Walter Daelemans
University of Antwerp
Linguistics – CNTS
Universiteitsplein 1
B-2610 Wilrijk
Belgium

90. Machine Learning of Natural Language

1. Introduction

In this article we provide an overview of recent research on the application of symbolic Machine Learning techniques to language data (Machine Learning of Natural Language, MLNL). Both in Quantitative Linguistics (QL) and in MLNL, the main goal is to describe the language as it is observed with rules, language models, or other descriptions. But whereas the motivation in QL is purely scientific (establishing the laws and mathematical properties of language), the motivation in MLNL is partly pragmatic: increasing the accuracy and efficiency of Natural Language Processing (NLP) systems, or the speed with which they can be built. Nevertheless, the extracted generalizations can provide worthwhile insight into the language task being studied in an MLNL framework.

There are different ways in which the algorithmic acquisition of language knowledge and behavior is studied. We cannot possibly discuss all this relevant work in the context of this article. One important area of research that will be omitted in this overview is the *computational modeling of human language acquisition* using statistical, machine learning or neural network methods (McClelland and Rumelhart 1986; Brent 1996; Broeder and Murre 2000). Salient modeling results in this area include the learning of word meaning (Siskind 1996; Resnik 1996), finding structure in sequences (Elman 1990), bootstrapping word segmentation (Brent and Cartwright 1996), and setting parameters in a principles and parameters approach (Niyogi and Berwick 1994; Dresher and Kaye 1990) as well as empiricist alternatives to the latter approach (MacWhinney and Leinbach 1991; Gillis et al. 1993; Gillis, Durieux and Daelemans 1995).

The development of algorithms for the *inference of formal grammars* from (mostly artificial) language data is another relevant area of research that will not be covered in this article. See Honavar and Slutzki (1998) for a collection of research in this area.
We restrict our overview here to language learning in the context of Natural Language Processing and Computational Linguistics. We first provide a brief introduction to the 'empirical revolution' in NLP, and the increased attention for Machine Learning (ML) methods that followed it. Then we provide a general model of ML, and a taxonomy and overview of the main learning algorithms. We then restrict our attention to two main

classes of *symbolic* ML methods applied to NLP (memory-based learning and rule learning), explain how they work, and describe some representative applications to language processing problems.

2. The Empirical Revolution in Natural Language Processing

Natural Language Processing (NLP) studies the knowledge representation and problem solving issues involved in learning, producing, and understanding language. Although the origins of NLP are both knowledge-based and statistical, as in other disciplines of Artificial Intelligence, the knowledge-based approach has historically dominated this field. This has resulted in an emphasis on logical semantics for meaning representation, on the development of grammar formalisms (especially lexicalist unification grammars), and on the design of associated parsing methods and lexical representation and organization methods. Well-known textbooks provide an overview of this approach (Gazdar and Mellish 1989; Allen 1995).

From the early nineties onwards, empirical methods based on corpus-based statistics, have gradually been re-introduced in the field, and have started to dominate it by the start of this century, as can be seen from the number of papers subscribing to this approach in computational linguistics journals and conference proceedings. There are many reasons for this. Firstly, computer processing and storage capabilities have advanced to such an extent that statistical pattern recognition methods have become feasible on the large amounts of text and speech data that are now available in electronic form. Secondly, there has been an increase of interest within NLP (prompted by application-oriented and competitive funding) for the development of methods that scale well and can be used in real applications without requiring a complete syntactic and semantic analysis of text. Finally, simple statistical methods have been enormously successful in speech technology and information retrieval, and have therefore been applied to NLP as well. See Brill and Mooney (1998) and Church and Mercer (1993) for overviews of this empirical 'revolution' in NLP. The maturity of the approach is borne out by the publication of a few recent textbooks (Charniak 1993; Manning and Schütze 1999).

Comparing these empirical methods to the knowledge-based approach, it is clear that the former have a number of advantages. In general, statistical approaches have a greater coverage of syntactic constructions and vocabulary, they are more robust (graceful degradation), they are reusable for different languages and domains, and development times for making applications and systems are shorter. On the other hand, knowledge-based methods allow the incorporation of linguistic knowledge and sophistication, making them sometimes more precise. Three crucial problems for (statistical) empirical methods are (i) the *sparse data problem*: often not enough data is available to estimate the probability of (low-frequency) events accurately, (ii) the *relevance problem*: it is often difficult to estimate reliably the importance or relevance of particular statistical events for the solution of the NLP problem, and (iii) the *interpretation problem*: most statistical techniques do not provide insight into how a trained statistical system solves a task.

The last few years have witnessed an increase of the use of machine learning methods in NLP. Some of these methods were created from within NLP (e.g. transformation-based error driven learning (Brill 1992), other techniques were imported from Machine Learning into NLP; e.g. induction of decision trees and rules (Quinlan 1993; Cohen 1995), inductive logic programming (Lavrac and D˘zeroski 1994), memory-based learning (Aha, Kibler, and Albert 1991), and support vector machines (Vapnik 1995). Machine Learning (ML) is the sub-discipline of Artificial Intelligence (AI) which studies algorithms that can learn either from experience or by reorganizing the knowledge they already have. See Langley (1996) and Mitchell (1997) for introductory material, Weiss and Kulikowski (1991) for methodological issues, and Natarajan (1991) for a formal-theoretical approach.

There are also several recent collections of papers on Machine Learning applied to Natural Language (Wermter, Riloff, and Scheler 1996, Brill and Mooney 1998, Daelemans, Weijters and Van den Bosch 1997, Cardie and Mooney 1999). Machine learning methods hold promise for solving the problems with statistical methods noted earlier. They incorporate new methods for smoothing data to solve sparse data problems and for assigning relevance to linguistic data, they allow the incorporation of linguistic background knowledge, and what they have learned is to a certain extent interpretable.

