An analytical relation between analogical modeling and memory based learning

Christer Johansson & Lars G. Johnsen
Dept. of Linguistics and Literature
University of Bergen
N-5007 Bergen, Norway
{christer.johansson, lars.johnsen}@lili.uib.no

Abstract

Analogical modeling (AM) is a memory based model. Known algorithms implementing AM depend on investigating all combinations of matching features, which in the worst case is exponential ($O(2^n)$). We formulate a representation theorem on analogical modeling which is used for implementing a range of approximations to AM with a much lower complexity. We will demonstrate how our model can be modified to reach better performance than the original AM model a popular categorization task (chunk tagging).

1 Introduction

Analogical modeling (AM) is a method to evaluate the analogical support for a classification (Skousen, 1989; Skousen, 1992; Skousen et al., 2002), which is compatible with psycholinguistic data (Chandler, 2002; Eddington, 2002; Mudrow, 2002, inter al.). Chandler (1993) propose it as an alternative to rule based and connectionist models of language processing and acquisition.

AM has also been used within Optimality Theory (Myers, 2002) and similar exemplar-based studies have looked at the development of phonological regularities with the number of exemplars (Pierrehumbert, 2001). AM defines a natural statistic, which can be implemented by comparisons of subsets of linguistic variables, without numerical calculations (Skousen, 1992; Skousen, 2002).

The original AM model compares all subsets of investigated variables. This causes an exponential explosion in the number of comparisons. This has made it difficult to investigate large models with many variables (> 10) and large databases. Johnsen & Johansson (2005) gives an accurate approximation of AM, which considers all analogical support from the database. The analytical upper and lower bounds of the approximation is also provided (ibid.).

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 Skousen's 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. Skousen's model require the computation of the full analogical set for E. We can show this computation to be approximated with resources that are close to a linear search through the database.

The Johnsen & Johansson approximation reduces the time complexity of the analogical algorithm. The results imply that AM and many MBL models are related by different evaluations of the nearest match set. MBL typically selects nearest neighbors, whereas AM attempt to look at all support from the entire database, thus not needing to specify how many neighbors to look for.

Full AM and the proposed approximation has both been tested on predicting chunk tags (Tjong Kim Sang and Buchholz, 2000), as well as some test sets provided by Skousen (1989; 1992), and the approximation agrees very well with full AM results. However, the initial results for the chunk tagging task where disappointing for both AM models. The performance of AM and MBL has previously been compared favourably (Daelemans, 2002; Eddington, 2002, inter al.), but for some reason the initial AM model does not do very well on the chunk tasks, presumably because this

S. Werner (ed.), *Proceedings of the 15th NODALIDA conference, Joensuu 2005*, Ling@JoY 1, 2006, pp. 88–93
ISBN 952-458-771-8, ISSN 1796-1114. http://ling.joensuu.fi/lingjoy/

task involves many heterogeneous patterns, i.e. patterns with more than one outcome. MBL relates to AM through the Johnsen & Johansson(2005) approximation of AM with an alternative weighting of contexts; a weighting that favors homogeneous contexts for AM, whereas MBL look at a weighted majority of outcomes from nearest neighbors, without concern for which patterns are homogeneous or not.

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 (2005) 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 (Johnsen and Johansson, 2005) 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 (2005) 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.

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

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:

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

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).

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

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

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

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 (Johnsen and Johansson, 2005)), 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 JJ, 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 lies in trying to compute 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 (Tjong Kim Sang and Buchholz, 2000). The best performance was obtained by a Support Vector Machine (Kudoh and Matsumoto, 2000, 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 (Veenstra and Bosch, 2000), a value which can be improved upon slightly, as is shown by using a system combining several different memory-based learners (Tjong Kim Sang, 2000, F=92.5). Johansson (2000) 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 lookups 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.

The implementation deviates slightly from the discussed, theoretical model. First, it implements a "sloppy" version of homogeneity check, which does not account for nondeterministic homogeneity (Skousen, 1989; Skousen, 1992; Skousen et al., 2002). We tried to implement this in an earlier version, but the results actually deteriorated, so we decided to go for the "sloppy" version. 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 (100), and their immediate left and right context are given weight as well (3 and 2 respectively). The weights are multiplied together for every matching feature position. The highest weight is thus $3 * 100 * 2 * 3 * 100 * 2 = 600 * 600 = 360000$. A method for automatically setting the focus weights is under development. 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, performs as the base line for five parts-of-speech features (F=87.13), and centering improves that to 88.87. Feedback on its own does not improve results (87.02; 6 pos), while feedback + centering (F=89.36) improves results more than just centering. Feedback on its own rather seems to deteriorate results. The model approaches MBL–results with

only centering using both lexical and parts-of-speech features, and performs slightly better with feedback and centering (F=92.05), although not as good as the SVM-implementation (Kudoh and Matsumoto, 2000), and not as good as the system combining several memory based learners (Tjong Kim Sang, 2000). A system that used 11 word features, and 11 parts-of-speech features, as well as a feedback feature was also trained, but did not perform better. The testing time for that 23-feature model was about 30 hours, making systematic testing difficult.

