A MODEL OF LEARNING GEOMETRIC REASONING

by

Geoffrey G. Towell
Richard Lehrer
Jude W. Shavlik

Computer Sciences Technical Report #923

March 1990

A Model of Learning Geometric Reasoning

Geoffrey G. Towell
Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
towell@cs.wisc.edu
(608) 262-6613

Richard Lehrer
Educational Psychology
University of Wisconsin
1025 West Johnson Street
lehrer@vms.macc.wisc.edu

Jude W. Shavlik
Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
shavlik@cs.wisc.edu

Keywords: modeling human learning, artificial neural networks, hybrid learning systems, correcting imperfect knowledge bases

(Submitted to the *Eleventh Annual Conference of the Cognitive Science Society.*)

A Model of Learning Geometric Reasoning

ABSTRACT

Skilled performance in geometry relies on knowledge of fundamental properties of figures, such as parallelism and congruency. However, research in mathematics education suggests that children's acquisition of this knowledge is often incomplete or underdeveloped. A stage model of the acquisition of this knowledge was suggested by the van Hieles. Unfortunately, the van Hiele model fails to specify mechanisms of stage transition, and more generally, provides snapshots of learning rather than a model of learning. In this paper we propose a model of the development of geometric reasoning that is explicitly a learning model. The model begins with an understanding of geometry similar to that of preschool children. Through the presentation of a series of examples, the model is shown to develop a more sophisticated understanding of geometry. Analysis of what the model learned from the examples is shown to make concrete pedagogical suggestions.

INTRODUCTION

Recent research about the development of skilled performance in geometry suggests that knowledge of fundamental spatial schemes is essential to the efficient construction of proofs [Koedinger and Anderson, 1990]. This view suggests the need for spatial spadework prior to deduction, in which students have opportunities to develop adequate descriptions of spatial properties and their relationships. An understanding of how this development takes place can serve to guide instruction, thereby maximizing the effectiveness of instruction during this period.

One model of the development of spatial properties and their relationships was suggested over 30 years ago by D. van Hiele-Geldof [1957] and subsequently elaborated by a variety of researchers [Fuys *et al.*, 1988; van Hiele, 1986]. Briefly stated, the van Hiele model proposes a series of levels of understanding that evolve as children engage in geometry. At the first level (level 0 in most accounts), children develop constructions about space that are closely tied to their informal knowledge. For example, they may sort shapes on the basis of contour. As a result, their thought is often characterized as "visual." Thus, squares that are similar in appearance to prototypical squares in the child's experience are identified as squares, but squares that depart significantly from the prototype's size or orientation are not classified appropriately.

At the second level (level 1) of the van Hiele model, thinking about space becomes more symbolic. Just as children's use of numeric symbols extends the range of their thought in addition and subtraction problems, so too does the use of symbolic properties extend their ability to reason

about objects in space. Thus, "it looks like" is gradually replaced with "it has all right angles", and the like. Practically, this means that children can describe commonalities and differences among objects that may not "look" alike. For example, squares may be characterized as figures with congruent sides and right angles. Recognition of these characteristics leads to an increase in classification accuracy even for squares that are markedly different from prototypical examples.

At the third level (level 2), children construct relationships among symbolic descriptions (properties). For example, they recognize that if the opposite sides of a quadrilateral are parallel, then the opposite sides must be congruent. Similarly, children develop ideas about relationships among figures; a square is a rhombus because it has the properties of a rhombus. The model from here progresses to describe two higher levels of axiomatic reasoning and deduction.

Although the van Hiele model has generated much research and continues to attract many adherents, there is good reason to question its adequacy as an explanation of the transitions observed in children's reasoning about geometry. First, the van Hiele model describes general benchmarks of thought. Although this type of benchmark can be useful to teachers in their design of classroom instruction and for assessment, it suffers from too much generality. To say that a student's understanding is limited to a "visual" analysis tells little about how such an analysis is conducted.

Second, the van Hiele model fails to specify any mechanisms for transition between levels (stages). For instance, there is not any principled account of why some types or contrasts among examples may be more conducive to learning than others.

Third, there is little empirical evidence to substantiate the demarcations of children's thinking into discrete and nonoverlapping levels, even by adherents of the theory.