3. Learning from Examples

In the machine learning algorithms we will discuss here, learning works by extracting generalizations from a set of examples of a desired input-output mapping. For example, for learning the generalizations involved in predicting the right plural suffix for a German noun, several examples of nouns (described in terms of their phonology, lexical information such as gender, etc.) with their corresponding plural suffix would be given. The relations between input (typically a feature vector, here the properties of German nouns) and output (typically a symbol, here the plural suffix), implicit in these examples, are discovered by the algorithm, and are used to predict the correct output when presented with a new, previously unseen, input pattern. In other words, the algorithm *classifies* a new input pattern as belonging to a particular output category. A machine learning algorithm trained on a particular set of data is therefore called a *classifier*. This type of learning is often called supervised learning, and is contrasted with unsupervised learning, where examples are presented without information about the desired output. It is then up to the system to find similarities in the examples in such a way that they can be exploited in solving the task. We will not discuss unsupervised learning any further here.

Many problems in NLP, especially disambiguation problems, can be formulated as classification tasks (Magerman 1994; Daelemans 1995; Cardie 1994). As an example, consider morphosyntactic disambiguation (part of speech tagging): learning the assignment of the contextually and lexically most probable word class of a word in context. E.g., in *the old man the boats*, context dictates that *old* is an adjective, and *man* a verb, contrary to what would be predicted by looking only at the lexical probabilities of these combinations of word and word class. This type of word class disambiguation has become a benchmark problem for learning approaches to NLP. An overview of machine learning work for tagging can be found in Daelemans (1999a).

In morphosyntactic word class tagging, abbreviated tagging from here, a sentence should be mapped into a string of morpho-syntactic tags (Table 1).

Input					Output				
John	will	join	the	board	Name	Modal	Verb	Determiner	Noun

Table 1. Tagging as a mapping from sentences to tag strings.

By approximating this mapping with a function from a focus word and its context to the disambiguated tag belonging to the focus word in that context (Table 2), the mapping becomes a *classification* task amenable to Machine Learning approaches. Of course, instead of (only) the words in the context, more information would be added in real experiments: e.g. varying sizes of context, morphological, syntactic, or any other available linguistic information.

Left Context		**Focus**	*Right Context*		**Class**
=	=	John	will	join	Name
=	John	will	join	the	Modal
John	will	join	the	board	Verb
will	join	the	board	=	Determiner
join	the	board	=	=	Noun

Table 2. Tagging as a mapping from focus words with context to tags.

It is easy to see how similar classification tasks can be set up for other NLP problems such as word sense disambiguation, term translation, morphology, etc. Even parsing can be handled this way by cascading different partial systems such as a tagger, a constituent finder, and a classifier disambiguating possible relations between constituents. In all these cases we have some focus unit (letter, word, constituent) and a representation of its context as features, and a unit at another linguistic level as output class.

3.1 Machine Learning

Conceptually, a learning system consists of a *performance component* which achieves a specific task (given an input, it produces an output), and a *learning component* which modifies the performance component on the basis of its experience in such a way that performance of the system in doing the same or similar tasks improves. As we have seen, examples take the form of pairs of inputs with their associated desired output.

To achieve its task, the performance component uses an internal representation. The task of the learning component may therefore be construed as a search in the space of possible representations (often called the hypothesis space) for a representation that is optimal for performing the mapping. In this article, we will consider among others decision trees, rules, and case bases as types of languages/formalisms for internal representations for language processing. In most cases, finding the optimal representation given a set of examples and a representation language is computationally intractable. Some form of heuristic search is therefore used by all learning systems.

In Machine Learning, the concept of *bias* refers to constraints on this search process. These constraints may be domain-dependent. In that case, knowledge about the task is used to make the search simpler. This bias may be present in the way the experience presented to the learning component (the training examples) is represented or in heuristic knowledge used to prune the search tree. The addition of linguistic bias to a learning system is the obvious way to let learning natural language processing systems profit from linguistic knowledge about the task. On top of that, there is also a more general notion of bias in the restrictions on what can be represented in the representation language used (language bias), or in general principles guiding the search of the search algorithm implicit in the learning algorithm (search bias, e.g. Ockham's razor).

3.2 Performance Evaluations

The success of a learning component in improving performance can be evaluated using a number of different quantitative and qualitative measures:

- *Generalization accuracy*. This is the performance accuracy of the system on previously unseen inputs (i.e., inputs it was not trained on). This aspect of learning is of course crucial: it gives an indication of the quality of the *inductive leap* made by the algorithm on the basis of the examples. A good generalization accuracy indicates that the learning system has not overfit its training examples, as would happen by generalizing on the basis of errors or exceptions present in them. To get a good estimate of the real generalization accuracy, *cross-validation* can be used, e.g. in 10-fold cross-validation an algorithm is tested on ten different partitions (90% training material, 10% testing material) of the full data set available. Each data item occurs once in one of the test sets. The average generalization accuracy on the ten test sets is then a good statistical estimate of the real accuracy. Apart from accuracy, for some NLP problems the notions of *recall* and *precision* are more appropriate. For example, when the task is chunking, i.e., finding noun phrases or prepositional phrases in text, precision measures the percentage of correct chunks in all chunks predicted by the algorithm, and recall measures the percentage of chunks present in the test data that was correctly identified by the algorithm. Combined precision-recall measures give a better indication of the goodness of a system in these cases than accuracy measurements.

- *Space and time complexity*. The amount of storage and processing involved in learning (training the system) and performance (producing output given the input).
- *Explanatory Quality*. Usefulness of the representations found by the learning system as an explanation of the way the task is achieved. Especially with good explanatory quality, the machine learning results may provide useful and new linguistic insight into the task being learned.

3.3 Overview of Methods

To sum up this section, we will give an intuitive description of how a number of learning algorithms works. We discuss the algorithms in an order of increasing abstraction of the internal representation used by the performance component, and created by the learning component. We start from storage and table-lookup of the 'raw' examples as a non-learning baseline.

