Instance-Family Abstraction in Memory-Based Language Learning

Antal van den Bosch
ILK / Computational Linguistics
Tilburg University
The Netherlands
Antal.vdnBosch@kub.nl

Abstract

Memory-based learning appears relatively successful when the learning data is highly disjunct, i.e., when classes are scattered over many small families of instances in instance space, as in many language learning tasks. Abstraction over borders of disjuncts tends to harm generalization performance. However, careful abstraction in memory-based learning may be harmless when it preserves the disjunctivity of the learning data. We investigate the effect of careful abstraction in a series of language-learning task studies, and a small benchmark-task study. We find that when combined with feature weighting or value-distance metrics, careful abstraction, as implemented in the new FAMBL algorithm, can equal the generalization accuracies of pure memory-based learning, while attaining fair levels of memory compression.

1 INTRODUCTION

Memory-based learning (Stanfill and Waltz, 1986; Aha, Kibler, and Albert, 1991; Aha, 1997) has been studied for some time now as an approach to learning language processing tasks. It is found by various studies to be successful, attaining adequate to excellent generalization accuracies on realistic, complex tasks as different as hyphenation, semantic parsing, part-of-speech tagging, morphological segmentation, and word pronunciation (Daelemans and Van den Bosch, 1992; Cardie, 1994; Cardie, 1996; Daelemans et al., 1996; Van den Bosch, 1997). Recent studies (Van den Bosch, 1997; Daelemans, Van den Bosch, and Zavrel, 1999) provide indications that forgetting (parts of) task instances during learning tends to hinder generalization accuracy of the trained classifiers, especially when these instances are estimated to be exceptional. Learning algorithms that do not forget anything about the learning material, i.e., pure memory-based learning algorithms, are found to obtain the best accuracies for the tasks studied when compared to decision-tree or edited memory-based learning algorithms.

However, these findings still leave room for the hypothesis that abstraction of a more careful type may be an equal alternative to pure memory-based learning. The topic of this paper is to perform empirical tests on language learning tasks to collect indications for the efficacy of careful abstraction in memory-based language learning, in comparison with pure memory-based learning.

The paper is structured as follows. Section 2 summarises methods for abstraction in memory-based learning; it reviews existing approaches, and presents FAMBL, a new memory-based learning algorithm that abstracts carefully by merging (families of) instances in memory. In Section 3 a number of memory-based learning algorithms performing careful abstraction are applied to six realistic large-scale language learning tasks. The section also briefly discusses to what extent language learning tasks differ from typical benchmark tasks. In Section 4, the efficacy of careful generalization over families of instances is discussed.

2 BACKGROUND: ABSTRACTION IN MEMORY-BASED LEARNING

Memory-based learning, also known as instance-based, example-based, lazy, case-based, exemplar-based, locally weighted, and analogical learning (Stanfill and Waltz, 1986; Aha, Kibler, and Albert, 1991; Salzberg, 1991; Kolodner, 1993; Aha, 1997; Atkeson, Moore,

and Schaal, 1997), is a class of supervised inductive learning algorithms for learning classification tasks. Memory-based learning treats a set of labeled (preclassified) training instances as points in a multidimensional feature space, and stores them as such in an *instance base* in memory (rather than performing some abstraction over them). New (test) instances are classified by matching them to all instances in the instance base, calculating with each match the *distance* between the new instance and a memory instance. The memory instances with the smallest distances are collected, and the classifications associated with these nearest neighbors are merged and extrapolated to assign a classification to the new instance.

Early work on the k-NN classifier pointed at advantageous properties of the classifier in terms of generalization accuracies, under certain assumptions, because of its reliance on full memory. However, the trade-off downside of full memory is computational inefficiency of the classification process, as compared to parametric classifiers that do abstract from the learning material. Therefore, several early investigations were performed into *editing* methods: finding criteria for the removal of instances from memory (Hart, 1968; Gates, 1972) without harming classification accuracy. Other studies on editing also explored the possibilities of detecting and removing noise from the learned data, so that classification accuracy might even improve (Wilson, 1972; Devijver and Kittler, 1980).

The renewed interest in the k-NN classifier from the late 1980s onwards in machine learning (Stanfill and Waltz, 1986; Aha, Kibler, and Albert, 1991; Salzberg, 1991) caused several new implementations of ideas on criteria for editing, but also other approaches to abstraction in memory-based learning emerged. Now, three types of careful abstraction in memory-based learning may be distinguished: (1) **Editing** (Hart, 1968; Wilson, 1972; Aha, Kibler, and Albert, 1991): removing instances according to a classification-related utility threshold they do not reach; (2) **Oblivious (partial) decision-tree abstraction** (Daelemans, Van den Bosch, and Weijters, 1997): compressing instances in the instance base into decision-tree paths; and (3) **Carefully merged instances** (Salzberg, 1991; Domingos, 1996): merging multiple instances in single generalized instances (or hyperrectangles). While individual instances are usually represented by propositional conjunctions of atomic feature values, merged instances can be conjunctions of *disjunctions of* feature values as in NGE (Nested Generalized Exemplars) (Salzberg, 1991), or rules with wildcards as in RISE (Rule Induction from a Set of Exemplars) (Domingos, 1996).