Lastly, a series of studies conducted with stimuli such as those displayed in Figure 1 questioned second, fourth, and fifth grade children about similarities and differences among these shapes [Lehrer *et al.*, 1989]. These studies suggested that children's descriptions of space use many of the same constructs used by experts, but that they also include features without diagnostic significance. For example, the majority of children at each of these grades report that shapes B and C are most similar, either because "they are both pointy", or because "you can move this line down here [the lower segments of B] and then they [B and C] would look alike." When questioned further, many children will assert that the shape C is a triangle and shape B "isn't really a triangle." Nevertheless, children go on to reassert their similarity. Thus, shape names do not imply a necessary set of properties for children the way they do for adults.

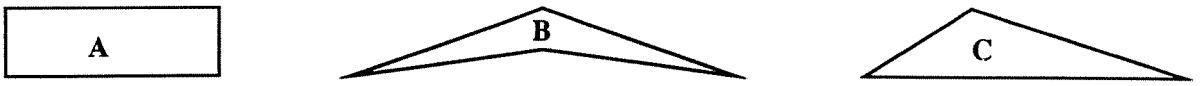

Figure 1: Sample triad.

We suggest that a model of the development of spatial features and their relationships should have the properties listed in Table 1. The following section describes the model we are developing and provides an initial suggestion that our model has these properties. Subsequent sections describe a test of this model and the implications of this test on instruction. The paper concludes with a description of current research issues for the model.

(1)	Show "mixture" between von Hiele levels.
(2)	Have a mechanism for skill development via incremental learning.
(3)	Allow for a combination of "visual" and ordinary features.
(4)	Treat shape names initially as features and later have them acquire diagnostic significance.
(5)	Combine explanation (rules) and pattern recognition/identification.

Table 1: Necessary properties of a model.

THE MODEL

The model we propose develops geometric reasoning by learning to identify the most similar pair of shapes in triads akin to Figure 1. It is based upon the KBANN *(Knowledge-Based Artificial Neural Networks)* computer system being developed at the University of Wisconsin [Shavlik and Towell, 1989; Towell *et al.*, 1990]. Briefly, KBANN uses a knowledge base of hierarchically structured rules, which may be both incomplete and incorrect, to form an artificial neural network (ANN) [Rumelhart *et al.*, 1986]. We do not mean this to be a model at the neurophysiological level. Rather, we intend it to be a model of how preconceived notions are incrementally adjusted through experience.

KBANN KBANN translates knowledge bases into ANNs for which the core layout is isomorphic to the knowledge bases; the correspondences are listed in Table 2. Values for the inter-unit weights and the thresholds of the units are set such that the ANN initially responds in the same manner as the knowledge base of inference rules.

As an example of the translation process in KBANN, consider the knowledge base presented in Figure 2a; PROLOG notation is used.[1] Figure 2b illustrates the hierarchical structure of these

[1]The top rule in Figure 2a reads "A is true if both B and C are true."

Knowledge Base	*Neural Network*
Final Conclusions	Output Units
Supporting Facts	Input Units
Intermediate Conclusions	Hidden Units
Dependencies	Weighted Connections

Table 2: Correspondences between knowledge-bases and neural networks.

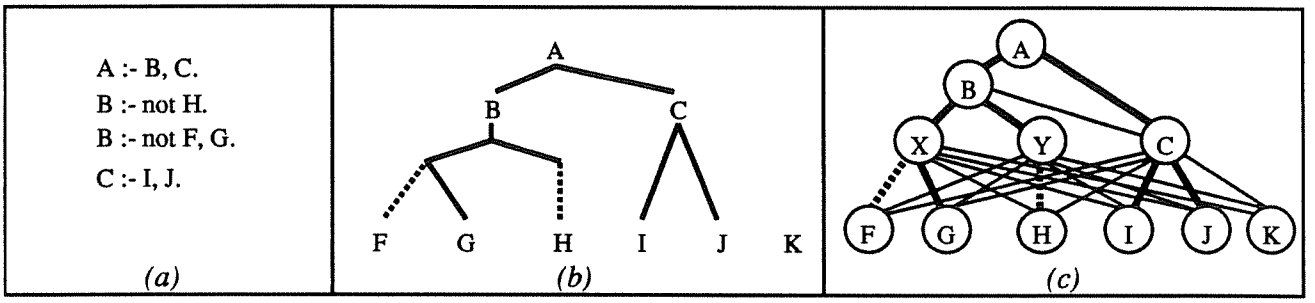

Figure 2: Translation of a knowledge-base into a network.

rules, with solid and dotted lines representing, respectively, antecedents which must and must not be present. Figure 2c represents the network that results from the translation into a neural network of this knowledge base. Thick lines in Figure 2c represent links in the neural network that correspond to dependencies in the explanation. Thin lines represent links added to the network to allow refinement of the domain theory; all these links initially have weights near zero. Units X and Y in Figure 2c were added to handle the disjunction involving predicate B. (See [Towell *et al.*, 1990] for details of the translation process.)