- *Table Look-Up*. Store all examples (patterns of input and their corresponding output) in a table. When a new input pattern is given to the performance system, look it up in the table, and retrieve the output of the stored example. In this approach the system does not actually learn anything, and it fails miserably whenever an input pattern is not present in the table (there is no generalization). Nevertheless, for language problems, when sufficient training data is available and a simple heuristic is used for missing patterns (e.g. take the class most often occurring in the training data), sometimes an astonishingly high accuracy is already obtained with this non-learning method.
- *Memory-Based Learning*. Store all examples in a table. When a new input pattern is given to the performance system, look up the most similar examples (in terms of number of feature values common to the stored pattern and the new pattern, for example) to the new pattern, and extrapolate from the tags assigned to these nearest matches to the new case. Various statistical and information-theoretic techniques can be used to design the *similarity metric*. The similarity metric is also a place where linguistic bias can be introduced in the learning algorithm, making the definition of what is similar domain-dependent.
- *Rule and Decision Tree Induction*. Use similarities and differences between examples to construct a decision tree or a rule set (these two are largely equivalent and can be translated to each other), and use this constructed representation to assign a class to a new input pattern. Forget the individual examples. A special subclass of these methods is *Inductive Logic Programming*, which in principle could learn problems for which feature-value-based algorithms fail by using first-order logic as a representation language.
- *Connectionism, Neural Networks*. Use the examples to train a network. In back-propagation learning, this training is done by repeatedly iterating over all examples, comparing for each example the output predicted by the network (random at first) to the desired output, and changing connection weights between network nodes in such a way that performance increases. Keep the connection weight matrix, and forget the examples.

- *Statistical Methods*. Compute a statistical model (e.g. about the *n*-grams occurring in the language) on the examples (the corpus), forget the examples, and use the model to extrapolate to the most probable analysis of new input.

In terms of abstraction versus data-orientation, stochastic, neural network, and rule induction approaches are *eager learning* techniques. These techniques abstract knowledge from the examples as soon as they are presented. Memory-Based Learning is a *lazy learning* technique; generalization only occurs when a new pattern is offered to the performance component, and abstraction is therefore implicit in the way the contents of the case base and the similarity metric interact.

A method that is unlike any other methods described in this inventory is the evolutionary programming approach (genetic algorithms and genetic programming). It is completely different from other learning methods, as it is not based on looking for *similarity* in data as the main bias. These methods basically perform a random search in the hypothesis space, directed by a heuristic fitness function. An initially randomly chosen population of representations (e.g. rules, or rule sets, or parametric descriptions of a neural network, etc.) is evolved over a number of generations. To decide survival into the next generation, a single fitness number is assigned to each individual in the population, based on an evaluation of the individual (e.g. testing the rule on some set of test data). The fittest individuals are selected for recombination and allowed to reproduce using crossover and mutation operators (Goldberg 1989; Mitchell 1996; Koza 1992). Applications to NLP are not (yet) numerous and mostly concern artificial language learning or the parameter optimization part of a hybrid approach in which the evolutionary method is combined with some other learning method. Kazakov and Manandhar (1998) is a good example of this approach. Computational complexity of these algorithms still seems to be the main obstacle to applying them to solve real-world language processing problems.

Popular as they may be, we will also not discuss neural network research further here. There is a considerable body of research on applying neural network technology to language processing problems (Reilly and Sharkey 1992; Sharkey 1992; Wermter, Riloff and Scheler 1996). In general, as with statistical methods, it is hard to interpret what has been learned from a trained neural network.

In the remainder of this article, we will discuss two important types of *symbolic* ML methods in turn, and provide an overview of how they have been applied to NLP tasks. These symbolic methods allow, at least in theory, to obtain knowledge that is comprehensible, making it possible to manually edit it, integrate it with hand-built systems, etc.

4. Memory-Based Learning

The memory-based learning paradigm is founded on the hypothesis that performance in cognitive tasks (in this case language processing) is based on reasoning on the basis of analogy of new situations to stored representations of earlier experiences rather than on the application of mental rules abstracted from representations of earlier experiences as in rule induction and rule-based processing.

The concept has appeared in several AI disciplines (from computer vision to robotics), using apart from memory-based learning also labels such as memory-based reasoning, case-based reasoning, exemplar-based learning, locally-weighted learning, and instance based learning (Stanfill and Waltz 1986; Cost and Salzberg 1993; Riesbeck and Schank 1989; Kolodner 1993; Atkeson, Moore and Schaal 1997; Aha 1997; Aamodt and Plaza 1994). Interestingly, when applied to NLP, it finds its inspiration not only in statistical pattern recognition (Fix and Hodges 1951; Cover and hart 1967), but also in the linguistics of de Saussure and Bloomfield, and in the operationalisation of analogy in linguistics of the American linguist Royal Skousen (1989; 1992). The linguistic motivation for this and other memory-based approaches is (i) the fact that in actual language use there is not a clear-cut all-or-none distinction between regular and irregular cases, (ii) the simplicity of the analogical approach as opposed to rule discovery, and (iii) the adaptability of the approach as opposed to the static, rigid rule-based alternative. Remarkably, seen from the outside, such an analogical approach appears to be rule-governed, and therefore adequately explains intuitions about linguistic generalizations as well.

4.1 Algorithm

Examples are represented as a vector of feature values with an associated category label. Features define a pattern space. During training, a set of examples (the training set) is presented in an incremental fashion to the learning algorithm, and added to memory. During processing, a vector of feature values (a previously unseen test pattern) is presented to the system. Its distance to all examples in memory is computed using a *similarity metric*, and the category of the most similar instance(s) is used as a basis to predict the category for the test pattern.

In this type of lazy learning, performance crucially depends on the similarity metric used. The most straightforward metric for a problem with nominal (non-numeric) feature values would be an *overlap metric*: similarity is defined as the number of feature values that are equal in two patterns being compared. In such a distance metric, all features describing an example are interpreted as being equally important in solving the classification problem, but this is not necessarily the case: e.g. in morphosyntactic disambiguation, the word class of the word immediately before a word to be tagged is obviously more important than the category of the word three positions earlier in the sentence. This is the feature relevance problem we introduced earlier as one of the problems for statistical methods. Various feature weighting and selection methods have been proposed to differentiate between the features on the basis of their relevance for solving the task (see Wettschereck, Aha and Mohri 1997) for an overview.

