Analogical Modeling: An Update

David Eddington and **Deryle Lonsdale**
Department of Linguistics & English Language
Brigham Young University
Provo, UT, USA 84602
{eddington,lonz}@byu.edu

Abstract

Analogical modeling is a supervised exemplar-based approach that has been widely applied to predict linguistic behavior. The paradigm has been well documented in the linguistics and cognition literature, but is less well known to the machine learning community. This paper sets out some of the basics of the approach, including a simplified example of the fundamental algorithm's operation. It then surveys some of the recent analogical modeling language applications, and sketches how the computational system has been enhanced lately to offer users increased flexibility and processing power. Some comparisons and contrasts are drawn between analogical modeling and other language modeling and machine learning approaches. The paper concludes with a discussion of ongoing issues that still confront developers and users of the analogical modeling framework.

1 Introduction

Analogical modeling (AM) is an exemplar-based modeling approach designed to predict linguistic behavior on the basis of stored memory tokens (Skousen, 1989; Skousen, 1992; Skousen, 1995; Skousen, 1998). It is founded on the premise that previous linguistic experience is stored in the mental lexicon. When the need arises to determine some linguistic behavior (pronunciation, morphological relationship, word, etc.), the lexicon itself is accessed. A search is conducted for the stored exemplars that are most similar to the one whose behavior is being predicted. The behavior of highly similar stored entities generally predicts the behavior of the one in question, although less similar ones have a small chance of applying as well.

AM has been implemented in a succession of computer implementations that allow users to specify an exemplar base and then to test unseen instances of language behavior against this accumulated store to predict (and quantify) the relevant and possible outcome(s).

A comprehensive examination of the system is beyond the scope of this paper[1]. Instead, an attempt will be made to situate AM within the space of exemplar-based modeling systems and, to a certain extent, within the larger context of machine learning systems and cognitive modeling systems in general.

This paper begins by giving an informal overview of the algorithm used to perform AM. To accomplish this it traces in schematic form an early but typical application of the AM approach: Spanish gender guessing. The subsequent section discusses AM with respect to other types of modeling systems, and gives an overview of the types of linguistic phenomena modeled by AM. The paper then discusses several items of recent and ongoing work with the AM program. The conclusion mentions possible areas for future improvement and investigation.

[1]For further information including a bibliography and software downloads see the project webpage at: http://humanities.byu.edu/am

Subcontexts	Members of the Subcontext	Pointers	# of Disagreements
pan	none	none	0
$\overline{\text{p}}$an	*plan* M	*plan* M > *plan* M	0
p$\overline{\text{a}}$n	none	none	0
pa$\overline{\text{n}}$	*paz* F, *par* M	*paz* F > *paz* F *par* M > *par* M **paz* F > *par* M **par* M > *paz* F	2
$\overline{\text{p}}$a$\overline{\text{n}}$	*cal* F	*cal* F > *cal* F	0
$\overline{\text{p}}\overline{\text{a}}$n	*tren* M, *ron* M	*tren* M > *tren* M *ron* M > *ron* M *tren* M > *ron* M *ron* M > *tren* M	0
p$\overline{\text{a}}\overline{\text{n}}$	none	none	0
$\overline{\text{p}}\overline{\text{a}}\overline{\text{n}}$	*rey* M	*rey* M > *rey* M	0

Table 1: Subcontexts and their disagreements.

2 The AM algorithm

Perhaps the best way to understand the AM algorithm is with a concrete illustration. Predictions are always made in terms of specific exemplars, therefore, for the purposes of the example, the following seven monosyllabic Spanish nouns and their corresponding gender will be considered[2]:

paz	F
plan	M
tren	M
cal	F
ron	M
par	M
rey	M

The task will be to predict the gender of *pan* 'bread' on the basis of these seven items as the set of exemplars. The three variables used are the phoneme or phoneme cluster of the onset, nucleus, and rhyme. First, all possible exemplar items are assigned to a series of subcontexts which are defined in terms of the given context pan. This yields eight subcontexts (Table 1). A bar over a variable indicates that any value of the variable except the barred value is permitted.

By assigning members to subcontexts it is possible to determine disagreements. Disagreements are marked with asterisks in Table 1. A disagreement occurs when words that are equally similar to the given context, exhibit different behaviors, in this case, different genders. The number of disagreements is determined by pairing all members of a subcontext with every other member, including itself, by means of unidirectional pointers, and counting the number of times the members of the pair have different behaviors. In this example, the only subcontext containing any disagreement is pa$\overline{\text{n}}$.