Level 0 Geometric Theory The model is initialized with a set of rules that roughly correspond to the theory of geometry held by children at level 0 on the von Hiele scale. This set of rules, designated LØ, implements a *feature counting* strategy in which similarity is judged according to the number of visual features shared by a pair of shapes. 7 visual features are counted by LØ: *tilted* (rotation from a standard orientation) *slanty-lines* (presence of lines other than vertical and horizontal), *physical description*, *pointy* (the figure appears to point in some direction) *point direction*, *area*, and *two long and two short sides* (typical of rectangles).

A subset of LØ appears in Table 3A.[2] The first rule in this table, rule 3.1, says that if the area of a shape (?OBJ1)[3] has some value (e.g., large) and the area of another shape (?OBJ2) has the same

[2]Tables 5 – 9 in the Appendix contain a complete listing of LØ.

[3]?X denotes a variable.

```
(1)  same-area(?OBJ1, ?OBJ2)     :- area(?OBJ1, ?AREA),          area(?OBJ2, ?AREA).
(2)  same-pdir(?OBJ1, ?OBJ2)     :- point-direction(?OBJ1, ?PD), point-direction(?OBJ2, ?PD).
(3)  same-pointy(?OBJ1, ?OBJ2)   :- pointy(?OBJ1, ?PTY),        pointy(?OBJ2, ?PTY).
(4)  very-similar(?OBJ1, ?OBJ2)  :- same-pdir(?OBJ1, ?OBJ2),    same-pointy(?OBJ1, ?OBJ2),
                                    same-area(?OBJ1, ?OBJ2).
(5)  similar(?OBJ1, ?OBJ2)       :- same-pdir(?OBJ1, ?OBJ2),    same-pointy(?OBJ1, ?OBJ2).
(6)  similar(?OBJ1, ?OBJ2)       :- same-pdir(?OBJ1, ?OBJ2),    same-area(?OBJ1, ?OBJ2).
(7)  similar(?OBJ1, ?OBJ2)       :- same-pointy(?OBJ1, ?OBJ2),  same-area(?OBJ1, ?OBJ2).
(8)  less-similar(?OBJ1, ?OBJ2)  :- same-pdir(?OBJ1, ?OBJ2).
(9)  less-similar(?OBJ1, ?OBJ2)  :- same-area(?OBJ1, ?OBJ2).
(10) less-similar(?OBJ1, ?OBJ2)  :- same-pointy(?OBJ1, ?OBJ2).
```

A: A small, feature-counting theory of geometry.

```
(11) triangle(?OBJ) :- area(?OBJ, medium), pointiness(?OBJ, very), point-direction(?OBJ, up).
(12) triangle(?OBJ) :- number-of-sides(?OBJ1, 3).
```

B: Rules to recognize triangles.

```
(13) most-similar(?OBJ1, ?OBJ2) :- very-similar(?OBJ1, ?OBJ2), not very-similar(?OBJ1, ?OBJ3),
                                                               not very-similar(?OBJ2, ?OBJ3).
(14) most-similar(?OBJ1, ?OBJ2) :- similar(?OBJ1, ?OBJ2),      not similar(?OBJ1, ?OBJ3),
                                                               not similar(?OBJ2, ?OBJ3).
(15) most-similar(?OBJ1, ?OBJ2) :- less-similar(?OBJ1, ?OBJ2), not less-similar(?OBJ1, ?OBJ3),
                                                               not less-similar(?OBJ2, ?OBJ3).
```

C: Rules for combining similarity judgments.

Table 3: Sample rules in LØ.

value, then the two shapes have the same area. Rule 3.4 is an example of a feature counting rule.[4] Using the rules in Table 3A, shapes B and C of Figure 1 are *very-similar* since they are the same in terms of *pointiness*, *point-direction*, and *area*. On the other hand, shapes A and C are *less-similar* as they only have the same *area*.