Another addition to the basic algorithm that has proved relevant for many natural language processing tasks is a value difference metric (Stanfill and Waltz 1986; Cost and Salzberg 1993). Such a metric assigns different distances to pairs of values for the same feature. In tagging e.g., such a metric would assign a smaller distance between NOUN-SINGULAR and NOUN-PLURAL than between NOUN-PLURAL and VERB. These biases can of course also be added by hand to the learner (e.g., by a domain expert). Several other improvements and modifications to the basic case-based learning scheme have been proposed and should be investigated for linguistic problems. Two promising further extensions are weighting the examples in memory, and minimizing storage by keeping only a selection of examples. In example weighting, examples are differentiated according to their quality as predictors for the category of new input patterns. This quality can be based on their typicality or on their actual performance as predictors on a held-out test set. In example selection, memory is pruned by deleting those examples which are bad predictors or which are redundant.

4.2 Memory-Based Language Processing (MBLP)

Cardie (1993, 1994) addresses case-based lexical, semantic, and structural disambiguation of full sentences in limited domains, co-reference and anaphora resolution. Her KENMORE environment is presented as a general framework for knowledge acquisition for NLP using different symbolic machine learning techniques. As an instance of this general methodology, a memory-based learning approach is suggested for both morphosyntactic and semantic tagging. The architecture presupposes a corpus, a sentence analyzer, and a learning algorithm. During knowledge acquisition (training) for a specific disambiguation task (e.g. tagging), a case is created for each instance of the problem in the corpus. Each case is an example of the input-output mapping to be learned; the input part is a context describing the ambiguity, and the output part is the solution to the particular ambiguity. The examples may be produced from an annotated version of the corpus, or through human interaction. During application, the case-base is used to predict the solution to a new instance of the ambiguity given the input (the context) without intervention.

Daelemans and colleagues in Antwerp and Tilburg have applied a specific approach to MBLP (based on global feature weighting, IB1-IG, and tree indexing for efficiency, IGTREE) to a large number of NLP tasks. The algorithms they use are described and reviewed in the documentation of the freely available TIMBL package implementing a large range of memory-based algorithms (Daelemans et al. 1999). Lehnert (1987), and Weijters (1991) are early examples of memory-based learning applied to grapheme-to-phoneme conversion. Ng and Lee (1996), and Fujii, Inui, Tokunaga, and Tanaka (1998) apply memory-based techniques to the problem of Word Sense Disambiguation. Similar nearest-neighbor-inspired approaches have been applied to context-sensitive parsing (Simmons and Yu 1992), and machine translation (Hermjakob 1997). There are also memory-based approaches to text categorization and filtering (Masand, Linoff and Waltz 1992; Yang and Chute 1994; Riloff and Lehnert 1994).

Other NLP work in the memory-based tradition includes Data-Oriented Parsing (DOP) (Scha, Bod, and Sima'an 1999), who use a corpus of parsed or semantically analyzed utterances (a *Treebank*) as a representation of a person's language experience, and analyzes new sentences searching for a recombination of subtrees that can be extracted from this Treebank. The frequencies of these subtrees in the corpus are used to compute the probability of analyses. Such a method uses an annotated corpus as grammar, an approach formalized as Stochastic Tree Substitution Grammar (STSG). The advantage of STSG is that lexical information and idiomatic expressions (multi-word lexical items) can in principle play a role in finding and ranking an analysis. An approach in between DOP and more conventional memory-based methods is MBSL (Argamon, Dagan, and Krymolowski, 1998).

Work on example-based machine translation, started by Nagao (1984), is also essentially memory-based. By storing a large set of (analyzed) sentences or sentence fragments in the source language with their associated translation in the target language as examples, a new source language sentence can be translated by finding examples in memory that are *similar* to it in terms of syntactic structure and word meaning, and extrapolating from the translations associated with these examples. A more complete overview of memory-based language processing research is provided in Daelemans (1999b).

4.3 Evaluation

Advantages commonly associated with a memory-based approach to NLP include ease of learning (simply storing examples), ease of integrating multiple sources of information, and the use of similarity-based reasoning as a smoothing method for estimating low-frequency events. Especially the last property is an important theoretical issue. In language processing tasks, unlike other typical AI tasks, low-frequency events are pervasive. Due to borrowing, historical change, and the complexity of language, most data sets representing NLP tasks contain few regularities, and many subregularities and exceptions. It is impossible for inductive algorithms to reliably distinguish noise from exceptions, so non-abstracting lazy memory-based learning algorithms should be at an advantage compared to eager learning methods such as decision tree learning or rule induction: `forgetting exceptions is harmful' (Daelemans, van den Bosch and Zavrel, 1999).

Another important advantage of the memory-based approach is the flexibility of case representations: there are several types of information that can be stored in the memory base. Combined with feature weighting approaches, this flexibility offers a new approach to information source integration (data fusion) in NLP. Additional advantages include incremental learning (new cases can be added incrementally to the case bases without need for relearning), explanation capabilities (the best memory matches serve as explanations for the tagging behavior of the system), and learning and processing speed in some implementations of memory-based learning.

5. Decision Tree and Rule Induction

The *decision tree-learning* paradigm is based on the assumption that similarities between examples can be used to automatically extract decision trees and categories with explanatory and generalization power. In other words, the extracted structure can be used to solve new instances of a problem, and to explain why a performance system behaves the way it does. In this paradigm, learning is *eager*, and abstraction occurs at learning time. There are systematic ways in which decision trees can be transformed into rule sets.

Decision tree induction is a well-developed field within AI; see e.g. Quinlan (1993) for a state-of-the-art system. More ancient statistical pattern recognition work (Hunt, Marin and Stone 1966; Breiman et al. 1984) also still makes for useful reading.

5.1 Algorithm

A decision tree is a data structure in which nodes represent tests, and arcs between nodes represent possible answers to tests. Leaf nodes represent answers to problems (classes). A problem is solved, by following a path from the root node through the decision tree until a leaf node is reached. The path taken depends on the answers that a particular problem provides to the tests at the nodes. Decision tree *learning* works by repeatedly dividing the set of examples into subsets according to whether the examples in a particular subset have a feature-value pair in common, until the subsets are homogeneous, i.e., all examples in the subset have the same category. The algorithm achieves this according to the simplified recursive scheme in Figure 1.