As noted in the introduction, recent studies have shown that editing in memory-based learning (1) and decision-tree induction (2), applied to language learning tasks, yield lower generalization accuracies as compared to pure memory-based learning tasks. We have argued earlier, supported by empirical evidence, that this is due to the fact that both methods tend to ignore small disjuncts (Daelemans, Van den Bosch, and Zavrel, 1999). In contrast, merging instances as in NGE and RISE is specifically designed to preserve disjuncts when appropriate. The experiments described in this paper are performed to investigate whether algorithms of this type indeed are able to maintain the level of accuracy of pure memory-based learning on langauge learning tasks. Apart from performing experiments with RISE and NGE, we introduce FAMBL, a new algorithm that combines some of the ideas that also underly NGE and RISE.

2.1 FAMBL: MERGING INSTANCE FAMILIES

FAMBL, for *FAMily-Based Learning*, is a new algorithm that constitutes an alternative approach to careful abstraction over instances. A first account of FAMBL is given in Van den Bosch (1999). In the current paper, we report on extended studies with FAMBL, among which a comparitive study on standard machine learning benchmark tasks. The core idea of FAMBL is to transform an instance base into a set of *instance family expressions*. An instance family expression is a hyperrectangle, but the procedure for merging instances differs from that in NGE or in RISE. First, we outline the ideas and assumptions underlying FAMBL. We then describe the learning algorithm.

The value of k in k-NN classification determines how many of nearest neighbours are used for extrapolating their (majority) classification to a new instance. Fixing k ignores the fact that an instance is often surrounded in instance space by a number of instances of the same class that may well be larger or smaller than k. We refer to such variable-sized set of same-class nearest neighbours as an instance's *family*. The number and sizes of families in a data set reflect the *disjunctivity* of the data set: the degree of scatteredness of classes into clusters. Many types of language data appear to be quite disjunct (Daelemans, Van den Bosch, and Zavrel, 1999).

Figure 1 illustrates how FAMBL determines the fam-

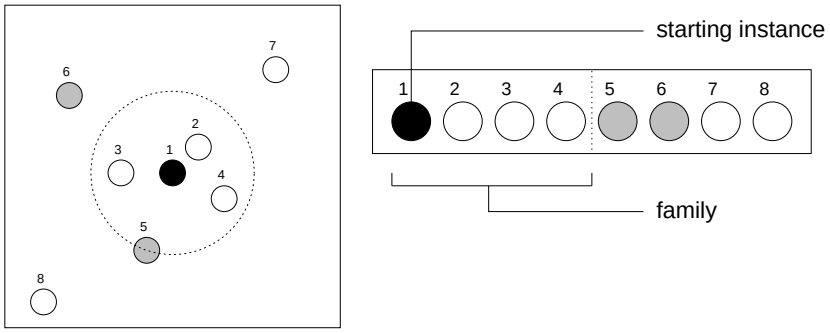

Figure 1: An example of a family in a two-dimensional instance space (left). The family, at the inside of the dotted circle, spans the focus instance (black) and the three nearest neighbours labeled with the same class (white). When ranked in the order of distance (right), the family boundary is put immediately before the first instance of a different class (grey).

ily of an instance, using a simple two-dimensional example instance space. All nearest neighbours of a randomly-picked starting instance (marked by the black dot) are searched and ranked in the order of their distance to the starting instance. Although there are five instances of the same class in the example space, the family of the starting instance contains only three, since its fourth-nearest instance is of a different class. Once a family is determined, FAMBL converts it to a *family expression*, by merging all instances belonging to that family simultaneously. When merged instances differ in values at a feature, the resultant family expression lists both values as disjunctions at that feature. In contrast with NGE, (i) family expressions are created in one operation, rather than incrementally; (ii) a family is abstracted only once and is not merged later on with other instances or family expressions.

The FAMBL algorithm has a learning component and a classification component. The learning component, in which all families in an instance base are determined and converted to expressions, is composed of two stages: a *probing* stage and a *family extraction* stage. The probing stage is a preprocessing stage to the actual family extraction as outlined above. The reason for preprocessing is visualized in Figure 2. The random selection of instances to be a starting point for family creation can be quite unfortunate. When, for example, the middle instance in the left part of Figure 2 is selected first, a seven-instance family is formed with relatively large within-family distances. Moreover, three other instances that are actually quite close to members of this big family become isolated and are necessarily extracted later on as single-instance families. The situation in the right part of Figure 2 displays a much more desirable situation, in which the space is more evenly divided between only two families instead of four.

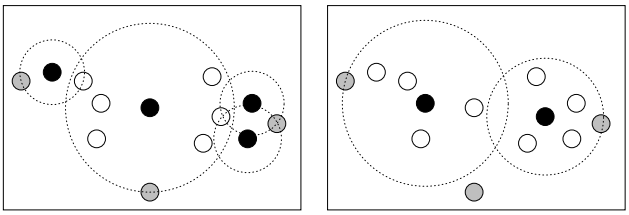

Figure 2: Illustration of the need for the preprocessing stage in FAMBL. The left figure shows a large family surrounding the middle starting-point instance (black dot), forcing the remaining three instances to be their own family. The right figure shows the same space, but with two other starting points, displaying a more evenly divided space over two families. White-dot instances are of the same class as the starting points; grey-dot instances have a different class.

