Analogical Modeling with Bias — allowing Feedback and Centering

Christer Johansson and Lars G. Johnsen

Dept. of Linguistics and Literature
University of Bergen
N-5007 Bergen, Norway
{christer.johansson, lars.johnsen}@lili.uib.no

Abstract. We show a computationally efficient approximation (cf. [1]) of a full analogy model [2, 3], implemented in a computer program, and tested on the CoNLL2000 chunk tagging task [4], putting clause boundaries around mainly np and vp phrases. Our implementation showed to be competitive with other memory based learners. It deviates only slightly from the theoretical model. First, it implements a version of homogeneity check, which does not account fully for nondeterministic homogeneity. Second, it allows feedback of the last classification, and thirdly it allows centering on some central feature positions. Positions containing a) those parts-of-speech tags and b) those words that are to be given a chunk tag are given a weight which is given by how many match patterns that are equally or more general. A match on two centered features gives its patterns an extra weight given by the number of features. The results can be summarized as follows: a) using only lexical features performs below baseline. b) The implementation without anything extra, performs as the baseline for five parts-of-speech features, and centering improves the results. c) Feedback on its own does not improve results, while feedback + centering improves results more than just centering. Feedback on its own makes results deteriorate. The results exceed F=92, which is comparable with some of the best reported results for Memory Based Learning on the chunk tagging task.

1 Introduction

Analogical modeling (AM) is a (memory based) method to evaluate the analogical support for a classification [2, 3, 5]. Chandler [6] suggested AM as an alternative to both rule based and connectionist models of language processing and acquisition. AM defines a natural statistic, which can be implemented by comparisons of subsets of linguistic variables, without numerical calculations [5]. The natural statistic works as a selection mechanism, selecting those patterns in the database which most clearly points out a class for a novel pattern.

The original AM model compares all subsets of investigated variables. This may cause an exponential explosion in the number of comparisons, which has made it difficult to investigate large models with many variables (> 10) combined

© A. Gelbukh (Editor)
Advances in Natural Language Processing
Research in Computing Science 18, 2006, pp. 57-68

with large databases. Johnsen and Johansson [1] gives an accurate approximation of AM, which considers all analogical support from the database. The essential simplification (ibid.) is that each exemplar in the database only contributes with its most specific match to the incoming pattern to be classified. This provides a basis for directly comparing Skousens model to other models of memory based learning (MBL). In MBL, an example E is classified as belonging to category C by computing the score of E by going through the whole database. Skousens model requires the computation of the full analogical set for E, which we can now show to be approximated with resources that are close to a linear search through the database. The Johnsen and Johansson [1] approximation reduces the time complexity of the full analogical algorithm, but it may also simplify the addition of mechanisms such as feedback of the last classification, and putting focus on central features. The results imply that memory based learning methods are related by their evaluations of the nearest match set, where typically MBL only selects nearest neighbors. The AM model always considers all members of the database.

We will demonstrate that the proposed approximation reaches a high level of performance on a popular tagging task [4], if the algorithm is extended by mechanisms to focus on relevant features, as well as a mechanism of feedback of the latest classification. Without these additional mechanism, analogical modeling gives fairly low performance on this type of larger scale tasks that involve ambiguity and selecting a best alternative.

The implementation deviates slightly from the discussed, theoretical model. First, it implements a sloppy version of homogeneity check, which does not account fully for nondeterministic homogeneity. Second, it allows feedback of the last classification, and thirdly it allows centering on some central feature positions. The positions containing a) the parts-of-speech tags and b) the words that are to be given a chunk tag are given a high weight, and their immediate left and right context are given a lower weight. The weights are multiplied together for every matching feature position.

The results can be summarized as follows: a) using only lexical features performs below baseline. b) The implementation without anything extra, performs as the baseline for five parts-of-speech features, and centering improves the results. c) Feedback on its own does not improve results, while feedback + centering improves results more than just centering. Feedback only makes results deteriorate. The results reach F=92, which is comparable with the best reported results for Memory Based Learning on the task.