Given a set of examples T

- If T contains one or more cases all belonging to the same class C_j, then the decision tree for T is a leaf with category C_j.
- If T contains different classes then
    - Choose a feature, and partition T into subsets that have the same value for the feature chosen. The decision tree consists of a node containing the feature name, and a branch for each value leading to a subset of T
    - Apply the procedure recursively to subsets created this way.

Figure 1. Recursive scheme for constructing decision trees

To classify new input patterns with a decision tree, start at the root node of the tree, and find the value in the input pattern for the corresponding feature. Take the branch corresponding to that value, and perform this process recursively until a leaf node is reached. The category corresponding to this leaf node is the output.

Again, we are confronted with a *feature relevance problem* in this approach. In order to obtain a concise tree with good generalization performance (i.e. a tree reflecting the structure of the domain), we have to select at each recursion of the above algorithm a test that is optimal in achieving this goal). The algorithm is non-backtracking (deterministic), and considering all trees consistent with the data is an NP-complete problem, so a reliable heuristic feature selection criterion is essential. Information-theoretic or statistical techniques maximizing homogeneity of subsets by selecting a particular feature are usually applied to this end. Several variants and extensions have been developed to the basic algorithm, e.g. for pruning (making the tree more compact by cutting off subtrees on the basis of a statistical criterion), grouping similar values of a feature into classes, making tree building incremental, etc.

5.2 Decision Tree Induction NLP

Work on parsing (including tagging) of text with decision trees was pioneered at IBM (Black et al. 1992; Magerman 1994}. SPATTER (Magerman 1995) starts from the premise that a parse tree can be viewed as the result of a series of classification problems (tagging, choosing between constituents, labeling constituents, etc.). The most probable sequence of decisions for a sentence, given a training corpus, is its most probable analysis. In the statistical decision tree technology used (based on Breiman et al. 1984), decision trees are constructed for each sub-problem in the parsing task. In such a decision tree, leaf nodes contain distributions over categories instead of a single category. E.g., in tagging, the feature associated with the root node of the decision tree might be the word to be tagged. In case its value is 'the', the category 'article' can be returned with certainty. In case its value is 'house', a test at the next level of the tree corresponds to the feature 'tag of the previous word'. In case its value is 'article', the probability distribution returned by the decision tree would be "noun (.8); verb (.2)". In practice, SPATTER uses binary trees, however. Searching for the most probable series of decisions for a sentence is done by means of stack decoder search with a breadth-first algorithm and probabilistic pruning. Schmid (1994) describes TREETAGGER, a tagger that takes basically the same approach as SPATTER, and Màrquez and Rodríguez (1998) is another approach to decision tree tagging that extracts separate decision trees for each tag (class) to be predicted. More recent work on dependency parsing (Haruno, Shirai and Ooyama 1999) for Japanese suggests the viability of the approach for parsing.

Other work using decision trees for NLP problems includes cue phrase disambiguation (Litman 1996), word sense disambiguation (Mooney 1996), and noun phrase coreference resolution (McCarthy and Lehnert 1995).

5.3 Evaluation

Decision tree models are equivalent in expressive power to interpolated n-gram models (Magerman, 1995), but whereas in n-gram models the number of parameters to be estimated grows exponentially with n, in decision-tree learning, the size of the model depends on the number of training examples, and remains constant with the number of decisions taken into account. Also, the decision tree approach automatically selects relevant context size: uninformative context positions are not used in the tree, and because of its computational properties (constant with wider context) larger contexts (corresponding to 4 or 5-grams) can initially be considered. That way, decision tree approaches are potentially more sensitive to context and therefore better equipped to solve long-distance dependencies. Another useful effect of using decision trees is greater robustness to sparse data problems.

Perhaps the most important advantage of the approach is the potential insight it may bring in what has been learned by the system. Rules, and to a lesser extent decision trees can be understandable ways of formulating the knowledge implicit in how the learning algorithm is trying to solve an NLP task. However, because of the complex interaction of regularities and exceptions, this is not always the case in practice. In Daelemans, Berck and Gillis, 1997, a linguistic theory about Dutch morphology was (post hoc) discovered by a decision tree learning algorithm.

5.4 Rule Induction and Inductive Logic Programming

It is possible to translate a decision tree into a rule set by extracting a rule for each path in a decision tree (the tests in the path constitute a conjunction of conditions, the leave node the conclusion of the rule), and then combining, and (statistically) simplifying the combined rules, and simplifying the resulting rule set by selecting a default rule (Quinlan 1993). Other strategies for extracting single rules or rule sets from data have been explored as well, e.g. for learning set-valued features (Cohen 1995). However, these rules will be *propositional*, i.e., they have the expressive power of a propositional logic whereas the formulation of some rules of language may require a complete first-order language.

A promising rule learning approach in this respect is Inductive Logic Programming (ILP, e.g., Lavrac and Dzeroski 1994), in which background knowledge, and positive and negative examples are used to induce a logic program compatible with the background knowledge and all of the positive examples, but none of the negative examples. ILP can induce first-order theories from examples, and is therefore suited for domains where propositional algorithms fail, i.e., where the task cannot be represented using (fixed-length) vectors of feature values. Language processing is a good candidate for such a domain. It is also motivated by the fact that a lot of expert linguistic knowledge is available which could be used to guide the search for learning good rules solving an NLP problem. ILP is ideally suited to use this expert knowledge through the addition of background knowledge predicates.

Part of speech tagging with ILP was thoroughly investigated by James Cussens (1997). Using a grammar as background knowledge made it possible to learn contexts more sophisticated than those allowed by propositional learners. E.g. it proved to be possible to learn rules of the type *a word followed by a noun, followed by a verb phrase, followed by an adjectival phrase cannot be tagged as a conjunction*. What constitutes a verb phrase and an adjectival phrase is defined in the background knowledge.

ILP methods have been used in learning transfer rules translating between semantic representations (Boström and Zemke 1997) (rules that transform a quasi logical form, QLF, of a sentence in English to a QLF of an equivalent sentence in Swedish, from example QLF pairs), in learning rules for morphology (Dzeroski and Erjavec 1997; Mooney and Califf 1996), and in other NLP applications, but without convincing results yet. The accuracy of these systems is often well below state of the art (this may be due to the limited size of the training sets used), and even with these small datasets, computational complexity is such that learning speed is prohibitive. The most impressive results to date are the grammar and parser learning experiments of Zelle and Mooney (1993, 1996).