In addition to the rules characterized in Table 3A, LØ includes shape naming rules (Table 3B). These rules match shapes against one or more prototypical instances for a shape name, rather than against symbolic descriptions of the shape names. Thus, a rule to recognize a triangle would look more like rule 3.11 than the more commonly accepted rule 3.12. As suggested by property (3) of Table 1, the shape naming rules participate in similarity judgments in LØ in the same manner as

[4] Feature counting rules are actually expressed in the form "A is true if N out of these M antecedents are true."

the visual features. So, the recognition that shapes A and B in Figure 1 are both quadrilaterals would count no more than shapes B and C having the same area.

To make explicit the relation between pairs and between levels of similarity as determined by the feature counting rules, LØ must also have rules similar to those in Table 3C. Using the rules in Table 3 it would be possible to conclude that, for Figure 1, the pair BC is the *most similar* as BC is *very similar* while neither AB nor AC is *very similar.*

Details of the implementation The rules presented in Table 3 have been expressed using variables for the sake of brevity. However, KBANN currently cannot handle variables. This has two effects. First, rules must be explicitly repeated for each pair of shapes. Hence, LØ contains about 250 rules. Second, it is possible for an ANN to learn to treat each pair of shapes differently. To prevent this, ANNs are given explicit instructions that all pairs are to be looked at in the same way. Within an ANN, this forces corresponding links to have equal weights.

In addition to the instruction to treat all pairs equally, the model is instructed to only make changes to the ANN in places that correspond to rules that consider the shapes. That is, the ANN can change neither the feature counting nor combining rules of Table 3. These two instructions constrain the problem and, we argue, make the learning problem for the model very similar to the problem faced by school children.

In addition to the 7 visual features, the model uses 13 symbolic features from proof geometry such as *number of sides*, *number of right angles.* The 20 features have an average of 4.2 possible values. (See Table 10 in the Appendix for a listing of all features and values.)

EXPERIMENT USING THE MODEL

As an initial experiment designed to test our model, we chose to train the model using two different sets of training materials. This experiment was chosen to highlight the differences between our model and the van Hiele model. Specifically, the van Hiele model provides no basis for distinguishing the effectiveness of two instructional sequences. By contrast, our model can test the effectiveness of multiple instructional sequences through the independent presentation and testing of each sequence.

For this experiment, we developed two sets of training shapes from which triads were selected. On each learning trial, the model was freshly restarted so that learning based upon one training set did not bias the other. The effectiveness of each training set was tested using a set of triads drawn from a third collection of shapes.

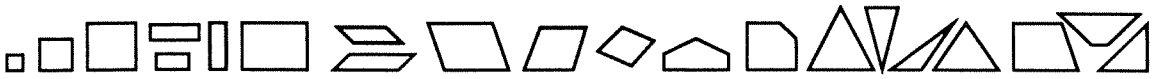

Figure 3: Representative textbook shapes.

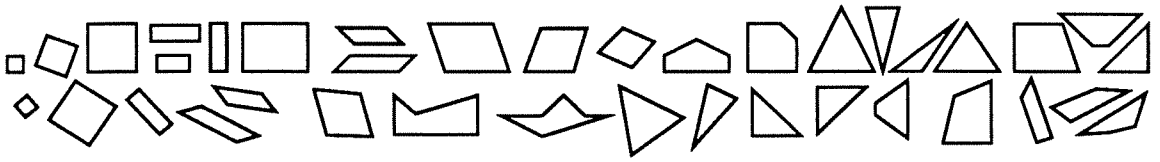

Figure 4: Representative shapes encountered using a modified version of LOGO.

Training Data The first set of shapes, consisting of 36 items, was derived from the geometry section of a fifth-grade textbook [Sobel, 1987]. The shapes (some of which appear in Figure 3) are almost all oriented so that one side is horizontal. Following the presentation format of the textbook, shapes with different number of sides were never presented in the same triads. The second set of shapes (some of which appear in Figure 4) consists of 81 items which might be produced by a child using a modified version of LOGO. This "LOGO" set lacks the orientation bias of the "textbook" set and allows for a free mixture of shapes in triads.

A collection of 33 triads were selected from each set to train the model. Using an equal number of triads in the LOGO and textbook sets is probably unfair since LOGO affords students the opportunity to see not only a more diverse set of shapes, but also to make more comparisons between shapes. However, holding the number of training examples constant makes it possible to judge solely the effect of the increased diversity that LOGO makes possible.

