# THE USE OF ARTIFICIAL INTELLIGENCE IN THE AUTOMATED PLACEMENT OF CARTOGRAPHIC NAMES

David S. Johnson
Dr. Umit Basoglu
Intergraph Corporation
One Madison Industrial Park
Huntsville, AL 35807-4201

## ABSTRACT

The placement priorities of cartographic names differ from one mapping agency to another. They also differ from one product to another within a single agency. Automated name placement software needs to be flexible enough to handle varying needs. Rule based processing provides this flexibility. Using product rules, placement is achieved for point, line and area features and conflicts pertaining to text are detected and resolved. The development of the product rules can be a very time-consuming process. The time necessary to develop these rules can be reduced through the use of neural network technology.

## INTRODUCTION

In the past, the placement of cartographic names has been a manual process which has generally consumed up to fifty percent of the map preparation time (Yoeli, 1972). More recently, cartographers have sought the help of a computer to reduce the time required to annotate maps. Many different algorithms have been developed to aid in this process. Among them are the priorities suggested by Yoeli (1972) for the placement of names for point features, the wavefront approach used for the placement of names for area features (Montanari, 1969), and the placement for linear names (Basoglu, 1984). While these are some of the more recent algorithms, they do not meet all of the needs of the cartographer.

The priorities suggested by Yoeli specify where the optimum text placement location is (e.g. upper-right), and the order that other locations should be tried when conflicts prevent the placement of the text. Yoeli's priorities work well when used for the placement of names associated with populated places. However, other cartographers prefer to use a different set of rules for that task. In addition, there is issue of what rules should be used for other types of point features. For example, the rules for the placement of elevation values associated with control points usually differ from the rules for the placement of names associated with populated places. Similarly, there are many alternative ways for labeling various area features and linear features.

The rules for labeling a feature depend on both the text placement philosophy of the cartographer who is positioning the text and the type of feature being processed. This means that any system designed to automate this process must be robust enough to support several placement algorithms and flexible enough to allow the cartographer to dictate when a particular algorithm should be used. The system should also be expandable to support the addition of new placement algorithms. Finally, the system should be intelligent enough to aid the cartographer in generating the placement rules based on existing maps. Through the use of artificial intelligence techniques, such a system is feasible.

According to Klimasauskas (1988), the major information processing techniques are "procedural languages (BASIC, FORTRAN, COBOL, C), expert systems, and neural networks". A combination of all three is needed to develop a system that meets the requirements stated above.

## THE USE OF PRODUCT RULES

An expert system is a system with a high degree of skill or knowledge about a certain subject. Two important roles in the development of an expert system are the role of the knowledge engineer (the person who implements the system) and the role of the domain expert (the person who is an expert in the domain of the task to be performed by the expert system). To implement an expert system, a knowledge engineer must be knowledgeable about the current techniques being used by the domain expert. Knowledge engineers gain their knowledge by having direct consultations with domain experts, by reading material written by domain experts, and/or by studying the results produced by domain experts.

The knowledge engineer and the domain expert often work together to come up with a list of the basic functionalities needed to perform the desired task. After the basic functionalities have been defined, it is the knowledge engineer's job to write the software that supports those functionalities. The domain expert writes or assists in writing a set of rules that specify when a particular functionality is to be used. For example, some information for en route airways on navigational charts is to be placed along the airway at a specified distance from the airport while other information is to be placed near the midpoint of the airway. The basic functionality needed to support this is placing text along a linear feature at a specified distance from a given endpoint of that feature and placing text at the midpoint of a linear feature. The airways example illustrates two functionalities commonly needed when processing linear features. If the system is to be used for more than labeling airways, then many more different functionalities will be needed.

The knowledge engineer and the domain expert should identify and incorporate as much functionality into the system as feasibly possible when the system is being designed. Chances are, as time goes by, more functionalities will be requested by the users of the system. In addition, some of the original algorithms may become outdated. Because of this, the system needs to be able to support the addition of new algorithms and the revision and replacement of existing algorithms. This is easily handled by the use of rulesets.

A simple example of what a rule might look like is as follows:

```
If a feature has a feature_code of "xx...x" and
     has a population greater than "50000"
then
     send load_text to the "text_processor" using city_name
     send compute_point_text_location to the "text processor" using "GT_50000"
     send place_text to the "text_processor" using feature
```

The "if" clause ensures that only cities with populations greater than 50,000 will be processed by the then clause of this rule. The "then" clause loads the city name, computes the position of the text based on a parameter code (GT_50000), and then places the text, linking it with the point feature. The parameter code is used by the compute_point_text_location algorithm as a look-up into a table of parameters containing information about point features. This information contains, among other things, the symbology and the optimum placement location to be used when placing the text, optional placement locations to be used in case a conflict occurs when attempting to place the text in the optimal location, and lists of what features and colors are to be considered as conflict. The algorithm will never place text on top of existing text.