6. Discussion

With the availability of only a relatively small body of empirical data and theoretical analysis on the applicability of inductive machine learning techniques to language learning (at least compared to other application areas of ML), it is too early for strong conclusions. On the empirical side, there is a hard-felt need for methodologically sound, reliable, comparative research on the application of these machine learning methods on diverse problems in language processing. As far as Quantitative Linguistics is concerned, these empirically learned generalizations could add useful insight which knowledge is used for learning specific language tasks. On the theoretical side, there is a need for more insight into the differences and similarities in how *generalization* is achieved in this area by different statistical and machine learning techniques. In the absence of this knowledge, our discussion will necessarily turn out to be preliminary and superficial. It will take the form of a number of theses.

Symbolic ML methods work. These methods have been applied successfully to a large number of NLP problems, and produce state-of-the-art accuracy and efficiency in practical systems, when compared to statistical and handcrafted knowledge-based systems. This shows that they are capable of extracting useful linguistic knowledge from data.

Symbolic ML implements a different type of statistics. Decision tree induction and memory-based learning are statistical methods, but they use a different kind of statistics than the more common maximum-likelihood and Markov modeling methods. E.g. in memory-based learning, no assumptions are made about the distribution of the data whereas most statistical techniques presuppose normal distributions. Different statistical methods have different properties that make them more or less suited for a particular type of application. If only for that reason, the applicability of all types of statistics to NLP problems should be studied thoroughly. Already from the preliminary empirical data, important advantages of these methods compared to current statistical methods suggest themselves.

- They require less training data.
- They require fewer parameters to be computed, and can therefore take into account more context.
- They provide elegant and computationally attractive solutions to the smoothing problem and to the integration of different information sources.
- Training is often much faster.

A disappointing finding is that there are few clear demonstrations that these methods can provide new insights or reusable rules from data that can consequently be combined with other knowledge sources or used.
Abstraction can be harmful. In many linguistic tasks, we have found (Daelemans, van den Bosch and Zavrel 1999) that an approach keeping complete memory of all training data provides better performance than techniques that abstract from low-frequency and exceptional events, such as rule(learning)-based systems. Neural networks and stochastic approaches are similar to rule- and decision tree induction methods in that they abstract from their experience (to a matrix of connection weights in neural networks, to a set of probabilities in stochastic approaches, and to a set of rules in rule-induction approaches), and forget about the original data on which these abstractions were based. The effect that full memory of all examples yields better generalization is probably related to the fact that natural language processing tasks such as morphosyntactic disambiguation can be characterised by the interaction of regularities, sub-regularities, and pockets of exceptions. Abstracting away from these exceptions causes a performance degradation because new similar exceptions are overgeneralized: being there is better than being probable.

Compared to the well-developed theoretical and empirical foundations of QL, the machine learning approach to linguistic knowledge discovery from language data has only just started. In all methods described, there is still a lot of room for improvement, especially in three areas: exploring variations or extensions of the basic algorithms, adding linguistic bias to the learning algorithms, and combining them with other approaches in hybrid architectures.

7. References

Aamodt, A. and E. Plaza (1994), Case-based reasoning: Foundational issues, methodological variations, and system approaches. In: *AI Communications*, 7, 39-59.

Aha, D. W. (Editor) *Lazy learning*. Dordrecht: Kluwer Academic Publishers, 1997.

Aha, D. W., D. Kibler, and M. Albert (1991), Instance-based learning algorithms. In: *Machine Learning*, 6, 37-66.

Allen, James (1995), *Natural Language Understanding*. Redwood City: The Benjamin/Cummings Publishing Company.

Argamon, S., I. Dagan, and Y. Krymolowski (1998), A memory-based approach to learning shallow natural language patterns. In: *Proceedings of the 36th annual meeting of the ACL*, 67-73, Montreal.

Atkeson, C., A. Moore, and S. Schaal (1997), Locally weighted learning. In: *Artificial Intelligence Review*, 11(1-5), 11-73.

Black, Ezra, Fred Jelinek, John Lafferty, David Magerman, Robert Mercer, and Salim Roukos (1992), Towards history-based grammars: using richer models for probabilistic parsing. In: Mitch Marcus (Editor) *Fifth DARPA Workshop on Speech and Natural Language*, San Mateo, CA: Morgan Kaufmann.

Boström, Henrik and Stefan Zemke (1997), Learning transfer rules by inductive logic programming (preliminary report), Stockholm University.

Breiman, L., J. Friedman, R. Ohlsen, and C. Stone (1984), *Classification and regression trees*, Belmont, CA: Wadsworth International Group.

Brent, M. and T. Cartwright (1996), Distributional regularity and phonotactic constraints are useful for segmentation. In: *Cognition*, 61, 93-125.

Brent, Michael (1996), Advances in the computational study of language acquisition. In: *Cognition* 61, 1-38.

Brill, E. (1992), A simple rule-based part-of-speech tagger. In: *Proceedings of the Third ACL Applied NLP*, 152-155, Trento, Italy.

Brill, Eric and Raymond J. Mooney (1998), An overview of empirical natural language processing. *The AI Magazine*, 18(4), 13-24.

Broeder, Peter and Jaap Murre (Editors), *Cognitive Models of Language Acquisition*. Cambridge University Press, 2000.

Cardie, C. (1993), A case-based approach to knowledge acquisition for domain-specific sentence analysis. In: *Proceedings of AAAI-93*, 798-803.

Cardie, C. (1994), Domain Specific Knowledge Acquisition for Conceptual Sentence Analysis. Ph.D. thesis, University of Massachusetts, Amherst, MA.

Cardie, Claire and Raymond J. Mooney (1999), Guest editors' introduction: Machine learning and natural language. In: *Machine Learning*, 11, 1-5.

Charniak, E. (1993), *Statistical Language Learning*. Cambridge, MA: The MIT Press.