In the probing stage, all families are extracted randomly and straightforwardly, while records are maintained of (i) the size of each family, and (ii) the average distance between the starting instance of each family and the other instances in the family. When all instances are captured in families, the mean and the median of both records are computed, and *both medians are used as threshold values for the second, actual family extraction phase.* This means that in the family extraction phase, no family is extracted that has more members than the probed median number of members, and no family is extracted that has an average distance from the starting instance to the other family members larger than the probed median value. Thus, the actual family extraction phase applies ex-

Procedure FAMBL FAMILY-EXTRACTION PHASE:

Input: A training set TS of instances $I_{1\ldots n}$, each instance being labeled with a family-membership flag set to $FALSE$

Output: A family set FS of family expressions $F_{1\ldots m}$, $m \leq n$

$i = f = 0$

1. Randomize the ordering of instances in TS
2. While not all family-membership flags are $TRUE$, Do
   - While the family-membership flag of I_i is $TRUE$ Do increase i
   - Compute NS, a ranked set of nearest neighbors to I_i with the same class as I_i, among all instances with family-membership flag $FALSE$. Nearest-neighbor instances of a different class with family-membership flag $TRUE$ are still used for marking the boundaries of the family.
   - Compute the number of members in the new virtual family: $|NS| + 1$
   - Compute the average distance of all instances in NS to I_i: $A_{NS,I}$
   - While $((|NS + 1| > M_1) OR (A_{NS,I} > M_2))$ Do remove the most distant family member to I in NS
   - Set the membership flags of I_i and all remaining instances in NS to $TRUE$
   - Merge I_i and all instances in NS into the family expression F_f and store sthis expression along with a count of the number of instance merged in it
   - $f = f + 1$

Figure 3: Schematized overview of the family-extraction phase in FAMBL.

tra careful abstraction, under the assumption that it is better to have several medium-sized, adjacent families of the same class than one big family overlapping the medium ones except for some adjacent boundary instances that get isolated.

Figure 3 summarizes the procedure followed by the actual family extraction stage. In short, family extraction is performed by repeatedly selecting a starting point, which is an instance that is not merged yet into a family, and building a family around it. This procedure repeats until all instances are in a family.

After learning, the original instance base is discarded, and further classification is based only on the set of family expressions yielded by the family-extraction phase. Classification in FAMBL works analogously to classification in pure memory-based learning, and classification in NGE: a match is made between a new test instance and all stored family expressions. When a family expression records a disjunction of values for a certain feature, matching is perfect when one of the disjunctive values matches the value at that feature in the new instance. Merged feature-value counts are summed for each feature-value match, to be used in case of ties: when two or more family expressions of different classes match equally well with the new instance, the expression is selected with the highest summed occurrence of matched features. When the tie remains, the class is selected that occurs the most frequently in the complete family expression set.

We conclude our description of the FAMBL algorithm by noting that FAMBL allows for the inclusion of informational abstraction in the form of feature-weighting, instance-weighting and value-difference metrics. For comparisons with other algorithms, as described in the next section, we have included these metrics as options in FAMBL.

3 EFFECTS OF CAREFUL ABSTRACTION IN LANGUAGE LEARNING AND BENCHMARK TASKS

As stated in the introduction, our primary goal is to investigate the effects of careful instance-family abstraction in memory-based learning of language processing tasks. In this section we report on experiments in which FAMBL, RISE, and NGE, in comparison with the pure memory-based learning algorithm IB1 (Aha, Kibler, and Albert, 1991; Daelemans and Van den Bosch, 1992), are applied to six language learning tasks, measuring the effects of their careful abstraction on generalization accuracy. First, we briefly introduce the six tasks investigated in this study. Table 1 lists the numbers of instances, feature values, and classes of the data sets of the selected six tasks.

Grapheme-phoneme conversion (henceforth referred to as GP) is the mapping between the spelling of a word and its phonemic transcription. We define the task as the mapping of fixed-sized instances representing parts of English words to a class representing the phoneme of the instance's middle letter. An example instance and its classifications is "_ _ h e a r t s _", mapping to class /A:/, which denotes an elongated short 'a'-sound to which the middle letter 'a' maps. The data used in the experiments described here are derived from the CELEX lexical data base of English (Baayen, Piepenbrock, and van Rijn, 1993).

Grapheme-phoneme conversion combined with stress assignment (henceforth GS) is similar to the GP task, but differs in two respects: (i) the windows only span seven letters, and (ii) the class represents a combined phoneme

and a stress marker. Except for the data (derived from CELEX), the task is the same as the NETtalk task (Sejnowski and Rosenberg, 1987). See Van den Bosch (1997) for more details.