We can show a computationally efficient approximation of a full analogy model, implemented in a computer program, and tested on the CoNLL2000 chunk tagging task. This showed to be competitive with other memory based learners.

An empirical confirmation of the computational complexity showed a very slow increase with an increased number of features, although processing times increased with more demands on memory, an effect which is likely due to limits on internal memory.

We are presently not using feature weighting, such as information gain, which typically works on the level of individual feature values. Future research involves working on a method for automatically finding the relevant variables, and finding optimal weights (focus) for these variables.

2 Background on AM

We will not go into details of analogical modeling, beyond what is necessary for comparing it with memory based learning. Johnsen & Johansson [1] showed that the outcome in AM can be determined by summing up scores for each match pattern, where we only have to match the input once with all the examples in the database.

Examples in the database and each new input are expressed by a vector of feature values, similar to standard MBL. The operation of AM depends on matches. Each feature value may either match, between an example and the new input, or not. This creates a match vector where matches are encoded with a 1 and non-matches with 0, for example $< 0, 1, 0, 1, 1 >$ for five features.

We may imagine these vectors as a pointer to a box where we collect all the corresponding outcomes in the database. After we have gone through the database, we can look in all the non-empty boxes (which typically is of a much lower number than the number of examples), and observe the distribution of the outcomes. We are interested in those boxes that contain only one outcome. We call these boxes *first stage homogeneous*. Boxes with more than one outcome are less important, and may be discarded if we find homogeneous boxes pointed to by a more specific context, i.e. a match vector with more matches. The remaining (non-empty) boxes need to be sorted according to how many matches the index pattern contains. A more general pattern (e.g. $< 0, 0, 1 >$ is either homogeneous for the same outcome as the more specific pattern that it dominates (e.g. $< 1, 0, 1 >$, $< 0, 1, 1 >$, or $< 1, 1, 1 >$), or it is indeed *heterogeneous* and should be discarded.

A score$(\theta(x))$ is summed up for the number of homogeneous elements it dominates. Each part in the summation corresponds to looking in one of the above mentioned "boxes" (x). Each score for each box has an associated constant c_x, which would give us the exact value for full analogical modeling, if it was known.

The scoring of the analogical set expressed in mathematical notation is:

$$\sum_{x \in \mathcal{M}} c_x score(\theta(x)) \tag{1}$$

where $\mathcal{M}$ is the match set, and x is a context in the match set.

The implication of the work in [1] is that the match set $\mathcal{M}$, which is simple to calculate, contains *all* the results necessary for computing the overall effect, without actually building the whole analogical structure. In order to accurately weigh each context we need to estimate how many extensions each homogeneous pattern has. Johnsen and Johansson [1] develops a maximum and minimum

bound for this, and also discusses the possibilities for using Monte Carlo methods for discovering a closer fit.

Let us start with a simple and hypothetical case where M has exactly two members x and y with a different outcome. Any supracontextual label shared between x and y will be heterogeneous. The number of these heterogeneous labels are *exactly the cardinality of the power set of the intersection between x and y*. To see this, consider an example

$$\tau = (c, a, t, e, g, o, r, y)$$

and let x and y be defined as (using supracontextual notation):

$$\begin{array}{l} x = (c, -, t, -, -, o, r, -) \\ \text{with unique } score(\theta(x)) = (3, 0) \approx 3\ r \\ y = (c, a, -, -, -, o, -, -) \\ \text{with } score(\theta(y)) = (0, 8) \approx 8\ e \end{array} \tag{2}$$

Their common and therefore heterogeneous supracontextual labels are

$$\left.\begin{array}{l} (c, -, -, -, -, o, -, -) \\ (c, -, -, -, -, -, -, -) \\ (-, -, -, -, -, o, -, -) \\ (-, -, -, -, -, -, -, -) \end{array}\right\} \tag{3}$$

The total number of elements that dominate x is sixteen; the homogeneous labels out of these sixteen are those that dominate x and no other element with a different outcome; in this case y. The labels x shares with y are the four labels in (3), and x has 16-4=12 homogeneous labels above it. How is that number reached using sets?