Testing Data The test set consists of 27 items each of which was used in a single triad. The nine resulting triads depicted in Figure 5 have also been used to test the ideas of geometry of second and fifth grade students [Lehrer *et al.*, 1989]. The second grade sample consisted of 47 students who were presented the these triads and asked to indicate which two were most alike and why. The fifth grade sample consisted of 28 children who received two weeks of geometry instruction with LOGO.

RESULTS AND DISCUSSION

Results Table 4 presents the answers generated by the model and the modal choices of children at each grade level. Responses by the model of "AB or AC" indicate that the model considers the pair AB and AC to be equally similar. This occurs when the the similarity valuation of two pairs of shapes differs by less than 5%. For the children's responses "AB / BC" indicates

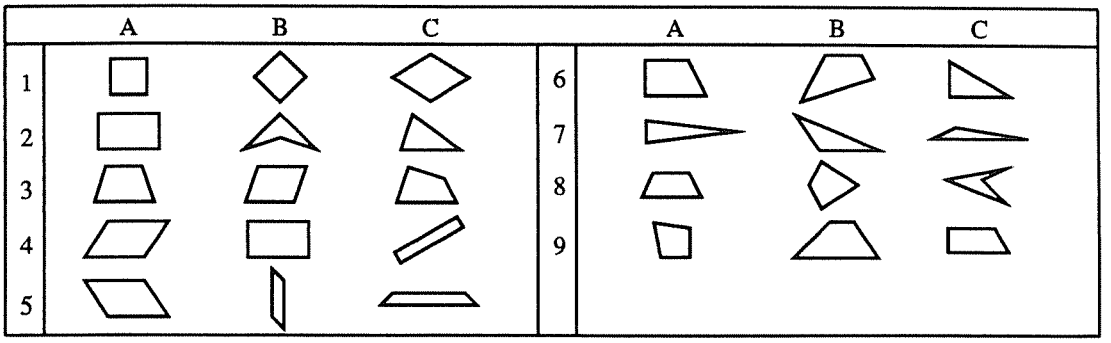

Figure 5: Triads used to test learning.

Triad Number	"Correct" Response	Response of Model After No Training	Response of Model After "Textbook" Triads	Response of Model After "LOGO" Triads	Children's Response Second Grade	Children's Response Fifth Grade with LOGO
1	AB	BC	BC	BC	BC	AB / BC
2	AB	BC	BC	AB	BC	AB
3	AB	AC	AC	AB	AC	AC / AB
4	BC	BC or AB	BC	BC or AB	BC / AB	BC
5	AB	BC	BC	AB	BC	AB
6	AB	BC	AB	BC	???	AB / BC
7	BC	AC	AC	AB or AC	AC	AC
8	AB	BC	AB	AB	BC	BC / AB
9	BC	AC	AC	BC	AB	BC

Table 4: Results of training.

that AB and BC were, respectively, the first and second most common responses. "???" indicates that all responses were equally common.

Using only the conception of geometry expressed in LØ, the model failed to uniquely identify the "correct" most similar pair for all of the testing triads. After training on the textbook examples, the model correctly classified three of the nine testing triads. After training on the LOGO examples, the model was correct on five of the nine test triads and uncertain between the correct answer and an incorrect answer on one other triad.

Responses of the model and children As might be expected, before training the model, LØ matched the modal choices of the second grade sample for nearly all nine triads. This merely indicates that the proposed starting point for the model has an empirical basis.

Table 4 shows a similarity between the answers generated by the model after LOGO training and the responses of fifth graders after using LOGO. On the seven triads for which the model provided

definite answers, that answer agrees with the most common response of the fifth graders on three triads and with their second most common response on the remaining four triads. When the model did not provide a single answer, one of the two pairs selected by the model was the most common response of the fifth graders.

It should be noted that the LOGO training data are being compared to a group of students who used LOGO for a comparatively brief duration. Closer matches might occur with a longer exposure (e.g., our training set may be overestimating what the children actually generated, especially for triangles). On the other hand, the textbook examples may be overrepresented because they were arranged in a context for training that was probably more extensive than that provided by the text. (This would not be true for LOGO because one can generate multiple instances on the screen for comparison.)