#	F+C	C	F	0
5 lex		82.91		80.40
5 pos		88.87		87.13
6 lex	86.33		83.17	
6 pos	89.36		87.02	
10		91.16		89.48
11	92.05		89.41	

Table 2: Results: F–scores. Number of features, Feedback + Centering, Centering only, Feedback only, nothing extra.

3.1 Computational complexity

The test 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 features 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 3. This shows an almost linear increase with the number of features, which has to do with a) that more comparisons are made because there are more features to compare, and b) that the match set $\mathcal{M}$ grows faster when there are more features. 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 features. That complexity does not grow faster is an indication that the match set does not grow very fast for this task. However, when using 23 features, computing demanded more time than accounted for by the trend shown for less features. We speculate that this is mainly because of the limited RAM memory available. The values in table 3 are approximated by the formula: $time = 0.05 * f^2 + f + 11.5$, with $R^2 > 0.98$; f = number of feature positions (variables), and time is in ms per database item, for processing all 49393 instances in the test set.

#	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

Table 3: Results: Processing time needed to solve the full task, per item in the database.

4 Future research

We are trying to invent a method for automatically focussing on the relevant variables (feature positions), set optimal weights for these variables. The goal is to get even better results from this method, than from using the rather ad hoc weights used in this presentation. The focus on central variables reminds of the backdown strategy used in (Johansson, 2000), the results are very similar to that NGRAM–method for using only parts-of-speech information. The lexical information might be integrated in an NGRAM–model, weighted in an efficient way to produce results similar to the best results in this presentation (F=92.05), but without the computational overhead of the analogical model. The advantage of such a model is that it could use larger databases, and therefore be much more practical, even if it does not deliver optimal results for smaller sets of data. Computation in the NGRAM–model (Johansson, 2000) consists of a fixed number of fast look-up operations, and training is feasible if the data can be sorted in reasonable time. Admittedly, non-naive implementations of a nearest neighbor model, such as TiMBL (Daelemans et al., 2004), are already doing well for large data sets, which make it hard to compete on combined accuracy and processing time.

5 Conclusion

We have shown an outline of a theoretical reconstruction of Skousen's Analogical Modeling of Language (Skousen, 1989; Skousen, 1992; Skousen et al., 2002), this is described in more

detail elsewhere (Johnsen and Johansson, 2005; Johnsen, 2005). 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. 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 for these variables.

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

S. Chandler. 1993. Are rules and modules really necessary for explaining language? *Journal of Psycholinguistic Research*, 22:593–606.

S. Chandler. 2002. Skousen's analogical approach as an exemplar-based model of categorization. In *Skousen et al.*, pages 51–105.

W. Daelemans, J. Zavrel, K. van der Sloot, and A. van den Bosch. 2004. *TiMBL: Tilburg Memory Based Learner, version 5.1. Reference guide*. ILK Technical report Series 04–02., Tilburg, the Netherlands.

W. Daelemans. 2002. A comparison of analogical modeling to memory-based language processing. In *Skousen et al.*, pages 157–179.

D. Eddington. 2002. A comparison of two analogical models: Tilburg memory-based learner versus analogical modeling. In *Skousen et al.*, pages 141–155.

C. Johansson. 2000. A context sensitive maximum likelihood approach to chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 136–138, Lisbon, Portugal.

L. Johnsen and C. Johansson. 2005. Efficient modeling of analogy. In A. Gelbukh, editor, *Proceedings of the 6th Conference on Intelligent Text Processing and Computational Linguistics*, volume 3406 of *Lecture Notes in Computer Science*, pages 682–691. Springer Verlag, Berlin, Germany.

L. Johnsen. 2005. *Commentary on exegesis of Johnsen and Johansson, 2005*. (ms.).

T. Kudoh and Y. Matsumoto. 2000. Use of support vector learning for chunk identification. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 142–144, Lisbon, Portugal.

M. Mudrow. 2002. Version spaces, neural networks, and analogical modeling. In *Skousen et al.*, pages 225–264.

J. Myers. 2002. Exemplar-driven analogy in optimality theory. In *Skousen et al.*, pages 265–300.

J. Pierrehumbert. 2001. Exemplar dynamics: Word frequency, lenition, and contrast. In *Frequency effects and the emergence of linguistic structure*, pages 137–157, Amsterdam, the Netherlands. John Benjamins.

R. Skousen, D. Lonsdale, and D.B. Parkinson. 2002. *Analogical Modeling: An exemplar-based approach to language*, volume 10 of *Human Cognitive Processing*. John Benjamins, Amsterdam, the Netherlands.

R. Skousen. 1989. *Analogical Modeling of Language*. Kluwer Academic, Dordrecht, the Netherlands.

R. Skousen. 1992. *Analogy and Structure*. Kluwer Academic, Dordrecht, the Netherlands.

R. Skousen. 2002. Issues in analogical modeling. In *Skousen et al.*, pages 27–48.

E.F. Tjong Kim Sang and S. Buchholz. 2000. Introduction to the CoNLL-2000 shared task: Chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 127–132, Lisbon, Portugal.

E.F. Tjong Kim Sang. 2000. Text chunking by system combination. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 151–153, Lisbon, Portugal.

J. Veenstra and A. van den Bosch. 2000. Single-classifier memory-based phrase chunking. In *Proceedings of CoNLL-2000 and LLL-2000*, pages 157–159, Lisbon, Portugal.