Viewed as sets, the elements x and y are represented as:

$x = \{c_1, t_3, o_6, r_7\}$ and $y = \{c_1, a_2, o_6\}$.

Their shared supracontexts are given by the power set of their common variables.

$$\begin{array}{l} x \cap y = \{c_1, t_3, o_6, r_7\} \cap \{c_1, a_2, o_6\} = \{c_1, o_6\} \\ \mathcal{P}(x \cap y) = \\ \quad \mathcal{P}(\{c_1, o_6\}) = \{\emptyset, \{c_1, o_6\}, \{o_6\}, \{c_1\}\} \end{array} \tag{4}$$

This set has four elements all in all, which all are equivalent to the labels in (3). The sets in (4) represent the heterogeneous supracontextual labels more general than either x or y and these are the only heterogeneous supracontexts in the lattice Λ of supracontextual labels, given the assumptions made above.

The power sets for x and y have 16 and 8 elements respectively, so the total number of homogeneous supracontextual labels more general than either x or y is the value for the coefficients c_x and c_y from (1) calculated as:

$$\left.\begin{array}{l} c_x = \|\mathcal{P}(x) - \mathcal{P}(x \cap y)\| = 16 - 4 = 12 \\ c_y = \|\mathcal{P}(y) - \mathcal{P}(x \cap y)\| = 8 - 4 = 4 \end{array}\right\} \tag{5}$$

Plugging these numbers into the formula (1) gives the score of the analogical set for this case:

$$\begin{aligned}&\sum_{x \in M} c_x score(\theta(x)) = \\ &= 12 score(\theta(x)) + 4 score(\theta(y)) \\ &= 12\,(3,0) + 4\,(0,8) \\ &= (36,0) + (0,32) \\ &= (36,32)\end{aligned} \tag{6}$$

In the general case however, the set $\mathcal{M}$ consists of more elements, complicating the computation somewhat. Each $x \in \mathcal{M}$ may share a number of supracontextual elements with other elements of $\mathcal{M}$ that have a different outcome. The situation may be as depicted in the following table, where columns are labelled by elements of $\mathcal{M}$ (in boldface) with their associated hypothetical outcomes (in italics).

Table 1. Accessing disagreement in $\mathcal{M} \times \mathcal{M}$

$\mathcal{M}$	$\mathbf{a}_r$	$\mathbf{g}_e$	$\mathbf{h}_r$	$\mathbf{d}_f$
$\mathbf{a}_r$		$\mathcal{P}_{(a \cap g)}$		$\mathcal{P}_{(a \cap d)}$
$\mathbf{g}_e$	$\mathcal{P}_{(g \cap a)}$		$\mathcal{P}_{(g \cap h)}$	
$\mathbf{h}_r$		$\mathcal{P}_{(h \cap g)}$		$\mathcal{P}_{(h \cap d)}$
$\mathbf{d}_f$	$\mathcal{P}_{(d \cap a)}$	$\mathcal{P}_{(d \cap g)}$	$\mathcal{P}_{(d \cap h)}$	

Each cell in Table 1 is associated with the power set $\mathcal{P}(x \cap y)$. This power set is only computed if the outcome of x is not equal to the outcome of y, and both outcomes are unique. The intersection is computed for all labels with a non-unique outcome, even for those with identical outcomes. If two elements are non-unique, any label that is a subset of both will match up with their respective and disjoint data sets (see propositions 1 and 2 in [1]), thereby *increasing* the number of disagreements, and consequently turning any such label into a heterogeneous label. Note that a and h have the same outcome in this table making their intersective cells empty.

Each non-empty cell corresponds to the simple case above. The complication stems from the fact that different cells may have non-empty intersections, i.e., it is possible that

$$\mathcal{P}(a \cap g) \cap \mathcal{P}(a \cap d) \neq \emptyset$$

Arithmetic difference of the cardinality of the cells may be way off the mark, due to the possibility that supracontexts may be subtracted more than once.