If the compute_point_text_location algorithm is enhanced, the ruleset will remain unchanged unless the arguments to the algorithm change (in the example above, the parameter code is the argument to the algorithm). If the user decides to use a different algorithm for labeling cities with a population greater than 50000, then two things must be done:

1) the new algorithm must be written and added to the existing set of functionalities
2) the ruleset must be modified to invoke the new algorithm instead of the current algorithm.

Much progress has been made in defining the necessary functionalities for placing map text and in developing computer algorithms which implement these functionalities (Freeman and Ahn, 1984). Although mapping agencies may want to implement their own placement algorithms, the algorithms suggested by Freeman and Ahn provide an excellent start for the implementation of an expert system.

Once a functionality has been added, it can be used for many different types of features. The domain expert can mix and match functionalities as needed, without affecting the underlying code. If a mapping agency decides to add a different type of map to its product line, then most, if not all, of the needed functionality will already exist. All that will be necessary is the creation of a ruleset specific to that type of map.

## THE USE OF NEURAL NETWORK TECHNOLOGY

One area of artificial intelligence that is becoming more and more popular is neural networks. In the past, some people have viewed neural networks as a competing technology of expert systems. However, these two technologies can be combined to complement each other. Conceivably, neural networks could be used to assist the domain expert in the generation of product rulesets.

Neural networks are good at examining existing data and forming assumptions based on that data. They accomplish this by means of assigning weighted values to each functionality. Some experts say the human brain operates in much the same way. The following is an excerpt from Klimasauskas'(1988) explanation of neural networks: "The basic information-processing unit in neural networks is the processing element, or neuron. These terms are used interchangeably. Brain researchers have identified over 100 different kinds of neurons. Processing elements also come in a variety of types. Each type, either in artificial or biological neural networks, operates in a specific way which assists in implementing a specific function. Common to all of them is that one or more inputs are modulated by connection weights to change the stimulation level internal to the neuron. Based on this internal stimulation or activation level, the neuron may or may not produce an output. The output is related to the internal activation level, but this relationship may be a non-linear or discontinuous function".

This same principle can be applied when devising a method of assisting the cartographer with the generation of product rules based on existing maps. By examining an existing placement location and the conditions surrounding the choice of that placement location, assessing the probability that the location is the optimum location, and storing this probability as a weight in the neural network, it is possible to "learn" the rules used by the cartographer when placing the text.

For example, if a point feature has text associated with it, the area surrounding the point can be examined to infer why the point was labeled in the manner that it was labeled. A simple case would be an oil well symbol with no other features close enough to the symbol to cause a conflict with any of the other point text placement positions. If there are no potential conflicts in the proximity of the oil well, then it may be inferred that the placement location used to label the oil well represents the optimum placement for text associated with oil wells. This is because the cartographer was not limited by any conflicts when deciding where to place the text. Since the cartographer could have chosen any of the placement locations, it can be inferred that the text was placed at the optimum placement location.

It is assumed that the functionality of the point placement algorithm includes ten different locations for the placement of point text. The goal of the neural network is to examine the map and to assign a weight to each of the ten locations. Based on the resulting weights, the rules and parameters used to label the point features can be calculated. The next step is to devise a means of computing the weight of each location. All weights should initially be set to zero. As each point is processed, the text placement location should be determined, a weight for that location should be computed, and that weight should be used to compute an average weight for that point text placement location. The weight should be computed as the number of possible placement locations minus the number of locations which could not be considered because of conflicts.

A weight of ten would be assigned to the location described in the example above because there were no conflicts. If there had been potential conflicts in three of the alternative locations, then a weight of seven would have been assigned to the location used to place the text. This newly computed weight would be used in the computation of an average weight for that location. That is,

$$avg(n+1) = \frac{(avg(n) * n) + wt(n+1)}{n+1}$$

where avg is the current average weight for a given location, n is the total number of weights that have been used to calculate the average weight, and wt(n+1) is the newly computed weight of the feature being processed. The more a well features the neural network can examine, the more accurate its weights will become. The location that ends up with the highest weight will be considered the optimum placement location for oil well symbols.