Church, K. W. and R. L. Mercer (1993), Introduction to the Special Issue on Computational Linguistics Using Large Corpora. In: *Computational Linguistics*, 19,1-24.

Cohen, W. W. (1995), Fast effective rule induction. In: *Proceedings of the Twelfth International Conference on Machine Learning*, Lake Tahoe, California, San Mateo, CA: Morgan Kaufmann.

Cost, S. and S. Salzberg (1993), A weighted nearest neighbor algorithm for learning with symbolic features. In: *Machine Learning*, 10,57-78.

Cover, T. M. and P. E. Hart (1967), Nearest neighbor pattern classification. Institute of Electrical and Electronics Engineers *Transactions on Information Theory*, 13,21-27.

Cussens, James (1997), Part-of-speech tagging using Progol. In: Nada Lavrac and Saso Dzeroski, editors, *Proceedings of the 7th International Workshop on Inductive Logic Programming*, 93-108, Berlin. Springer.

Daelemans, W. (1995), Memory-based lexical acquisition and processing. In: P. Steffens (editor), *Machine Translation and the Lexicon*, Lecture Notes in Artificial Intelligence. Springer-Verlag, Berlin, pages 85-98.

Daelemans, W., A. Van den Bosch, and J. Zavrel (1999), Forgetting exceptions is harmful in language learning. In: *Machine Learning* 34, 11-41.

Daelemans, W., A. Weijters, and A. Van den Bosch (editors), *Workshop Notes of the ECML/MLnet familiarization workshop on Empirical learning of natural language processing tasks*, Prague, Czech Republic. University of Economics, 1997.

Daelemans, W., J. Zavrel, K. Van der Sloot, and A. Van den Bosch (1999), TiMBL: Tilburg Memory Based Learner, version 2.0, reference manual. Technical Report ILK-9901, ILK, Tilburg University.

Daelemans, Walter (1999a), Machine learning approaches. In: Hans van Halteren (editor), *Syntactic Wordclass Tagging*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Daelemans, Walter (editor), Memory-based Language Processing, In: *Special Issue of Journal of Experimental and Theoretical AI*, 11(3), Taylor & Francis, 1999b.

Daelemans, Walter, Peter Berck, and Steven Gillis (1997), Data mining as a method for linguistic analysis: Dutch diminutives. In: *Folia Linguistica*, XXXI(1-2).

Dresher, E. and J. Kaye (1990), A computational learning model for metrical phonology. In: *Cognition*, 32(2), 137-195.

Dzeroski, Saso and Tomaz Erjavec (1997), Induction of Slovene nominal paradigms. In: Nada Lavrac and Saso Dzeroski (editors) *Proceedings of the 7th International Workshop on Inductive Logic Programming*, 141-148, Berlin: Springer.

Elman, J. (1990), Finding structure in time. In: *Cognitive Science*, 14, 179-211.

Fix, E. and J. L. Hodges (1951), Discriminatory analysis --- nonparametric discrimination; consistency properties. Technical Report Project 21-49-004, Report No. 4, USAF School of Aviation Medicine, Randolph Field, Texas.

Fujii, Atsushi, Kentaro Inui, Takenobu Tokunaga, and Hozumi Tanaka (1998), Selective sampling for example-based word sense disambiguation. In: *Computational Linguistics*, 24(4), 573-597.

Gazdar, G. and C. Mellish (1989), *Natural Language in LISP: an introduction to computational linguistics*, Reading, MA: Addison Wesley.

Gillis, S., G. Durieux, and W. Daelemans (1995), A computational model of P&P: Dresher and Kaye (1990) revisited. In: M. Verrips and F. Wijnen (editors), *Approaches to parameter setting*, volume 4 of Amsterdam Studies in Child Language Development, 135-173.

Gillis, S., G. Durieux, W. Daelemans, and A. Van den Bosch (1993), Learnability and markedness: Dutch stress assignment. In: *Proceedings of the 15th Conference of the Cognitive Science Society* 1993, Boulder, CO, 452-457.

Goldberg, D. (1989), *Genetic Algorithms in Search, Optimization and Machine Learning.* Reading, Mass.: Addison Wesley.

Haruno, Masahiko, Satoshi Shirai, and Yoshifumi Ooyama (1999), Using decision trees to construct a practical parser. In: *Machine Learning*, 34, 131-149.

Hermjakob, Ulf and Raymond J. Mooney (1997), Learning parse and translation decisions from examples with rich context. In: *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics* (ACL '97), 482-489, Madrid, Spain, Association for Computational Linguistics.

Honavar, Vasant and Giora Slutzki (editors), *Grammatical inference: 4th international colloquium*, ICGI-98, Ames, Iowa, USA, July 12--14, 1998: proceedings, volume 1433 of Lecture Notes in Computer Science and Lecture Notes in Artificial Intelligence, New York, NY, USA. Springer-Verlag Inc., 1998.

Hunt, E. B., J. Marin, and P. J. Stone (1966), *Experiments in induction.* New York, NY: Academic Press.

Kazakov, D. and S. Manandhar (1998), A hybrid approach to word segmentation. In: D. Page (editor), *ILP98*, volume 1446 of Lecture Notes on Artificial Intelligence, 125-134, Berlin: Springer Verlag.

Kolodner, J. (1993), *Case-based reasoning*. San Mateo, CA: Morgan Kaufmann.

Koza, J.R. (1992), *Genetic Programming: on the programming of Computers by means of natural Selection*. MIT Press.

Langley, P. (1996), *Elements of machine learning*. San Mateo, CA: Morgan Kaufmann.

Lavrac, N. and S. Dzeroski (1994), *Inductive logic programming*. Chichester, UK: Ellis Horwood.

Lehnert, W. (1987), Case-based problem solving with a large knowledge base of learned cases. In: *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, pages 301-306, Los Altos, CA. Morgan Kaufmann.

Litman, Diane J. (1996), Cue phrase classification using machine learning. *Journal of Artificial Intelligence Research*, 5, 53-94.

MacWhinney, Brian and Jared Leinbach (1991), Implementations are not conceptualizations: Revising the verb learning model. In: *Cognition*, 40 (1-2), 121-150.