Morphological segmentation (henceforth MS) is the segmentation of words into labeled morphemes. Each instance represents a window snapshot of a word of nine letters. Its class represents the presence or absence of a morpheme boundary immediately before the middle letter. If present, it also encodes the type of morpheme starting at that position. See Van den Bosch, Daelemans, and Weijters (1996) for more details.

Base-NP chunking (henceforth NP) is the segmentation of sentences into non-recursive NPs. Veenstra (1998) used the Base-NP tag set as presented in (Ramshaw and Marcus, 1995): *I* for inside a Base-NP, *O* for outside a Base-NP, and *B* for the first word in a Base-NP following another Base-NP. See Veenstra (1998) for more details, and Daelemans, Van den Bosch, and Zavrel (1999) for a series of experiments on the particular data set also used here.

PP attachment (henceforth PP) is the attachment of a PP in the sequence VP NP PP (VP = verb phrase, NP = noun phrase, PP = prepositional phrase). The data consists of four-tuples of words, extracted from the Wall Street Journal Treebank. From the original data set, used by Ratnaparkhi, Reynar, and Roukos (1994), Collins and Brooks (1995), and Zavrel, Daelemans, and Veenstra (1997), Daelemans, Van den Bosch, and Zavrel (1999) took the train and test set together to form the particular data also used here.

Part-of-speech tagging (henceforth POS) is the disambiguation of syntactic classes of words in particular contexts. We assume a tagger architecture that processes a sentence from a disambiguated left to an ambiguous right context , as described in (Daelemans et al., 1996). The data set for the part-of-speech tagging task, extracted from the LOB corpus, contains 1,046,151 instances.

It was not possible to run experiments with both RISE and NGE on the maximal data set sizes (cf. Table 1) due to the large processing and memory demands of their current implementations, in relation to the computational resources available[1]. Therefore, we divided our experiments in three batches, which are described in Subsections 3.1, 3.2, and 3.3:

- In Subsection 3.1, experiments are described that deal with a 10% subset selection of the GP dataset. All algorithms (including RISE and NGE) are involved, using several combinations of feature weighting, instance weighting, and value distance metrics for the case of IB1 and FAMBL.
- In Subsection 3.2, FAMBL with gain-ratio feature weighting, RISE, and IB1 with gain-ratio feature weighting are applied to growing data sets of the GP, GS, and MS tasks, from 0.1% subsets via 1% and 10% portions to the full data sets. These experiments provide indications of the differences between RISE and FAMBL.
- In Subsection 3.3, FAMBL and IB1 are applied to the full-sized six data sets.

To make the link between language learning and machine learning in general more specific, we also report on a series of experiments in which FAMBL is applied to a number of well-known benchmark tasks, in Subsection 3.4. The results of these analyses specifically serve to show the difference in data characteristics between language learning and typical benchmark tasks.

3.1 CAREFUL ABSTRACTION VERSUS IB1 ON A GRAPHEME-PHONEME SUBSET

First, we performed a series of experiments concerning the application of a range of careful-abstracting methods to grapheme-phoneme conversion (GP). From an original instance base of 77,565 word-pronunciation pairs extracted from the CELEX lexical data base of English (Baayen, Piepenbrock, and van Rijn, 1993) we created ten equal-sized data sets each containing 7,757 word-pronunciation pairs. Using windowing and partitioning of this data in 90% training and 10% test instances, ten training and test sets are derived containing on average 60,813 and 6761 instances, respectively. These are token counts; in the training sets, 54,295 instance types occur (on average). One experiment consists of applying one algorithm to each of the ten training sets, and a test on each of the respective test sets. Apart from the careful abstractors RISE, NGE, and FAMBL, we include the pure memory-based learning algorithm IB1(-GR) (Aha, Kibler, and Albert, 1991; Daelemans and Van den Bosch, 1992) in the comparison. Each experiment yields (i) the mean generalization accuracy, in percentages correctly classified test instances, (ii) a standard deviation on this mean, and (iii) a count on the number of items in memory, i.e., (generalized) instances. Table 2 lists these

[1] Experiments were run on a dual-Pentium II 300 MHz machine, running Solaris, with 512 Mb of RAM.

TASK	# FEAT.	# VALUES OF FEATURE 1	2	3	4	5	6	7	8	9	10	11	# CLASS	# DATA SET INSTANCES
GP	9	42	42	42	42	41	42	42	42	42			61	675,745
GS	7	42	42	42	41	42	42	42					159	675,745
MS	9	42	42	42	42	41	42	42	42	42			2	573,544
PP	4	3,474	4,612	68	5,780								2	23,898
NP	11	20,231	20,282	20,245	20,263	86	87	86	89	3	3	3	3	251,124
POS	5	170	170	498	492	480							169	1,046,151

Table 1: Specifications of the six investigated language learning tasks: numbers of features, values per feature, classes, and instances.

experimental outcomes for all algorithms tested. As some of these algorithms use (combinations of) metrics for weighting or value-distance estimation: gain ratio (GR) (Quinlan, 1993), class-prediction strength (CPS, or CPS with Laplace correction as in RISE), or value-difference metrics (MVDM, or SVDM in RISE), we have marked the use of such metrics explicitly in the table.