Subcontexts are then arranged into more comprehensive groups called supracontexts as shown in Table 2. In Table 2, a hyphen indicates a wildcard.

In the subcontextual analysis a tally of the all of the subcontextual disagreements is made. The supracontextual analysis consists of analyzing all of the words that appear in a given supracontext, and again tallying disagreements (Table 3).

In the supracontexutal analysis, words that have more than one variable in common with pan appear in more that one supracontext. This is AM's way of allowing the gender of words which are more similar to pan to influence the gender assignment of pan to a greater extent.

The purpose of AM's algorithm is to determine which members of the exemplar base are most likely to affect the gender assignment of pan, and also to calculate the extent of analogical influence exerted.

[2]This example is a simplification; see (Eddington, 2002) for a complete AM treatment of this phenomenon.

Supracontext	Subcontexts in Supracontext	# Subcontextual Disagreements
p a n	pan	0
p a –	pan, *pa$\overline{\text{n}}$	2
p – n	pan, p$\overline{\text{a}}$n	0
– a n	pan, $\overline{\text{p}}$an	0
p – –	pan, p$\overline{\text{a}}$n, *pa$\overline{\text{n}}$, p$\overline{\text{a}}\overline{\text{n}}$	2
– a –	pan, $\overline{\text{p}}$an, *pa$\overline{\text{n}}$, $\overline{\text{p}}$a$\overline{\text{n}}$	2
– – n	pan, $\overline{\text{p}}$an, p$\overline{\text{a}}$n, $\overline{\text{p}}\overline{\text{a}}$n	0
– – –	pan, $\overline{\text{p}}$an, p$\overline{\text{a}}$n, *pa$\overline{\text{n}}$, $\overline{\text{p}}$a$\overline{\text{n}}$, $\overline{\text{p}}\overline{\text{a}}n, p\overline{\text{a}}\overline{\text{n}}$, $\overline{\text{p}}\overline{\text{a}}\overline{\text{n}}$	2

Table 2: Subcontextual analysis.

This is accomplished by calculating heterogeneity. Heterogeneity is determined by comparing the number of disagreements in the supracontextual and subcontextual analyses. If there are more disagreements in the supracontextual analysis, the supracontext is heterogenous, and its members are eliminated from consideration as possible analogs. If the number of disagreements does not increase, the supracontext is homogenous.

Words belonging to homogenous supracontexts comprise the analogical set (Table 4). In the example under consideration, disagreements increase in the supracontexts – – –, and – a –. Therefore, their members are eliminated from consideration. *Cal*, and *rey* appear exclusively in these heterogenous supracontexts. As a result, they do not form part of the analogical set. The word *plan* is also a member of both – – –, and – a –, however, it is also a member of the homogenous supracontexts – a n, and – – n, so it will still be available to influence *pan*.

It should not be surprising that *rey* would be eliminated through heterogeneity; it has no phonemes in common with pan. However, consider the words *ron* and *cal*. Both share only one feature with pan, yet heterogeneity eliminates only *cal*, and not *ron*. This is due to the fact that *ron* appears in the supracontext – – n, and all of the members of that supracontext are masculine, therefore, there is no disagreement. *Cal* F, on the other hand, competes with masculine words in all of the supracontexts in which it appears.

The analogical set contains all of the exemplar items that can possibly influence the gender assignment of *pan*. There are two methods for calculating the influence of the analogical set on the behavior of the given context. One is to assign the most frequently occurring behavior to the given context (selection by plurality). Of the 18 pointers in the set, 14 point to masculine. Therefore, selection by plurality would assign masculine gender to *pan*. The other method (random selection) involves randomly selecting a pointer, and assigning the behavior of the word indicated by the pointer to the given context. In this case, the probability of masculine gender assignment would be 77.78% (14/18).

3 Situating AM

The AM theoretical construct has been implemented computationally through several generations of application programs. Pascal code for the first version was listed as an appendix in (Skousen, 1989). A subsequent version was implemented in C. In recent years the system was reimplemented as a Perl module; extensive details on how to use the current version are available in (Skousen et al., 2002).

