12.32
-kje	4	5.19	0	0	0	0	0	0
-pje	2	1.92	2	1.92	2	1.92	5	4.81

The overall best results are achieved with the most elaborate corpus (containing all information about the three last syllables), suggesting that, contra Trommelen, important information is lost by restricting attention to only the last syllable. As far as the different encodings of the last syllable are concerned, however, the learnability experiment corroborates Trommelen's claim that stress and onset are not necessary to predict the correct diminutive allomorph. When we look at the error rates for individual allomorphs, a more complex picture emerges. The error rate on *-etje* dramatically increases (from 7% to 14%) when restricting information to the last syllable. The *-kje* allomorph, on the other hand, is learned perfectly on the basis of the last syllable alone. What has happened here is that the learning method has *overgeneralised* the rule

DIM→kje / /Iŋ/#_

because the data do not contain enough information to correctly handle the notoriously difficult opposition between words like *leerling* (pupil, takes *-etje*) and *koning* (king, takes *-kje*). Furthermore, the error rate on *-pje* is doubled when onset information is left out from the corpus.

We can conclude from these experiments that although the broad lines of the analysis by Trommelen (1983) are correct, the learnability results point at a number of problems with it (notably with *-kje* versus *-etje* and with *-pje*). We will move now to the use of C4.5 as a generator of generalizations about the domain. And compare these generalizations to the analysis of Trommelen.

6 Linguistic Discovery

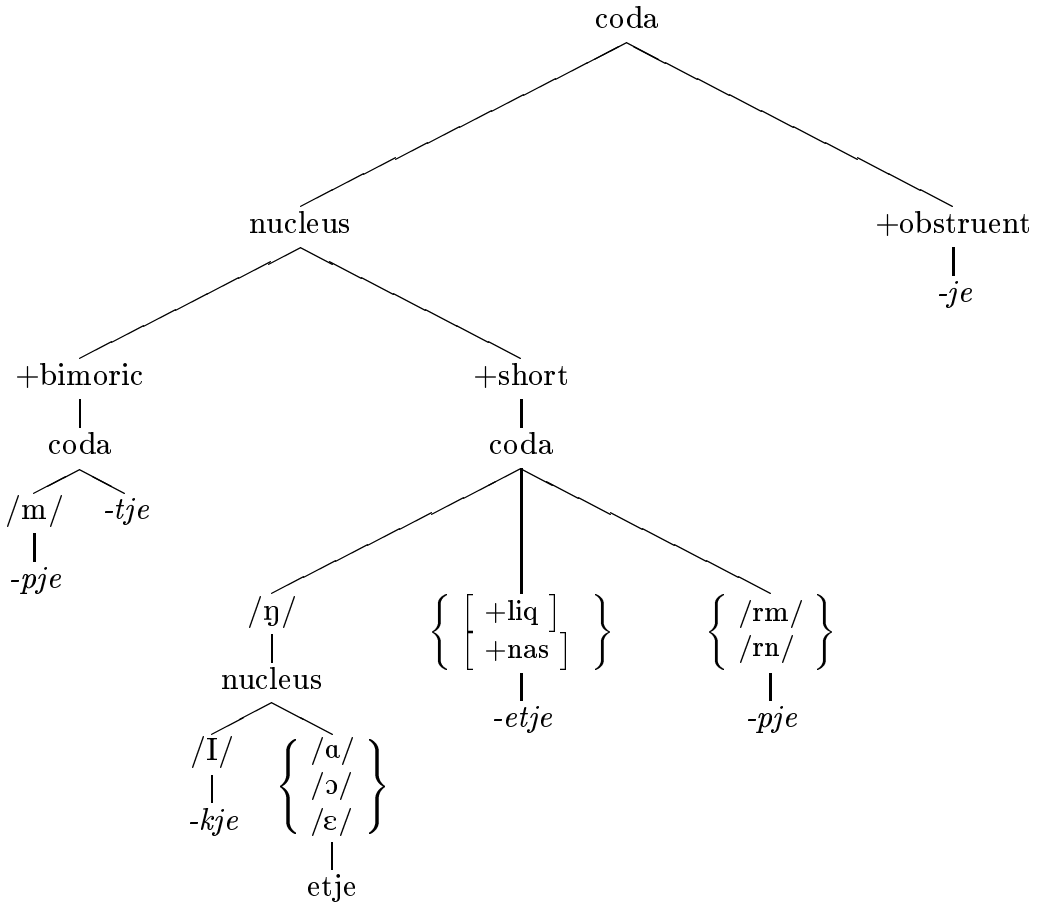
When looking only at the rhyme of the last syllable (the NC corpus), the raw decision tree generated by C4.5 looks as follows:

```
Decision Tree:

coda in {rk,nt,lt,rt,p,k,t,st,s,ts,rs,rp,f,
         x,lk,Nk,mp,xt,rst,ns,nst,
```

```
         rx,kt,ft,lf,mt,lp,ks,ls,kst,lx}: J
coda in {n,=,l,j,r,m,N,rn,rm,w,lm}:
|   nucleus in {I,A,},O,E}:
|   |   coda in {n,l,r,m}: E
|   |   coda in {=,j,rn}: T
|   |   coda in {rm,lm}: P
|   |   coda = N:
|   |   |   nucleus = I: K
|   |   |   nucleus in {A,O,E}: E
|   nucleus in {K,a,e,u,M,@,y,o,i,L,),|,<}:
|   |   coda in {n,=,l,j,r,rn,w}: T
|   |   coda = m: P
```

Notice that the phoneme representation used by CELEX (called DISC) is shown here instead of the more standard IPA font, and that the value grouping mechanism of C4.5 has created a number of phonological classes by collapsing different phonemes into sets indicated by curly brackets. A reformulation of the previous tree using standard phonological terminology and the IPA font looks like this.