## EXTENSION OF NEURAL NETWORKS FOR CARTOGRAPHY

The above example illustrates how the traditional implementation of neural networks could be used to determine the optimum location for label placement for point symbols. The discussion below suggests an enhancement to the traditional implementation to speed up the learning process and to improve on the calculations of the weights.

It could be that for ninety percent of the cases where the text was placed in the cartographer's optimum location, there were at least two conflicts in other locations. Although those conflicts did not affect the choice of placement location, they will affect the calculations. This situation could result in the average weight for the cartographer's optimum placement location being equal to approximately eight. It is possible that for ninety percent of the cases where the text was placed in the cartographer's second-best location, there were conflicts in only one other location (which would have to have been the optimum location). This situation could result in the average weight for the cartographer's second-best placement location being equal to approximately nine, which is higher than the weight that was assigned to the cartographer's optimum placement position. Ideally, this situation would correct itself as more and more maps are examined. To avoid the above problem, the learning process can be helped by assigning a "locked-in" factor to each of the weights.

Initially, none of the weights would be considered locked in. Consider the first example where a feature exists with no conflict; it is known for certain that the placement position is the optimum position. When the weight that is to be assigned to a placement position is known for certain, that position can be marked as locked in. Once the optimum placement location has been locked in, it is possible to lock in the second-best location, then the third-best location. The second-best location could be locked in when text is placed in that location and there is only one other location containing text. Since that one other location has been locked in as the best location, it can be inferred that this location is the second best location. The process would continue for remaining locations.

The user should be able to tell the neural network what to do if text is encountered in a location not represented by the ten given locations. For example, the neural network could be told to flag any such text so that a new placement location can be added to the list of existing positions, or it could be told to ignore the text. The neural network should also be knowledgeable enough to recognize some of the special cases. For example, when placing text associated with a point feature that is located on a land mass and near a water body, some cartographers prefer to place the text so that it is located entirely within the land mass or within the water body.

Text locations for area features and linear features can be "learned" in a fashion similar to the way text locations for point features can be learned. But so far, the example has been oversimplified. Referring back to the sample rule, it can be seen that text location is only one aspect of the rule. Other important information used by the sample rule is the population, the city_name, and the feature_code that was used as an index into a table of parameters. Creating a neural network capable of generating a rule similar to the sample rule for each type of feature would be difficult, impractical, and in some cases impossible. However, if a mapping agency likes the appearance of a particular set of maps and wants to learn more about the rules used to generate these maps, then the use of neural network technology will be useful. Methodology to find all point features, to divide them according to feature code, and then to determine the point placement priorities used when placing the text for each type of feature are possible enhancements. This information could be used simply to generate a report or the software could use the information to generate a shell of a ruleset that would later be completed by the cartographer. The software could also examine the different symbologies used to represent the point features and use that information to generate a shell of a parameter file that could later be completed by the cartographer. Information about area features and linear features could be processed similarly to the way point features are processed.

## CONCLUSION

Manual placement of cartographic names is a time consuming process. Early research into automating this process was constrained by the software and hardware capabilities of that time. The average user now has access to more memory, more disk storage, and faster computers. Software technology has also continued to evolve. Research in the area of artificial intelligence has resulted in the development of many useful applications that are based on this technology. The technology of expert systems can be used to automate the text placement process by using a system that is flexible enough to meet the needs of different users and that can grow with the user. The technology of neural networks can be used to complement the expert system by assisting in the creation of the product rules that are used by the expert system. The use of a "locked-in" factor associated with the weights in the neural network can be used to speed up the learning process. This factor is also useful when trying to determine how accurate the weights are. The larger the percentage of weights that are locked in, the more accurate the weights are likely to be.

## REFERENCES


Basoglu, U., "A New Approach to Automated Name Placement Systems," Ph.D. Dissertation, University of Wisconsin at Madison, Madison, Wisconsin, May 1984.

Freeman, H. and J. Ahn, "Autonap - an Expert System for Automatic Map Name Placement", Proceedings International Symposium on Spatial Data Handling, Zurich, Switzerland, 1984.

Klimasauskas, Casimir C., "Neural Networks: A Short Course From Theory to Application", PC AI, Vol. 2, No. 4, November/December 1988, pp. 26-30.

Montanari, Ugo, "Continuous Skeletons from Digitized Images", Journal of the ACM, Vol. 16, No. 4, October 1969, pp. 534-549.

Yoeli, P., "The Logic of Automated Map Lettering", The Cartographic Journal, Vol. 9, No. 2, 1972.