Something more sophisticated is needed to compute the desired coefficients c_x. A couple of approximations are given in the following.

The approximations are gotten at by first collecting all subcontexts different from $a$in a set $\delta(a)$:

$$\delta(a) = \{x \in \mathcal{M} | o(a) \neq o(x)\}$$

This equation represents the column labels for the row for a. The total number of homogeneous supracontexts ($=c_a$) more general than a is the cardinality of the set difference

$$\mathcal{P}(a) - \bigcup_{x \in \delta(a)} \mathcal{P}(a \cap x) \tag{7}$$

The second term in (7) corresponds to the value of the function H in [1], and is the union of the power sets in the row for a. It represents the collection of supracontextual labels more general than a, which also are shared with another subcontext, thus making all of them heterogeneous. The first term, $\Pi(a)$, is the set of all supracontextual labels more general than a. Therefore, the difference between these two sets is equal to the collection of homogeneous supracontextual labels more general than a. However, it is not the content of these sets that concerns us here; the goal is to find the cardinality of this difference.

The cardinality of $\mathcal{P}(a)$ is given the normal way as

$$\|\mathcal{P}(a)\| = 2^{\|a\|}$$

but how are the behemoth union to be computed? This raises the question of computing the union of power sets:

$$\bigcup_{x \in \delta(a)} \mathcal{P}(a \cap x) \tag{8}$$

The exponential order of the analogical algorithm stems from the computational complexity of this set. The union is bounded both from below and above. A lower bound is:

$$\mathcal{P}(max(\{a \cap x | x \in \delta(a)\})$$

and a higher bound is:

$$\mathcal{P}(\bigcup_{x \in \delta(a)} a \cap x)$$

Both these bounds are fairly simple to calculate. In the implementation (written in C), we have chosen a weighted average between the lower bound and the higher bound as a good approximation. We found that values that are weighted in favor of the higher bound gave better performance. This is not equivalent to say that the true AM values are closer to the higher bound.

3 Results from the Implementation

We have evaluated the performance of our implementation using the chunk identification task from CoNLL–2000 [4]. The best performance was obtained by a Support Vector Machine [7, F=93.48]. This implementation has the disadvantage that it is computationally very intensive, and it might not be applicable to much larger data sets. Their results have later improved even more for the same task. The standard for memory based learning on this task is an F value of 91.54 [8], a value which can be improved upon slightly, as is shown by using a system combining several different memory-based learners [9, F=92.5]. Johansson [10] submitted an NGRAM model which used only 5 parts-of-speech tags , centered around the item to be classified. That model (ibid.) used a backdown strategy to select the largest context attested in the training data, and gave the most frequent class of that context. It used a maximum of 4 look-ups from a table, and is most likely the fastest submitted model. The table could be created by sorting the database. The advantage being that it could handle very large databases (as long as they could be sorted in reasonable time). The model gives a minimum baseline for what a modest $NGRAM$-model could achieve (F=87.23) on the chunking task.

3.1 Deviations from AM

The implementation deviates slightly from the discussed, theoretical model. First, it implements a version of homogeneity check, which does not account for non-deterministic homogeneity [2, 3, 5]. We tried a more extensive homogeneity check in an earlier version, but the results actually deteriorated. Second, it allows feedback of the last classification, and thirdly it allows centering on some central feature positions. The positions containing a) the parts-of-speech tags and b) the words that are to be given a chunk tag are given a weight given by how many more general patterns exist.