The system assumes *a priori* labeled input instances, and thus only works in supervised learning mode. Each exemplar must be represented as a fixed-length vector comprised of features that may or may not eventually be relevant to the phenomenon being modeled. The features themselves may be of fixed or variable length, as long as they are appropriately delimited. An outcome is specified for each exemplar instance vector. Vector features are purely symbolic and nominal, not admitting continuous values. When the use of continuous-valued features is needed (e.g. integers or real numbers),

Supracontexts	Words in Supracontext	Pointers	# of Subcontextual Disagreements
p a n	none	none	0
p a –	*par* M, *paz* F	*paz* F > *paz* F *par* M > *par* M **paz* F > *par* M **par* M > *paz* F	2
p – n	none	none	0
– a n	*plan* M	*plan* M > *plan* M	0
p – –	*par* M, *paz* F	*paz* F > *paz* F *par* M > *par* M **paz* F > *par* M **par* M > *paz* F	2
– a –	*plan* M, *cal* F, *par* M, *paz* F	*plan* M > *plan* M **plan* M > *cal* F *plan* M > *par* M **plan* M > *paz* F *cal* F > *cal* F **cal* F > *plan* M **cal* F > *par* M *cal* F > *paz* F **paz* F > *plan* M **paz* F > *par* M (not all shown; all the rest involve agreement)	6
– – n	*plan* M, *tren* M, *ron* M	(not shown)	0
– – –	*plan* M, *tren* M, *ron* M, *cal* F, *paz* F, *par* M, *rey* M	(not shown)	12

Table 3: Supracontextual analysis.

the features must be symbolically quantized within pre-specified ranges in order to be meaningful to the system. More details on how to choose, quantize, and encode exemplar vectors is given in (Skousen et al., 2002).

As the system loads up the feature vectors, it builds the contextual lattice as sketched in the previous section. Once the lattice has been constructed, the system can process incoming test items to determine their outcome(s). Each test item consists of a feature vector containing the same number of features as the exemplar vectors do. Each test item also has the appropriate outcome specified, if known, or else some placeholder symbol. Each test item is matched against the lattice, its analogical set computed, and outcomes reported. AM, unlike some other machine learning approaches, can report all possible outcomes, and associates each with a probability rating or confidence rating reported as a percentage.

3.1 Sample implementations

AM has been applied to a number of different linguistic phenomena in several different languages. Although a comprehensive enumeration is not possible here, a representative sampling of applications gives an appreciation for the breadth and depth of coverage possible.

The earliest documented AM results with language data—and indeed the preponderance of work since—focus primarily on phonology, morphology,

Homogenous Supracontext	Words in Supracontext	Pointers	# of Pointers to Masculine	# of Pointers to Feminine
p a –	*par* M, *paz* F	*par* M > *par* M *par* M > *paz* F *paz* F > *par* M *paz* F > *paz* F	2	2
– a n	*plan* M	*plan* M > *plan* M	1	0
p – –	*par* M, *paz* F	*par* M > *par* M *par* M > *paz* F *paz* F > *par* M *paz* F > *paz* F	2	2
– – n	*plan* M, *tren* M, *ron* M	(not shown)	9	0

Table 4: Non-empty homogenous supracontexts.

and lexical selection. Early applications (Skousen, 1989) included predicting Finnish past tense forms, describing allomorphic variation (e.g. the a/an distinction in English), and predicting lexical selection based on sociolinguistic variables (e.g. Arabic forms for "friend"). Subsequent efforts (Skousen et al., 2002) addressed such phenomena as German pluralization, Dutch compound linkers, Spanish gender on nouns, stress in Dutch, and Turkish consonantal alternation.

Most of these implementations involve feature vectors that specify relatively low-level linguistic features such as sound segments, phonetic or orthographic environments, syllable structure, and word boundaries. Often the information for constructing these features comes from widely-used annotated language resources such as lexicons or corpora. For example, several English, German, and Dutch applications have used features partially derived the CELEX lexical database (); others have used WordNet (for English) (), TELL (for Turkish) (), and LEXESP (for Spanish) ().

With the advent of large-scale corpora, both annotated and unannotated, increasing amounts of information are available to language modelers. Recent AM efforts have benefited from speech and text corpora. The popular TIMIT speech database provided exemplars for recent work on flapping in English (Eddington, 2007). In probably the first AM application to diachronic language description, a recent paper (Chapman and Skousen, 2005) used the Helsinki corpus for modeling the historical development of English negative prefixes.