ALGORITHM	METRICS GR	CPS	VDM	GEN. ACC. (%)	±	# MEMORY ITEMS
FAMBL	x		x	89.0	0.6	31,948
IB1	x		x	88.9	0.6	54,294
RISE		x	x	88.9	0.6	20,252
IB1	x			88.8	0.6	54,294
FAMBL	x			88.8	0.6	35,141
FAMBL			x	88.0	0.7	34,052
IB1			x	87.9	0.7	54,294
IB1				78.1	0.8	54,294
FAMBL				72.3	0.8	31,889
NGE		x		61.8	0.9	25,627

Table 2: Overview of generalization accuracies and memory usage obtained with pure memory-based learning and careful abstraction methods.

The results indicate a group of five best-performing algorithms that, in pair-wise comparisons using one-tailed t-tests, do not perform significantly different: (i) FAMBL with GR and MVDM, (ii) IB1 (with GR and MVDM), (iii) RISE (with CPS and SVDM), (iv) IB1 (with GR), and (v) FAMBL (with GR). A general property of this group is that it contains all algorithms that employ GR feature weighting.

Within the group of five best-performing algorithms, two are careful abstractors: RISE and FAMBL. As compared to the 54,295 instance types that are all stored in pure memory-based learning, RISE obtains an item compression of 62.7%. FAMBL with GR and MVDM compresses less: 41.2%. NGE with its default CPS exemplar weighting performs significantly worse than the unweighted IB1 and FAMBL, and also worse than FAMBL with CPS. This suggests that using CPS in isolation hampers performance on the GP task, while the incrementality of NGE is an additional cause for its low performance on this task.

3.2 COMPARING FAMBL, RISE, AND IB1 ON GROWING DATA SETS

It is conceivable that data set size matters in the differences between careful abstractors and pure memory-based learning. Adding more instances to a data set may alter the average number, size and within-family distance of the disjuncts in that data set. The outcome of careful abstraction may be affected by this change at a different scale than it may affect pure memory-based learning. In this subsection we report on experiments in which the careful abstractors FAMBL with GR weighting, RISE, and the pure memory-based learning algorithm IB1 with GR weighting, are applied to data sets of increasing size, of the three morpho-phonological tasks GP, GS, and MS (the other tasks could not be performed with RISE due to their high numbers of feature values, and the dependency of RISE on storing SVDM matrices in memory).

From each dataset, we extracted three smaller datasets, containing 1/1000th, 1/100th, and 1/10th of the words in the original data set, respectively. Each data set and the complete data sets themselves were used in a 10-fold CV experiment with IB1-IG, FAMBL, and RISE. RISE was not applied to the full data sets due to the computational limitations mentioned earlier.

The generalization accuracies yielded by the three learning algorithms on the three tasks are listed in Table 3. On a global level, they show that generaliza-

TASK	PORTION (%)	GENERALIZATION ACCURACY (%) IB1-GR	FAMBL-GR	RISE
GP	1/1000	63.1 ± 4.3	63.1 ± 4.3	55.0 ± 7.0
	1/100	79.1 ± 1.3	79.3 ± 1.4	80.2 ± 1.4
	1/10	88.8 ± 0.6	88.8 ± 0.6	88.9 ± 0.6
	1/1	97.4 ± 0.1	97.4 ± 0.1	—
GS	1/1000	52.0 ± 7.3	52.0 ± 7.3	46.2 ± 6.7
	1/100	69.6 ± 1.7	69.6 ± 1.7	72.8 ± 1.3
	1/10	81.8 ± 0.8	81.7 ± 0.8	82.3 ± 0.8
	1/1	93.5 ± 0.2	93.2 ± 0.2	—
MS	1/1000	77.5 ± 6.1	77.1 ± 5.6	85.5 ± 5.1
	1/100	87.3 ± 1.2	86.7 ± 1.2	89.9 ± 1.1
	1/10	93.0 ± 0.4	92.7 ± 0.4	93.4 ± 0.3
	1/1	98.0 ± 0.1	97.8 ± 0.1	—

Table 3: Overview of average generalization accuracies obtained with IB1-GR, FAMBL-GR and RISE on increasing portions of the GP, GS, and MS data sets. Generalization accuracy denotes the percentage of correctly classified test instances. '—' means that the experiment could not be performed.

TASK	PORTION (%)	# MEMORY ITEM COMPR. FAMBL-GR	RISE
GP	1/1000	0.0	2.9
	1/100	22.8	44.4
	1/10	35.2	62.7
	1/1	50.7	—
GS	1/1000	0.0	7.3
	1/100	19.1	38.8
	1/10	26.1	49.1
	1/1	35.7	—
MS	1/1000	14.4	23.9
	1/100	27.8	37.7
	1/10	47.7	49.5
	1/1	53.7	—

Table 4: Overview of average reduction in the number of stored memory items (families or rules) by FAMBL-GR and RISE, respectively, on increasing portions of the GP, GS, and MS data sets. '—' means that the experiment could not be performed.

tion accuracies do not differ much between algorithms at each data set size. There is reason to believe that RISE, when trained and tested on the full data sets, would perform similarly to the other three algorithms. One-tailed t-tests indicate that RISE has a significant advantage over the other algorithms on some reduced data sets (it is significantly better with $p < 0.01$ than the second-best algorithm in the 1/100th GS task, with $p < 0.05$ in the 1/100th MS task, and with $p < 0.01$ in the 1/10th MS task). The overall best performances are obtained with IB1-IG on the complete data sets (cf. next subsection).