Analysis of learning by the model Our model allows deeper comparisons than mere number of test examples correct because the ANN following learning can be understood through comparison to its starting state [Towell *et al.*, 1990]. Our plans are to use this sort of analysis as the basis for detailed comparisons of learning by the model and by children. However, the analysis presented here looks only to explain why the model acts as it does after learning.

For example, after LOGO training the model was incorrect on triad 1 because although it learned to ignore *slanty lines* and *pointiness*, the model continued to rely upon *tilted*. To correct this error, triads could be added to the training set which highlight the importance *all angles congruent* and *all sides congruent* and the irrelevance of *tilted*. Also after LOGO training, the model was correct on triad 8 because it applied a newly learned feature *convexity* while ignoring features such as *pointiness* and *area*. The model was unable to make unique decisions on triads 6 and 7 because it mixed visual with symbolic features when making its decisions. Thus, the model acts as if is at van Hiele level 0 on triad 1, level 1 on triad 8, and somewhere between on triads 6 and 7.

Further analysis of the model after LOGO training suggests that it began to discover the sort of relationships characteristic of van Hiele level 2. For instance, consideration of the number of sides was clearly linked to consideration of the number of angles. In other words, the network learned that these two features, which have no necessary relation, are the same for closed figures. So, depending on the problem, the model might give a response characteristic of any of van Hiele levels 0, 1, or 2 or some combination of these levels.

Analysis of the model after training on the textbook triads reveals that these triads largely failed to move the model towards level 1. While the model learned to ignore the visual features *area* and *point direction*, it still relied upon other visual features such as *pointy* and *2 long and 2 short sides*. These changes can be traced directly to the characteristics of the textbook training set. *Point direction* and *area* are eliminated because the textbook set makes this contrast quite effectively. Conversely, *pointy* remained significant because triangles which are all *pointy* were never presented in a triad with *pointy* non-triangles. Hence, the model never learned that a pair of figures can be pointy and yet not be similar in a meaningful way.

Despite the inability of the textbook examples to move the model towards level 1, the model learned rules characteristic of level 2. For instance, a rule developed for recognizing squares which used a combination of *number of right angles*, *all angles equal* and *number of lines of symmetry*. Unfortunately, while this rule indicated that the pair AB for test triad 1 was the most similar, it was overshadowed by the judgments of visual features.

CURRENT RESEARCH ISSUES

Rules such as 3.11 are intended to let the model recognize shapes by relationship to prototypes and to treat these shape names as equivalent to other visual features. However, these shape-naming rules were never generalized. As a result, shape names did not acquire diagnostic significance under either set of training examples. Nevertheless, after LOGO training the model learned relationships outside of the provided shape naming rules which were specific to pentagons, hexagons, and octagons. Thus, the network was able to learn to general shape definitions, but was unable to do so from the supplied prototypes. We are investigating how the model can be modified to encourage the use and modification of the shape prototypes. One approach we intend to pursue is to directly train the hidden units related to the shape-naming rules. Through this explicit training, the shape naming rules should generalize from the supplied prototypes.

This explicit training of shape naming will allow us to modify the training style for textbook shapes to more closely reflect the content of textbooks. Realistically, textbooks do not present triads; instead single shapes are presented and their most specific name is given. The type of training implied by textbook presentation style is not possible for the model as described in this paper. As a result, future comparisons of textbook and LOGO training might be more accurate.

Finally, we plan expand significantly on the comparison of the development of geometric reasoning in our model and in children. Initially, this effort will involve making more detailed comparisons

of responses from our model and from children on sets of test triads as suggested in the discussion section. Following this work, we plan to simulate actual instruction given to children and compare the learning of the model to that of children. Part of this effort will require augmenting our model with the ability to add new rules after training has commenced. Through this addition, we will be able use our model to test the combined effect of direct instruction with example presentation.

CONCLUSIONS

In the introduction, we proposed a list of five necessary properties of a model of geometry learning (Table 1) and argued that the van Hiele model does not have these properties. Subsequently we proposed a model for geometry learning based upon the KBANN machine learning algorithm. In the course of this paper we have shown that our model has four of the five properties, and have suggested that enhancements to our model that will give it all five properties.

We also proposed that a model should be able to make specific suggestions about the content of classroom instruction and criticized the van Hiele model for its inability to do so. In contrast, the experiment showed that, using our model, it is possible to determine the effectiveness of a training sequence and to make specific pedagogical recommendations.

ACKNOWLEDGMENTS

We wish to acknowledge the support of the University of Wisconsin Graduate School and the Ameritech Foundation.