Increasingly, modelers are using their own experimentally obtained results to compare against more standard resources. The Arabic lexical selection example involved only features encoding novel researcher-collected situational observations. A recent paper on the comparative form of English adjectives (Elzinga, 2006) matches human judgments against occurrences from the internet collected via the Google search engine.

The choice of which features to include in the exemplar and test instances is an important issue for the AM user. Since features have to be enumerated and encoded *a priori*, at least some of the features chosen should reflect salient properties of the phenomenon in question. This does not presuppose a prescient omniscience; it is not the case that all and only the relevant features must appear in the vectors. Any features that end up not contributing to the outcome simply constitute irrelevant overhead. Of course, in an exponential system such overhead should be avoided wherever possible.

3.2 Comparison to other approaches

The algorithm sketched in Section 2 is what sets the AM paradigm apart from other machine learning and language modeling approaches. The system does not employ sub-symbolic representations or continuous-valued features and thus differs in fundamental ways from connectionist systems. The computation of analogical (dis)similarity resembles in many ways nearest-neighbor approaches, though in some cases a "gang" of similar items, none of which is the nearest neighbor, may conspire to overcome the nearest neighbor and impose their own outcome. These gang effects are difficult to identify, though some have been discussed in the literature, and represent a unique aspect of AM results. When gang effects are not present, AM processing generally obtains results comparable to memory-based learning (e.g. in TiMBL). The latter, though, allows for continuous-valued features.

Another aspect where AM and some approaches differ is that AM considers each feature in any given vector of equal weight; only across the exemplar base do features emerge as more or less relevant. In this respect AM does not perform feature weighting in a separate processing stage across the exemplar base before testing new items. Similarly, no dependencies can be specified explicitly across features.

Extensive comparisons have been made between AM and other machine learning approaches including memory-based learning, version spaces, neural networks, and Optimality Theory; see (Skousen et al., 2002) for complete details.

The concept of an analogical set with quantifiable support is also unique to AM, and its relevance is being explored. For example, recent work (Chandler, 2005) has shown how AM results can map onto response times in certain types of psycholinguistic experiments.

In fact, another notable feature of AM is its ability to model human language errors and variants that seem to result from analogy. Skousen, in early AM work, discusses how "leakage" (i.e. erroneous AM guesses) often tends to follow observable usage errors (Skousen, 1989).

(??? Dave: a couple of sentences about your Spanish data...)

A comparison of corpus data for Russian verbs of motion—and subsequent modeling in AM combining lexical, morphological, and semantic variables—showed a tendency for the system's errors to coincide with those of language learners (Dodge and Lonsdale, 2006).

Certainly further comparison of the behavior observed in AM modeling tasks versus human performance constitues a promising future direction to pursue.

When considering the place of AM in the machine learning realm, one area deserves considerable more effort. The natural language processing (NLP) field has profited greatly from machine learning, in part due to the increased availability of large-scale language resources. Though early work in AM illustrated the promise of the approach to typical NLP tasks (Jones, 1996), the paradigm has been underrepresented in recent NLP shared tasks and competitive evaluations addressing such problems as word-sense disambiguation, part-of-speech tagging, shallow parsing, semantic role labeling, and phoneme classification.

4 Recent progress in AM

The AM system has undergone substantial development since the last documented release of the system (Skousen et al., 2002), and this has permitted applications in areas that would have been previously impossible. In this section we survey some of the recent AM developments and consequent research they have enabled.

4.1 Algorithmic refinement

As explained above, the AM system's basic data structure is a fully connected lattice representing (dis)agreements among data items. The original algorithm for handling the lattice involved a straightforward but exhaustive traversal and was hence very costly. The most current version of the system involves a completely new way of extracting information from the lattice.

The crucial insight is to focus on heterogeneity instead of homogeneity. For each supracontext in the lattice, heterogeneity can be locally determined by looking for plurality of outcomes and plurality of subcontexts. When heterogeneity is detected, all more general supracontexts can be automatically discarded from that point on.

Implementing this insight has resulted in a substantial speed-up in processing time, and the ability to handle a greater number of features, exemplars, and test items. Figure 1 illustrates the run time for different versions of the AM program on a typical task (English part-of-speech tagging) involving 50,000 exemplars, 1000 test items, and 58 outcomes. It shows that the previous version of the system grew exponentially as the number of variables grew (though the maximum number of variables possible was fixed at 29); the algorithm also performed particularly poorly for a vector size of 3 features. This contrasts with the current version of the system, which runs largely linearly until the current maximum vector size of 59 features.