Table 4 lists the memory item compression levels (in percentages) obtained by the two careful abstractors as opposed to IB1-IG. The unit of counting is the abstract notion of memory item, which is an instance type (not instance token) in IB1-IG, a family expression in FAMBL and a rule in RISE. First, larger data sets allow for more compression. All careful abstractors are able to reduce the number of generalized instances (families, rules, end nodes) further, relative to the total number of instance types, when the number of training instances increases: apparently, more of the learning material can be merged together when more data of the same task is available. Second, RISE compresses more than FAMBL on the tested data set sizes. This is at the cost of considerably longer learning times (in the order of 10 to 100 times longer than those of FAMBL), due to the possibly large number (in observed practice, between 5 and 15) of wrapped rule induction cycles needed to arrive at the final rule set.

3.3 COMPARING FAMBL AND IB1 ON SIX FULL-SIZED LANGUAGE LEARNING TASKS

For each task, FAMBL with GR feature weighting is compared with IB1 with GR feature weighting (i.e., FAMBL's pure memory-based counterpart). Table 5 lists the generalization accuracies obtained in these comparisons, on the five tasks. The results of IB1-GR on GS, NP, PP, and POS are reproduced from (Daelemans, Van den Bosch, and Zavrel, 1999). IB1-GR is significantly more accurate than FAMBL on the MS, GS, and PP tasks. Nevertheless, at least for the MS and GS tasks, the reported differences of a few tenths of percentages are negligible from a language-engineering standpoint. Still, the results show that FAMBL's careful abstraction is just not careful enough on these tasks.

We also monitored for all experiments the number of families that FAMBL probed and extracted. Table 6 displays these results averaged over the ten experiments performed on each task. The table also displays the percentage of compression over the number of memory items (instance types vs. family expressions), reported for the probing stage as well as the family stage.

In the extraction phase, compression (the percentage

TASK	IB1-GR %	±	>FAMBL?	FAMBL-GR %	±
GP	97.4	0.1		97.3	0.1
GS	93.5	0.2	**	93.2	0.2
MS	98.0	0.1	***	97.8	0.1
NP	98.1	0.1		98.0	0.1
PP	83.5	1.2	**	81.8	1.1
POS	97.9	0.1		97.8	0.0

Table 5: Generalization accuracies (percentages correctly classified test instances, with standard deviations) of IB1-GR and FAMBL on the six language learning tasks. Asterisks denote the outcomes of one-tailed t-tests, denoting a significantly better accuracy of IB1-IG compared to FAMBL. '**' denotes $p < 0.01$; '***' denotes $p < 0.001$.

TASK	PROBING STAGE #	% ITEM COMPR.	EXTRACTION STAGE #	% ITEM COMPR.
GP	31,224	90.5	162,954	50.7
GS	37,457	83.2	153,441	31.0
MS	18,783	93.4	131,776	53.7
NP	5238	97.7	59,376	72.4
PP	157	99.3	6414	70.1
POS	11,397	98.0	140,488	75.1

Table 6: Measurements on FAMBL output during the probing stage (left) and the family extraction stage (right) on each of the six learning tasks, of the number of families and the item compression compared to the original instance base.

reduction on the number of items in memory, from instances in pure memory-based learning to family expressions) ranges from 31.0% with the GS task to 75.1% with POS, which is considerable. The lowest compression is obtained with GS, on which FAMBL did not outperform IGTREE. This suggests that the GS task data has properties that FAMBL is less prepared to handle adequately. We have two working hypotheses on what these properties might be: First, the GS data is very disjunct: FAMBL detects a relatively high number of families during probing and family extraction. The random selection of starting points for family extraction, although heuristically patched with the preprocessing of the probing phase, may still lead to unwanted effects with such high disjunctivity. Second, as is the case with IB1, FAMBL tends to blur *feature interaction*: it allows the combination of feature values that never occurred in that constellation in the learning material, while for some tasks, including GS, this generalization may be unwanted.

3.4 FAMBL AND BENCHMARK TASKS

By design, FAMBL, augmented with GR feature weighting, may have a bias to language-like data. Only a small part of typical benchmark data sets is language-related, so a direct reference to well-known benchmark tasks cannot be made. To make the difference between language data and typical benchmark data more explicit, we applied FAMBL to a selection of benchmark tasks with symbolic feature values from the UCI repository (Blake, Keogh, and Merz, 1998), using 10-fold cross-validation experiments. Table 7 shows the average results from FAMBL-GR's probing phase on the number of probed families and the clusteredness of the data. Clusteredness is defined as the number of families per class, averaged over classes, weighted by their frequency. The difference in data set sizes is obvious: benchmark data sets are usually much smaller than the language data sets. Much less families are probed with the benchmark data, and furthermore, classes are scattered in much less disjunct clusters (families) with the benchmark data as compared to the language data, the PP data set being an exception.

