PROCESS DESIGN AND CONTROL

Feasibility of Using Neural Networks and Genetic Algorithms To Predict and Optimize Coated Paper and Board Brightness

Amit Kumar and Vincent C. Hand*

Department of Paper Science and Engineering, Miami University, Gaskill Hall, Oxford, Ohio 45056

The final brightness of coated paper depends on a large number of interacting variables, including top- and bottom-coat formulation and coater settings. This paper reports the use and adjustments of neural networks to predict the brightness of a double-coated paper product based on a historical database. The trained neural network was integrated with a genetic algorithm to achieve a desired brightness at minimum cost. The neural network predicted brightness to within 3% of the true value. The optimum solution proposed by the genetic algorithm was 14% cheaper than the best solution in the database and appeared technically feasible and consistent with conventional understanding of the coating process. The accuracy of both the neural network and genetic algorithm was substantially improved by factorial experiments to maximize performance. For this complex optimization, the capabilities of the genetic algorithm appeared to be limited by computer hardware and software resources. To our knowledge, this represents the first use of an integrated neural network and genetic algorithm in the paper industry.

Introduction

Many interdependent variables affect the paper coating process, from coating formulation to machine settings. Consequently, no complete mathematical models of the coating process have been generated from first principles. Optimization of coating models is even more difficult.

Neural Networks (NNs). Neural networks (NNs) can assimilate operating data from industrial processes and "learn" the complex relationships within the process. They have been used in many applications in the paper industry and elsewhere. Beaverstock and Wolchina[1] were among the first to apply NNs in the paper industry, modeling a brownstock washer. NN models of Kappa number,[2] wood-chip refining,[3] and boiler emissions[4] illustrate the range of applications. Novak and colleagues[5] briefly reported on a NN model of the paper coating process.

The operation of NNs has been described many times before, for example, by Wasserman[6] or by Lawrence.[7] A NN is composed of a number of interconnected processing elements or "neurons". The strengths of the individual interconnections between neurons are adjustable and are often referred to as "weights". The NN program develops a model during training, from repetitive exposures to data and readjustment of the weights. In the widely used back-propagation NN, the network takes a set of inputs and produces a predicted output, which it then compares to the actual output. An error signal, the difference between the actual output and the predicted output, is propagated back through the network. The error signal alters the weights of all interconnections, so that subsequent predictions are closer to the actual value. Eventually the network constructs a consistent internal representation of the data or a set of weights for all interconnections. Then the trained network can predict outputs from sets of inputs previously unknown to it.

Genetic Algorithms (GAs). GAs have found application in optimization for a variety of engineering processes.[8,9] Their basic operation has been described in many books,[10,11] and is summarized here. GAs are inspired by the biological concept of natural selection and evolution. Just as the most fit organisms are most likely to survive and reproduce, solutions that are closest to an optimum are likely to form the basis for even better solutions. Individual organisms, which are solutions to the problem of survival, evolve through genetic recombination and mutation. In a GA, an individual "gene" encodes all of the inputs corresponding to one possible solution to an optimization problem. The most fit solutions are those closest to the optimum. The optimization process follows these steps:

1. Randomly create a population (predetermined number) of possible solutions.

2. Determine the fitness value of each of the proposed solutions.

3. Create an intermediate population by extracting solutions with a relatively higher fitness value.

4. Recombine and reproduce higher fitness solutions. Two solutions are divided at the same randomly chosen point, and genetic information from one side of the division is exchanged, in a process called "crossover". This generates two new solutions, or offspring, that replace genes from the previous generation. If only genes with a lower fitness are replaced, this is an "elitist" strategy.

* To whom correspondence should be addressed. Phone: 1-513-529-5811. Fax: 1-513-529-5814. E-mail: handvc@muohio.edu.

10.1021/ie000346i CCC: $19.00
Published on Web 11/04/2000

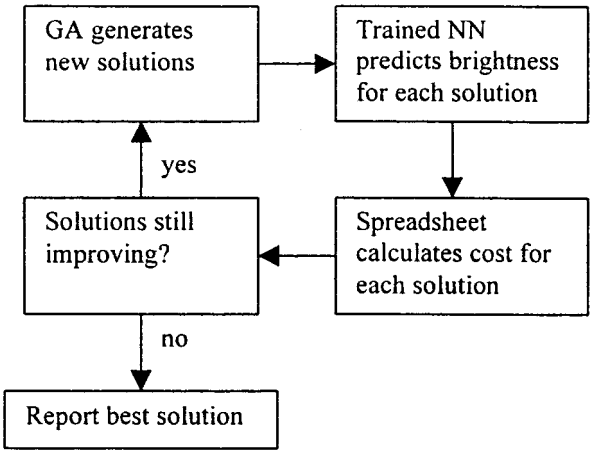

Figure 1. Flowchart of integration of NN and GA to optimize sheet brightness and cost. Brightness is the fitness value.