The rewrite of the algorithm also incorporated more decomposition in the handling of the lattice, in theory laying the groundwork for a truly parallel implementation. To date, though, no concurrent parallel implementation has been developed.

Even with this improvement, exponentiality still poses a problem. Two independent efforts have been pursued recently to address the processing bottleneck issue. One involves using Monte Carlo sampling to estimate the results that traditional AM processing provides (Johnsen and Johansson, 2005). The other method is to compute the dissimilarities via a quantum algorithm (Skousen, 2005). To discuss either would be beyond the scope of this paper.

4.2 Modularization

The newest version of the AM implementation has been repackaged as a Perl module. This was done primarily to improve flexibility in using the system. Its deployment within Perl supports better cross-architecture functionality. The provision of a number of processing hooks allows for various user-programmable add-on procedures for customizing the input, matching, and output stages of processing.

One examples of the use of customized output involves the integration of AM with a cognitive modeling system for natural language called NL-Soar. In this work, which involves parsing, the cognitive agent must occasionally resolve prepositional-phrase attachment ambiguities (Lonsdale and Manookin, 2004; Manookin and Lonsdale, 2004); it does so by invoking AM to resolve the ambiguity using a PP-attachment exemplar base of previously mined by others **??**. This integration would not have been possible with prior versions of the AM system.

Sequential output example: use of these options has been helpful in profiling system performance or tailoring system output. Lonsdale's work on Farsi name vocalization and Romanization (Lonsdale, 2006) is an example of work only possible with the changes to the most recent version.

(DWL: I still need to add a little bit here...)

4.3 Lattice visualization

For many users of the AM system, insight into the operation of the AM algorithm has proven difficult, with the system operating as a black box. A new visualization tool allows users to load up their exemplars, enter test items, and step through the lattice incrementally to view the effects of matches, homogeneity/heterogeneity, and gangs.

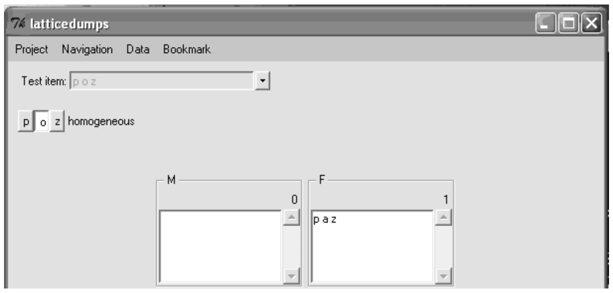

Figure 2: A screenshot of the AM lattice visualizer.

Figure 2 shows ...

4.4 Architectural improvements

Loading a large number of exemplar items is time-consuming. This is keenly felt in sequential implementations, where the outcome of one test might depend on the results of a previous one (cf. the Farsi example described above).

To alleviate this problem, a new implementation provides a client/server infrastructure. The exemplar data can be loaded on a server machine once, and th server stands by for test item queries (over a socket) from client machines.

4.5 User tools

Recent use of the system by users has resulted in two other tools worthy of note.

Development of exemplar and test item feature vectors can sometime result in ill-formed vectors which are difficult to locate. A new Perl script allows users to check their feature files and receive a report of which vectors do not comply to formatting specifications.

For users unable to configure the basic system parameters within the Perl code, use of the system can be awkward. A basic Perl/TK graphical user interface has been developed to provide users top-level