This decision tree should be read as follows: first check the coda (of the last syllable). If it ends in an obstruent, the allomorph is *-je*. If not, check

the nucleus. If it is bimoric, and the coda is /m/, decide *-pje*, if the coda is not /m/, decide *-tje*. It is interesting to note that what we have called the *bimoric* category here corresponds completely with the category hypothesised by Trommelen and containing the long vowels, the diphthongs and the ə. In other words, C4.5 has discovered this set of phonemes to be a useful category in solving the diminutive formation problem. Moving to the right hand side of the decision tree, we find that when the coda is not an obstruent, the nucleus is short and the coda is /ŋ/, we have to look at the nucleus again to decide between *-kje* and *-etje* (this is where the overgeneralization to *-kje* for words in -ing occurs). Finally, the coda (nasa-liquid or not) helps us distinguish between *-etje* and *-pje* for those cases where the nucleus is short. It should be clear that this tree can easily be formulated as a set of rules without loss of accuracy. While constructing the decision tree, several phonologically relevant categories are 'discovered' by the value grouping mechanism in C4.5, including nasaliquids, the obstruents, the short vowels, and the bimoric vowels (which was a completely new category when Trommelen proposed it in 1983).

In the previous section, we noticed that the *-etje* versus *-kje* problem for words ending in *-ing* could not be solved by referring only to the last syllable (C4.5 and any other statistically based induction algorithm would overgeneralize to *-kje*). The following is the knowledge derived by C4.5 from the full corpus, with all information about the three last syllables (the 3 SYLL corpus). We provide the rule version of the inferred knowledge this time.

```
        Default class is -tje


1.      IF coda last is /lm/ or /rm/
        THEN  -pje

2.      IF nucleus last is [+bimoric]
           coda last is  /m/
        THEN  -pje

3.      IF nucleus last is [+short]
           coda last is [+nas] or [+liq]
        THEN  -etje

4.      IF coda last is /N/
        THEN IF  nucleus penultimate is empty (monosyllabic word) or
                 schwa
             THEN  -etje
             ELSE  -kje

5.      IF coda last is [+obstruent]
```

```
THEN  -je
```

The default class is *-tje*, which is the allomorph chosen when none of the other rules apply. This explains why this rule set looks simpler than the decision tree earlier.

The first thing which is interesting in this rule set, is that only three of the twelve presented features (coda and nucleus of the last syllable, nucleus of the penultimate syllable) are used in the rules. Contrary to the hypothesis of Trommelen, apart from the rhyme of the last syllable, the nucleus of the penultimate syllable is taken to be relevant as well.

The induced rules roughly correspond to the previous decision tree, but in addition a solution is provided to the *-etje* versus *-kje* problem for words ending in *-ing* (rule 4) making use of information about the nucleus of the penultimate syllable. Rule 4 states that words ending in /ŋ/ get *-etje* as diminutive allomorph when they are monosyllables (nucleus of the penultimate syllable is empty) or when they have a ə as penultimate nucleus, and *-kje* otherwise. As far as we now, a similar rule has not yet been proposed in the published literature on diminutive formation.

Although a comparison of the performance of the rules suggested by Trommelen (section 2) and the rules suggested by C4.5 is not of central concern in this paper, it provides a useful perspective on the accuracy of automatically induced rules. The following table shows for each allomorph the number of errors by the C4.5 rules (trained using corpus NC, i.e. only the rhyme of the last syllable) as opposed to an implementation of the rules suggested by Trommelen. One problem with the latter is that they often suggest more than one allomorph (the rules are not mutually exclusive). In those cases where more than one rule applies, a choice was made at random. The comparison shows that C4.5 did a good job of finding an elegant and accurate rule-based description of the problem.

Suffix	Trommelen	C4.5
-tje	53	11
-je	12	12
-etje	28	39
-kje	38	0
-pje	21	4
Total	152	66

7 Conclusion

We have shown by example that data mining techniques can profitably be used in linguistics as a tool for the comparison of linguistic theories and hypotheses

or for the discovery of new linguistic theories in the form of linguistic rules or categories.

The case study we presented concerns diminutive formation in Dutch, for which we showed that (i) data mining techniques can be used to corroborate and falsify some of the existing theories about the phenomenon, and (ii) data mining techniques can be used to (re)discover interesting linguistic rules (e.g. the rule solving the *-etje* versus *-kje* problem) and categories (e.g. the category of bimoric vowels).

References

Cohen, A. (1958). Het Nederlandse diminutiefsuffix; een morfologische proeve. *De Nieuwe Taalgids*, jaargang 51.

Kruizinga, E. (1917). De vorm van verkleinwoorden. *De Nieuwe Taalgids.*

Piatetsky-Shapiro, G., and Frawley, W. (1991). *Knowledge Discovery in Databases.* AAAI Press.

Quinlan, J. R. (1993). *C4.5 Programs for machine learning.*

Trommelen, M. (1983). *The syllable in Dutch, with special reference to diminutive formation.* Dordrecht: Foris.

Weiss, S., and Kulikowski, C. (1991). *Computer Systems that Learn.* San Mateo: Morgan Kaufmann.

Winkel, L. A. T. (1866). *Leerboek der Nederlandsche Spelling.* Leiden: D. Noothoven en Van Goor.