References

[Fuys *et al.*, 1988] D. Fuys, D. Geddes, and R. Tischler. The van Hiele model of thinking in geometry among adolescents. In *Journal for Research in Mathematics Education, Monograph No. 3.* National Council of Teachers of Mathematics, Reston, VA, 1988.

[Koedinger and Anderson, 1990] K. R. Koedinger and J. R. Anderson. Theoretical and empirical motivations for the design of ANGLE: A new geometry learning environment. In *Proceedings of the AAAI Symposium on Knowledge-Based Environments for Learning and Teaching*, Stanford, CA, March 1990.

[Lehrer *et al.*, 1989] R. Lehrer, W. Knight, M. Love, and L. Sancilio. Software to link action and description in pre-proof geometry. Presented at the Annual Meeting of the American Educational Research Association, March 1989.

[Rumelhart *et al.*, 1986] D. E. Rumelhart, G. E. Hinton, and J. R. Williams. Learning internal representations by error propagation. In D. E. Rumelhart and J. L. McClelland, editors, *Parallel Distributed Processing, Vol. 1*, pages 318–362. MIT Press, Cambridge, MA., 1986.

[Shavlik and Towell, 1989] J. W. Shavlik and G. G. Towell. An approach to combining explanation-based and neural learning algorithms. *Connection Science*, 1:233–255, 1989.

[Sobel, 1987] M. A. Sobel, editor. *Mathematics.* McGraw-Hill, New York, 1987.

[Towell *et al.*, 1990] G. G. Towell, J. W. Shavlik, and M. O. Noordewier. Refinement of approximate domain theories by knowledge-based artificial neural networks. In *Proceedings of the Seventh International Conference on Machine Learning*, Austin. TX, June 1990.

[van Hiele-Geldof, 1957] D. van Hiele-Geldof. *De didaktick van de Meetkunde in de eerste klass van het (The Didactics of Geometry in the Lowest Class of Secondary School).* PhD thesis, University of Utretch, 1957. English translation by M. Verdonck.

[van Hiele, 1986] P. M. van Hiele. *Structure and insight.* Academic Press, New York, 1986.

APPENDIX — Complete Listing of Rules and Features

name(triangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,somewhat), point-direction(?obj,up), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(triangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,very), point-direction(?obj,up), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(triangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,little), pointy(?obj,somewhat), point-direction(?obj,right), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(triangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,big), pointy(?obj,somewhat), point-direction(?obj,up), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(triangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,very), point-direction(?obj,down), shape(?obj,skinny), 2-long-and-2-short-sides(?obj,no).
name(triangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,little), pointy(?obj,somewhat), point-direction(?obj,down), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(pentagon,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(pentagon,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,large), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(hexagon,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(hexagon,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,large), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(quadrilateral,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,big), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(quadrilateral,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).

Table 5: Shape naming rules — part I.

name(square,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(square,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,big), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(rectangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,medium), 2-long-and-2-short-sides(?obj,yes).
name(rectangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,skinny), 2-long-and-2-short-sides(?obj,yes).
name(rectangle,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,no), area(?obj,medium), pointy(?obj,not), point-direction(?obj,none), shape(?obj,fat), 2-long-and-2-short-sides(?obj,yes).
name(rhombus,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,somewhat), point-direction(?obj,right), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(rhombus,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,little), pointy(?obj,somewhat), point-direction(?obj,up), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(parallelogram,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,somewhat), point-direction(?obj,left), shape(?obj,medium), 2-long-and-2-short-sides(?obj,yes).
name(parallelogram,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,big), pointy(?obj,somewhat), point-direction(?obj,right), shape(?obj,fat), 2-long-and-2-short-sides(?obj,yes).
name(parallelogram,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,little), pointy(?obj,very), point-direction(?obj,left), shape(?obj,skinny), 2-long-and-2-short-sides(?obj,yes).
name(trapezoid,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,very), point-direction(?obj,right), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(trapezoid,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,medium), pointy(?obj,somewhat), point-direction(?obj,up), shape(?obj,medium), 2-long-and-2-short-sides(?obj,no).
name(trapezoid,?obj)	:-	tilted(?obj,0), slanty-lines(?obj,yes), area(?obj,big), pointy(?obj,somewhat), point-direction(?obj,left), shape(?obj,fat), 2-long-and-2-short-sides(?obj,no).