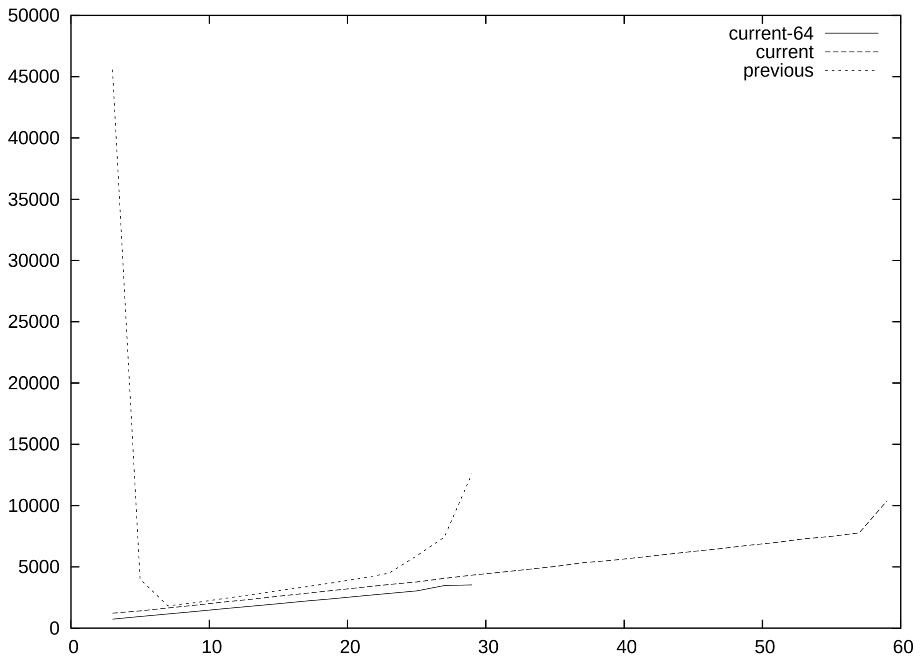

Figure 1: Performance statistics of previous and current versions of AM

control to the overall system, allowing them to select pertinent parameters and input files. In this way the users will not need to modify any Perl code.

5 Final observations

(Dave: were you going to take a short first stab at this section?)

Acknowledgements

We would like to thank Theron Stanford for his programming support work.

References

Steve Chandler. 2005. Computer simulations of RT in analogical modeling. Paper presented at the Thirty-second LACUS Forum at Dartmouth College, August 4, 2005.

Don Chapman and Royal Skousen. 2005. The negative prefix in english and analogical modeling of language. *Journal of English Language and Linguistics*, 9:333–357.

Inna Danielyan Dodge and Deryle Lonsdale. 2006. Modeling Russian verbs of motion: an analogical account. In *Proceedings of the 28th Annual Conference of the Cognitive Science Society*, pages 1239–1244. Lawrence Erlbaum.

David Eddington. 2002. Spanish gender assignment in an analogical framework. *Journal of Quantitative Linguistics*, 9:49–75.

David Eddington. 2007. Flapping and other variants of /t/ in american english: Allophonic distribution without constraints, rules, or abstractions. *Cognitive Linguistics*, 18:23–46.

Dirk Elzinga. 2006. English adjective comparison and analogy. *Lingua*, 116:757–770.

Lars Johnsen and Christer Johansson. 2005. Efficient modeling of analogy. In *Proceedings of the CiCLing Conference*, volume 3406 of *Lecture Notes in Computer Science*, pages 694–703. Springer Berlin/Heidelberg.

Daniel Jones. 1996. *Analogical Natural Language Processing*. Studies in Computational Linguistics. University College of London Press Limited, London, England.

Deryle Lonsdale and Michael Manookin. 2004. Combining learning approaches for incremental on-line parsing. In *Proceedings of the Sixth International Conference on Cognitive Modeling*, pages 160–165. Lawrence Erlbaum.

Deryle Lonsdale. 2006. Exploring syllables, Romanization, and analogy in names. In *Proceedings of the Sixth Annual Family History Technology Workshop*. BYU Computer Science Department.

Michael Manookin and Deryle Lonsdale. 2004. Resolving automatic prepositional phrase attachments by non-statistical means. In *LACUS Forum XXX: Language, Thought, and Reality*, pages 291–300. The Linguistic Association of Canada and the United States.

Royal Skousen, Deryle Lonsdale, and Dilworth B. Parkinson, editors. 2002. *Analogical Modeling: An exemplar-based approach to language*. John Benjamins, Amsterdam/Philadelphia.

Royal Skousen. 1989. *Analogical Modeling of Language*. Kluwer Academic Publishers, Dordrecht, Netherlands.

Royal Skousen. 1992. *Analogy and Structure*. Kluwer Academic Publishers, Dordrecht, Netherlands.

Royal Skousen. 1995. Analogy: A non-rule alternative to neural networks. *Rivista di linguistica*, 7:213–231.

Royal Skousen. 1998. Natural statistics in language modeling. *Journal of Quantitative Linguistics*, 5:246–255.

Royal Skousen. 2005. Quantum analogical modeling: A general quantum computing algorithm for predicting language behavior. Posted at arxiv.org.