Giving Proper Weight on the Focussed Items The number of patterns with a lower or an equal number of hits is given by:

$$a) \sum_{k=0}^{hits} \binom{n}{k} \quad b) \; n \sum_{k=0}^{hits} \binom{n}{k} \quad c) \sum_{k=0}^{hits} \binom{n}{k} / n; \tag{9}$$

The term *hits* refers to the number of matching features in the match pattern we are considering. Formula a) in 9 refers to the case where we have found one centered feature. Formula b) refers to when both centered features have been found, and finally formula c) refers to when none of the centered features have been found. In the case that none of the features have been found the effect is spread evenly over the available variables (feature positions). If one matching centered feature is found the effect is concentrated to one, and if both centered features match we might have choosen the second centered feature in $n-1$ ways. We have added 1, in part to make the formulas more uniform.

3.2 Performance

From Table 2 we can see that using only lexical features (i.e. 5 lex and 6 lex), performs below baseline (F=87.23). The implementation without anything extra (F=87.65), performs only slightly better than the base line for five parts-of-speech features, and centering improves that to 88.50. Feedback on its own has a small effect (87.39; 6 pos, 82.95 6 lex), while feedback + centering (F=89.03: 6 pos, 87.69; 6 lex) improves results compared to just centering. Feedback does introduce some mistakes from the mechanism, and centering allows some of those to be corrected.

Table 2. Results: F-scores. # features, Feedback + Centering, Centering only, Feedback only, nothing extra. Results within () are results without the binomial weighting from eqn. 9c

#	F+C	C	F	0
5 lex		85.95		82.50 (80.40)
5 pos		88.50		87.65 (87.13)
6 lex	87.69		82.95 (83.17)	
6 pos	89.03		87.39 (87.02)	
10		91.45		89.69 (89.48)
11	92.23		89.58 (89.41)	

The model with only centering using both lexical and parts-of-speech features approaches MBL results, and performs slightly better with feedback and centering (F=92.23, see Table 3), although not as good as the SVM-implementation [7], nor the system combining several memory based learners [9].

Table 3. Detailed results for each category. (11 F+C)

selected	precision	recall	$F_{\beta=1}$
ADJP	72.91%	67.58%	70.14
ADVP	77.87%	78.41%	78.14
CONJP	36.84%	77.78%	50.00
INTJ	100.00%	50.00%	66.67
NP	92.15%	93.29%	92.72
PP	95.91%	97.38%	96.64
PRT	71.72%	66.98%	69.27
SBAR	87.82%	82.24%	84.94
VP	92.21%	92.74%	92.48
accuracy	precision	recall	FB1
95.13%	91.86%	92.60%	92.23

We have also tried out an idea of providing a model for novel items by constructing instances where low frequency words were replaced by a marker common to unseen words in the test set. This idea resulted in the same, or slightly worse results. This indicates that the algorithm does not get any new information from these added constructions, i.e. the information was already available, and it showed to be fairly hard to alter the classification by adding items to the database.

3.3 Computational Performance and Expected Complexity

The time complexity of the abstract algorithm for the worst case was asserted to be $O(log(N)N$ [1]. The current implementation has a nested loop over the match set, which in the worst case may grow to be as large as the database. This would make the algorithm $O(N^2)$ in the worst case. We do not expect that to happen in the average case. What was the performance on the *CoNLL* task?

The tests were made using a 867MHz PowerPC G4, with 1 MB L3 cache and 256 MB SDRAM memory using Mac OS X version 10.3.9.

When the number of variables changed, the number of unique patterns varied. The time to process all test patterns were therefor divided by the number of unique database items and reported as how many milliseconds per database item the processing took.

The results are shown in Table 4. This shows an almost linear increase with the number of variables, which has to do with a) that more comparisons are made because there are more variables (and features values) to compare, and b) that the match set $\mathcal{M}$ grows slightly faster when there are more variables. A deviation from the linear marks this increase in match set growth.

Table 4. Processing time needed to solve the full task, per item in the database.

#	*D*	ms
5 lex	213532	17.44
5 pos	92392	17.84
6 lex	213562	19.14
6 pos	92392	19.80
10	213562	26.07
11	213591	28.68

When the size of the database is accounted for, the main contribution to complexity is proportional to the square of the size of the match set times the number of variables. That time expenditure does not grow faster indicates that the match set does not grow very fast for this task. The values in Table 4 are approximated by the formula: $time = 0.05 * f^2 + f + 11.5$, with $R^2 > 0.98$; f = number of variables, and time is in ms per database item, for processing all 49393 instances in the test set.