Table 6: Shape naming rules — part II.

same-name(?obj1,?obj2)	:-	name(?nme,?obj1), name(?nme,?obj2).
same-tilted(?obj1,?obj2)	:-	tilted(?obj1,?tlt), tilted(?obj2,?tlt).
same-slanty-lines(?obj1,?obj2)	:-	slanty-lines(?obj1,?slant), slanty-lines(?obj2,?slany).
same-area(?obj1,?obj2)	:-	area(?obj1,?area), (?obj2,?area).
same-pointy(?obj1,?obj2)	:-	pointy(?obj1,?pty), pointy(?obj2,?pty).
same-point-direction(?obj1,?obj2)	:-	point-direction(?obj1,?pd), point-direction(?obj2,?pd)
same-shape(?obj1,?obj2)	:-	shape(?obj1,?shp), shape(?obj2,?shp).
same-2-2(?obj1,?obj2)	:-	2-long-and-2-short-sides(?obj1,?two), 2-long-and-2-short-sides(?obj2,?two).

Table 7: Feature counting rules.

Let *8-Antecedents* represent the following set of eight antecedents: same-2-2(?obj1,?obj2), same-shape(?obj1,?obj2), same-point-direction(?obj1,mkvobj2), same-pointy(?obj1,?obj2), same-area(?obj1,?obj2), same-slanty-lines(?obj1,?obj2), same-tilted(?obj1,?obj2), same-name(?obj1,?obj2)		
similarity(?obj1,?obj2,very)	:-	n-true-antecedents(7, *8-Antecedents*).
similarity(?obj1,?obj2,quite)	:-	n-true-antecedents(6, *8-Antecedents*).
similarity(?obj1,?obj2,mostly)	:-	n-true-antecedents(5, *8-Antecedents*).
similarity(?obj1,?obj2,fairly)	:-	n-true-antecedents(4, *8-Antecedents*).
similarity(?obj1,?obj2,sort-of)	:-	n-true-antecedents(3, *8-Antecedents*).
similarity(?obj1,?obj2,not-very)	:-	n-true-antecedents(2,*8-Antecedents*).

Table 8: Similarity recognition rules.

most-similar-pair(?obj1,?obj2)	:-	similarity(?obj1,?obj2,very), not similarity(?obj1,?obj3,very), not similarity(?obj3,?obj2,very).
most-similar-pair(?obj1,?obj2)	:-	similarity(?obj1,?obj2,quite), not similarity(?obj1,?obj3,quite), not similarity(?obj3,?obj2,quite).
most-similar-pair(?obj1,?obj2)	:-	similarity(?obj1,?obj2,mostly), not similarity(?obj1,?obj3,mostly), not similarity(?obj3,?obj2,mostly).
most-similar-pair(?obj1,?obj2)	:-	similarity(?obj1,?obj2,fairly), not similarity(?obj1,?obj3,fairly), not similarity(?obj3,?obj2,fairly).
most-similar-pair(?obj1,?obj2)	:-	similarity(?obj1,?obj2,sort-of), not similarity(?obj1,?obj3,sort-of), not similarity(?obj3,?obj2,sort-of).
most-similar-pair(?obj1,?obj2)	:-	similarity(?obj1,?obj2,not-very), not similarity(?obj1,?obj3,not-very), not similarity(?obj3,?obj2,not-very).

Table 9: Combining rules.

Feature Name	*Possible Values*
Visual Features	
Tilted	0 10 20 30 40
Slanty	Yes No
Area	Little Medium Big
Shape	Skinny Fat Medium
Pointy	Yes No
Point Direction	None Up Down Right Left
2 long and 2 short sides	Yes No
Symbolic Features	
Convex	Yes No
Number of Sides	3 4 5 6 8
Number of Angles	3 4 5 6 8
Number of Right Angles	0 1 2 3 4
Number of Pairs of Parallel Sides	0 1 2 3 4
Number of Pairs of Equal Opposite Angles	0 1 2 3 4
Adajacent Angles Sum to 180	Yes No
Number of Pairs of Opposite Sides Equal	0 1 2 3 4
Number of Lines of Symmetry	0 1 2 3 4 5 6 8
All Sides Equal	Yes No
All Angles Equal	Yes No
Number of Equal Sides	0 2 3 4 5 6 8
Number of Equal Angles	0 2 3 4 5 6 8

Table 10: Features and their possible values.