TASK	DATA SET SIZE	# PROBED FAMILIES	CLUSTER-EDNESS
GP	675,745	31,224	1908
MS	573,544	18,783	8435
GS	675,745	37,457	1693
NP	251,124	5238	2413
PP	23,898	157	78
POS	1,046,152	11,397	479
audiology	226	72	7
kr vs kp	3,198	137	69
mushroom	8,124	23	11
nursery	12,961	682	198
soybean-l	683	90	9
splice	3,190	334	128
tic-tac-toe	958	161	84
votes	435	38	19

Table 7: Comparison of data set size, average numbers of probed families, and clusteredness, between the language data and a selection of UCI benchmark data, as measured in FAMBL's probing phase (averaged over 10-fold cross validation experiments).

Table 8 lists the average generalization accuracies pro-

duced by IB1-GR, FAMBL-GR, and FAMBL-GR-MVDM, a variation of FAMBL that combines GR feature weighting and the MVDM value difference metric (Cost and Salzberg, 1993), on the selected UCI benchmark data sets. We have added the generalization accuracies obtained by C4.5 and C4.5RULES (Quinlan, 1993), with default parameter settings, on the same data sets. Both C4.5 and C4.5RULES employ GR feature weighting, as a means to decide on feature test ordering, and both represent non-careful abstraction.

	GENERALIZATION ACCURACY (%)				
TASK	IB1-GR	FAMBL-GR	FAMBL-GR-MVDM	C4.5	C4.5-RULES
audiology	79.3	79.3	79.8	78.0	78.4
kr vs kp	97.7	97.1	97.3	99.5	99.6
mushroom	100.0	100.0	100.0	100.0	100.0
nursery	94.7	94.6	98.6	97.8	98.3
soybean-l	91.8	91.8	93.9	91.2	92.0
splice	91.9	86.6	94.4	94.0	92.8
tic-tac-toe	89.5	85.7	91.8	84.7	98.9
votes	93.8	94.3	94.3	95.9	95.5

Table 8: Average (10-fold) generalization accuracies of IB1-GR, FAMBL-GR, FAMBL-GR-MVDM, C4.5, and C4.5RULES, on eight UCI benchmark tasks.

In view of the earlier discussions on running significance tests on algorithmic performance differences obtained from applications to benchmark tasks (Salzberg, 1997; Dietterich, 1998), we do not intend to base any claims on the outcomes of such tests; we merely state that the performances of FAMBL-GR and FAMBL-GR-MVDM are mostly at the level of at least one of the generalization accuracies of IB1-GR, C4.5 or C4.5RULES for each of the data sets. A notable deviation is that FAMBL-GR performs significantly worse than all other algorithms on the 'splice' (gene splicing point detection) task, according to one-tailed t-tests with $p < 0.05$. FAMBL-GR-MVDM performs better on 'splice', and displays quite favorable performance on other tasks ('nursery'; 'soybean-l') as well. In sum, FAMBL appears to be applicable in a successful manner to at least some benchmark data sets, but more empirical work is needed to chart the causes of the differences found, and to elicit the general areas of applicability of FAMBL.

4 DISCUSSION

We have reported on the application of careful-abstraction methods in memory-based learning that abstract over families of instances, to language learning tasks and a small selection of non-linguistic benchmark tasks. In a first study, on learning grapheme-phoneme conversion from a moderately-sized data set, we found that the careful abstractors RISE and FAMBL were able to equal the generalization accuracy of their pure memory-based counterpart IB1-GR. All best-performing algorithms implement (combinations of) weighting metrics. In a second case study on the morpho-phonological tasks GP, GS, and MS, we saw that when data set size increases, careful abstractors can abstract more while keeping up with the performance of pure memory-based learning. In the third case study we applied the pure memory-based learning algorithm IB1-GR and the careful abstractor FAMBL-GR to a range of language learning tasks. FAMBL performed close to IB1-GR, equalling it on three tasks. Closer analyses are needed to determine what factors cause the slight disadvantage of FAMBL on the other three tasks GS, MS, and PP. From a language-engineering perspective, FAMBL offers adequate generalization accuracy on all six tasks, adding computationally interesting memory compression.

In a broader machine-learning context, our study suggests that careful abstraction over families of instances may be applicable to non-linguistic learning tasks as well, but more empirical work in this direction is needed. Currently, benchmark repositories hardly host non-linguistic data sets that mimic essential characteristics of language learning tasks (high disjunctivity, large data sets), although in real domains such data may exist: medical diagnosis may be an example. We envisage a large-scale study in which artificial data sets are generated according to a range of carefully-selected parameters (governing at least data set size and disjunctivity). Data may then be simulated along a continuum from low-disjunctive data to mimicking language data, in order to get a better understanding of which properties of language data favor the use of memory-based learning.