4 Future Research

The relevant features for focus can be found automatically in many cases, by looking at how the variable in general correlates with the outcome. This may expand the model to consider more than two focussed variables.

We are presently not using feature weighting, such as information gain, which typically works on the level of individual feature values. This might give some room for future improvement.

5 Conclusion

We have shown an outline of a theoretical reconstruction of Skousen's Analogical Modeling of Language [2, 3, 5], this is described in more detail elsewhere [1, 11]. This reconstruction led to a more efficient approximation of full analogy modeling, and the results were implemented in a computer program, and tested on the *CoNLL* – 2000 chunk tagging task. Our implementation showed to be competitive with other memory based learners, after accounting for the specificity of the match patterns and how many patterns were more general than the pattern under consideration.

Admittedly, non-naive implementations of a nearest neighbor model, such as TiMBL [12], are already doing well for large data sets, which make it hard to compete on combined accuracy and processing time. The main contribution of this model is that it implements a memory based model without the need to specify how many nearest neighbors (or neighbor distances) to consider. In the spirit of AM there are no parameters to set, and no calculation of gain for knowing some individual item. We have added the possibility to center on particular variables, to separate them from context variables. We have also provided the possibility to use feedback of the last categorization. As there are some approximations involved, we have made it possible to alter the weights to make it possible to optimize the results. The highest performance reached so far has been F=92.25 (compared to the reported F=92.23 for the standard settings).

The implementation has reached its level of performance without calculating the information gain of knowing some individual feature values. In this, the implementation follows the philosophy of parameterless analogical modeling. It should be interesting for linguistic models that a model based on selection can reach fairly high performance without calculating any statistics based on the individual items.

Acknowledgement Support by a grant from the Norwegian Research Council under the KUNSTI programme (project BREDT) is kindly acknowledged. The computer program will be made available for download from http://bredt.uib.no with some example data.

References

1. Johnsen, L., Johansson, C.: Efficient modeling of analogy. In Gelbukh, A., ed.: Proceedings of the 6th Conference on Intelligent Text Processing and Computational Linguistics. Volume 3406 of Lecture Notes in Computer Science. Springer Verlag, Berlin, Germany (2005) 682–691
2. Skousen, R.: Analogical Modeling of Language. Kluwer Academic, Dordrecht, the Netherlands (1989)
3. Skousen, R.: Analogy and Structure. Kluwer Academic, Dordrecht, the Netherlands (1992)
4. Tjong Kim Sang, E., Buchholz, S.: Introduction to the CoNLL-2000 shared task: Chunking. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 127–132
5. Skousen, R., Lonsdale, D., Parkinson, D.: Analogical Modeling: An exemplar-based approach to language. Volume 10 of Human Cognitive Processing. John Benjamins, Amsterdam, the Netherlands (2002)
6. Chandler, S.: Are rules and modules really necessary for explaining language? Journal of Psycholinguistic Research **22** (1993) 593–606
7. Kudoh, T., Matsumoto, Y.: Use of support vector learning for chunk identification. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 142–144
8. Veenstra, J., Bosch, A.v.d.: Single-classifier memory-based phrase chunking. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 157–159
9. Tjong Kim Sang, E.: Text chunking by system combination. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 151–153
10. Johansson, C.: A context sensitive maximum likelihood approach to chunking. In: Proceedings of CoNLL-2000 and LLL-2000, Lisbon, Portugal (2000) 136–138
11. Johnsen, L.: Commentary on exegesis of Johnsen and Johansson, 2005, (ms.) (2005)
12. Daelemans, W., Zavrel, J., van der Sloot, K., van den Bosch, A.: TiMBL: Tilburg Memory Based Learner, version 5.1. Reference guide. ILK Technical report Series 04-02., Tilburg, the Netherlands (2004)