Magerman, D. (1995), Statistical decision tree models for parsing. In: *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics*, 276-283.

Magerman, D. M. (1994), Natural language parsing as statistical pattern recognition. Dissertation, Stanford University.

Manning, Christopher D. and Hinrich Schütze (1999), *Foundations of Statistical Natural Language Processing*. Cambridge, Massachusetts: The MIT Press.

Màrquez, L. and H. Rodríguez (1998), Part-of-speech tagging using decision trees. In: Claire Nédellec and Céline Rouveirol (editors), *Proceedings of the 10th European Conference on Machine Learning* (ECML-98), volume 1398 of LNAI, 25-36.

Masand, Briji, Gordon Linoff, and David Waltz (1992), Classifying news stories using memory-based reasoning. In: Nicholas J. Belkin, Peter Ingwersen, and Annelise Mark Pejtersen, (editors), *Proceedings of SIGIR-92, 15th ACM International Conference on Research and Development in Information Retrieval*, pages 59-65, Kobenhavn, DK. ACM Press, New York, US.

McCarthy, J. F. and W. G. Lehnert (1995), Using decision trees for coreference resolution. In*: Proceedings of the 14th IJCAI*, pages 1050-1055, Montreal, Canada, San Mateo, CA: Morgan-Kaufmann.

McClelland, J. L. and D. E. Rumelhart (editors), *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, volume 2: Psychological and Biological Models.* Cambridge, MA: The MIT Press, 1986.

Mitchell, M. (1996). *An Introduction to Genetic Algorithms*. MIT Press.

Mitchell, T. (1997), *Machine learning*. New York, NY: McGraw Hill.

Mooney, R. J. (1996), Comparative experiments on disambiguating word senses: An illustration of the role of bias in machine learning. In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP*, 82-91.

Mooney, R. J. and M. E. Califf (1996), Learning the past tense of English verbs using inductive logic programming. In: S. Wermter, E. Riloff, and G. Scheler, (editors), *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*. Springer, Berlin, 370-384.

Nagao, M. (1984), A framework of a mechanical translation between Japanese and English by analogy principle. In: A. Elithorn and R. Banerji (editors), *Artificial and human intelligence.* Amsterdam: North-Holland, 173-180.

Natarajan, B.K. (1991), *Machine Learning: A Theoretical Approach*. San Mateo, CA: Morgan Kaufmann.

Ng, Hwee Tou and Hian Beng Lee (1996), Integrating multiple knowledge sources to disambiguate word sense: An exemplar-based approach. In: *Proceedings of the 34th meeting of the Association for Computational Linguistics.*

Niyogi, P. and R. C. Berwick (1994), A Markov language learning model for finite parameter spaces. In: *Proceedings of 32nd meeting of Association for Computational Linguistics,*

Quinlan, J.R. (1993), *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Reilly, R. G. and N. E. Sharkey (editors), *Connectionist Approaches to Natural Language Processing*. Hillsdale, NJ: Lawrence Erlbaum Associates, 1992.

Resnik, Philip (1996), Selectional constraints: an information-theoretic model and its computational realization. In: *Cognition*, 61: 127-159.

Riesbeck, C. and R. Schank (1989), *Inside Case-Based Reasoning*. Northvale, NJ: Erlbaum.

Riloff, Ellen and Wendy Lehnert (1994), Information extraction as a basis for high-precision text classification. In: *ACM Transactions on Information Systems*, 12(3), 296-333.

Scha, Remko, Rens Bod, and Khalil Sima'an (1999), A memory-based model of syntactic analysis: data-oriented parsing. In: *Journal of Experimental and Theoretical Artificial Intelligence*, 11, 409-440.

Schmid, H. (1994), Probabilistic part-of-speech tagging using decision trees. In: *Proceedings of the International Conference on New Methods in Language Processing.*

Sharkey, N. (1992), *Connectionist Natural Language Processing*. New York: Weather Hill.

Simmons, Robert F. and Yeong-Ho Yu (1992), The acquisition and use of context-dependent grammars for English. In: *Computational Linguistics*, 18(4), 391-418, December.

Siskind, Jeffrey (1996), A computational study of cross-situational techniques for learning word-to-meaning mappings. In: *Cognition*, 61, 39-91.

Skousen, R. (1989), *Analogical modeling of language*. Dordrecht: Kluwer Academic Publishers.

Skousen, R. (1992), *Analogy and Structure*. Dordrecht: Kluwer Academic Publishers.

Stanfill, C. and D. Waltz (1986), Toward memory-based reasoning. In: *Communications of the ACM*, 29(12), 1213-1228, December.

Vapnik, V. N. (1995), *The Nature of Statistical Learning Theory*. New York: Springer.

Weijters, A. (1991), A simple look-up procedure superior to Nettalk? In: *Proceedings of the International Conference on Artificial Neural Networks - ICANN-91*, Espoo, Finland.

Weiss, S. and C. Kulikowski (1991), *Computer systems that learn*. San Mateo, CA: Morgan Kaufmann.

Wermter, Stefan, Ellen Riloff, and Gabriele Scheler (editors), Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing. Lecture Notes in Artificial Intelligence, volume 1040. Berlin: Springer, 1996.

Wettschereck, D., D. W. Aha, and T. Mohri (1997), A review and comparative evaluation of feature-weighting methods for a class of lazy learning algorithms. In: *Artificial Intelligence Review* 11, 273-314.

Yang, Yiming and Christopher G. Chute (1994), An example-based mapping method for text categorization and retrieval. In: *ACM Transactions on Information Systems,* 12(3), 252-277, July. Special Issue on Text Categorization.

Zelle, J. M. and R. J. Mooney (1993), Learning semantic grammars with constructive inductive logic programming. In: *Proceedings of the 11th National Conference on Artificial Intelligence*, 817-822, Washington, D.C., July. AAAI Press/MIT Press.

Zelle, J. M. and R. J. Mooney (1996), Comparative results on using inductive logic programming for corpus-based parser construction. In: S. Wermter, E. Riloff, and G. Scheler, (editors), *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing*. Springer, Berlin, 355-369.