Acknowledgements

This research was done in the context of the "Induction of Linguistic Knowledge" (ILK) research programme, supported partially by the Netherlands Organization for Scientific Research (NWO). The author wishes to thank Walter Daelemans, the other members of the Tilburg ILK group, and the anonymous reviewers for their comments. Pedro Domingos (RISE) and Dietrich Wettschereck (NGE) are thanked for kindly granting permission to use their software.

References

Aha, D. W. 1997. Lazy learning: Special issue editorial. *Artificial Intelligence Review*, 11:7–10.

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

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

Baayen, R. H., R. Piepenbrock, and H. van Rijn. 1993. *The CELEX lexical data base on CD-ROM*. Linguistic Data Consortium, Philadelphia, PA.

Blake, C., E. Keogh, and C.J. Merz. 1998. UCI repository of machine learning databases, http://www.ics.uci.edu/~mlearn/mlrepository.html.

Cardie, Claire. 1994. *Domain Specific Knowledge Acquisition for Conceptual Sentence Analysis*. Ph.D. thesis, University of Massachusets, Amherst, MA.

Cardie, Claire. 1996. Automatic feature set selection for case-based learning of linguistic knowledge. In *Proc. of Conference on Empirical Methods in NLP*. University of Pennsylvania.

Collins, M.J and J. Brooks. 1995. Prepositional phrase attachment through a backed-off model. In *Proc. of Third Workshop on Very Large Corpora*, Cambridge.

Cost, S. and S. Salzberg. 1993. A weighted nearest neighbour algorithm for learning with symbolic features. *Machine Learning*, 10:57–78.

Daelemans, W. and A. Van den Bosch. 1992. Generalisation performance of backpropagation learning on a syllabification task. In M. F. J. Drossaers and A. Nijholt, editors, *Proc. of TWLT3: Connectionism and Natural Language Processing*, pages 27–37, Enschede. Twente University.

Daelemans, W., A. Van den Bosch, and A. Weijters. 1997. IGTree: using trees for compression and classification in lazy learning algorithms. *Artificial Intelligence Review*, 11:407–423.

Daelemans, W., A. Van den Bosch, and J. Zavrel. 1999. Forgetting exceptions is harmful in language learning. *Machine Learning*, 11(1–3):11–43.

Daelemans, Walter, Jakub Zavrel, Peter Berck, and Steven Gillis. 1996. MBT: A memory-based part of speech tagger generator. In E. Ejerhed and I.Dagan, editors, *Proc. of Fourth Workshop on Very Large Corpora*, pages 14–27. ACL SIGDAT.

Devijver, P. A. and J. Kittler. 1980. On the edited nearest neighbor rule. In *Proceedings of the Fifth International Conference on Pattern Recognition*. The Institute of Electrical and Electronics Engineers.

Dietterich, T. G. 1998. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, 10(7).

Domingos, P. 1996. Unifying instance-based and rule-based induction. *Machine Learning*, 24:141–168.

Gates, G. W. 1972. The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, 18:431–433.

Hart, P. E. 1968. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, 14:515–516.

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

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

Ramshaw, L.A. and M.P. Marcus. 1995. Text chunking using transformation-based learning. In *Proc. of Third Workshop on Very Large Corpora*, pages 82–94, June.

Ratnaparkhi, A., J. Reynar, and S. Roukos. 1994. A maximum entropy model for prepositional phrase attachment. In *Workshop on Human Language Technology*, Plainsboro, NJ, March. ARPA.

Salzberg, S. 1991. A nearest hyperrectangle learning method. *Machine Learning*, 6:277–309.

Salzberg, S. L. 1997. On comparing classifiers: Pitfalls to avoid and a recommended approach. *Data Mining and Knowledge Discovery*, 1(3).

Sejnowski, T. J. and C. S. Rosenberg. 1987. Parallel networks that learn to pronounce English text. *Complex Systems*, 1:145–168.

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

Van den Bosch, A. 1997. *Learning to pronounce written words: A study in inductive language learning*. Ph.D. thesis, Universiteit Maastricht.

Van den Bosch, A. 1999. Careful abstraction from instance families in memory-based language learning. *Journal for Experimental and Theoretrical Artificial Intelligence*. conditionally accepted.

Van den Bosch, A., W. Daelemans, and A. Weijters. 1996. Morphological analysis as classification: an inductive-learning approach. In K. Oflazer and H. Somers, editors, *Proceedings of the Second International Conference on New Methods in Natural Language Processing, NeMLaP-2, Ankara, Turkey*, pages 79–89.

Veenstra, J. B. 1998. Fast NP chunking using memory-based learning techniques. In *Proceedings of BENELEARN'98*, Wageningen, The Netherlands.

Wilson, D. 1972. Asymptotic properties of nearest neighbor rules using edited data. *Institute of Electrical and Electronic Engineers Transactions on Systems, Man and Cybernetics*, 2:408–421.

Zavrel, J., W. Daelemans, and J. Veenstra. 1997. Resolving PP attachment ambiguities with memory-based learning. In M. Ellison, editor, *Proc. of the Workshop on Computational Language Learning (CoNLL'97)*, ACL, Madrid.