5. Mutation modifies a small fraction of the total number of solutions by randomly changing the values of one input variable. This reduces the possibility of the GA getting stuck in a local optimum.

6. Repeat steps 2−5 until the fitness value stops increasing.

(Note that steps 4 and 5 have created a new population consisting of the best solutions from the previous generation, along with some new solutions generated from crossover and mutation of these parent solutions.)

Integrating NNs and GAs. GAs have been used as one method for training NNs,[12] but the combination of the two methods for prediction and optimization seems relatively rare. Although a well-trained NN is good at predicting the output corresponding to a set of inputs, it cannot be efficiently used for optimization. A NN can accurately predict the result of any combination of inputs, but it cannot determine what optimum combination of inputs is required for a desired output. Optimizing using a NN can be done by applying "what if" questions repeatedly to the trained network. However, if a large number of variables are involved, this technique must be automated. GAs can be successfully used for optimization, but they require a way to evaluate the fitness of various solutions. A GA generates possible combinations of the input variables, each combination being a possible solution to the problem.

Figure 1 illustrates the integration of a NN and a GA, where the NN is trained to predict the brightness of a coated paper product from operational and formulation parameters, and both brightness and cost are optimized. The NN model predicts the brightness for the inputs comprising each solution generated by the GA. The GA uses the prediction from the NN to evaluate the fitness of each solution generated. The process is repeated, generating combinations of input variables and calculating the fitness of the combinations, until an optimal solution is achieved. Sette and co-workers[13] applied a similar approach to optimize the fiber/yarn production process. The approach has also been used in optimizing groundwater remediation.[14]

The primary objective of this paper was to demonstrate the integration of a NN and a GA to achieve a target brightness for coated paper at a minimum cost, utilizing data from a large historical database of coating trials. This was intended to investigate the feasibility of applying this approach to solve other optimization problems associated with paper coating and other manufacturing processes. A secondary objective was to evaluate simple factorial design techniques to maximize the performance of NNs and GAs used in the work. To our knowledge, this is the first reported use of GAs in the paper industry.

Methods

Data Source. We used a database for the pilot-scale coating trials performed at Mead Corp. Central Research. The database included the conditions and results for more than 3000 trials over a period of approximately 10 years. Each trial record had up to 60 fields. For this research, the brightness of a double-coated paper product with an air knife top coat was modeled and optimized. Records that did not include brightness information, had more or less than two coats, or did not use an air knife top coat were eliminated from further consideration.

A total of 43 input variables (Table 1) were used for the NN modeling to predict top-coat brightness. Each brand for the ingredient used in the top-coat formulation was treated as a separate input variable.

Data Conditioning. Some records in the database had missing values for an input variable. Missing values were replaced by a mean value for that input variable. Brand names for some of the coating formulation ingredients were combined, if the number of trials for these brands were relatively few and the brands were similar. This reduced the total number of input variables and improved training of NNs. The database was then examined for obvious outliers, and corresponding records were removed. In the end, there were 556 usable records that included data on brightness for a double-coated paper product with an air knife top coat. Out of these 556 records, 391 records, the training set, were used to train the network. Twenty percent, 110 records, the test set, were used to test the performance of the network during training. The remaining 55 records, the validation set, were kept aside and were used to test the NN prediction efficiency of the fully trained net. Stratified random sampling was used to ensure that various kinds of trials were represented in all three sets.

Table 1. List of Input Variables Used in Modeling[a]

coating formulation (top coat)	coating formulation (bottom coat)	other parameters
amount and brand of high gloss clay (2)	amount of structured clay (1)	calendering (1)
amount and brand of calcined clay (2)	amount of calcined clay (1)	times brushed (1)
amount and brand of no. 2 clay (3)	amount of calcium carbonate (1)	base sheet brightness (1)
amount and brand of titanium dioxide (3)	amount of no. 2 clay (1)	
amount and brand of latex binders (9)	amount of latex binders (2)	
amount and brand of other binders (5)	amount of other binders (2)	
amount and brand of plastic pigments (4)	amount of plastic pigments (1)	
coat weight (1)	coat weight (1)	
percent solids (1)		

[a] Numbers in parentheses indicate the total number of variables for that category.

The brightness ranged from about 76 %GE brightness to a little over 86%, with most values between 80 and 86%.

Software. NeuroShell2 and GeneHunter from Ward Systems Group, Frederick, MD, were used for building the NN model and GA optimization, respectively. These applications allow integration of NN and GA. GeneHunter operates as an add-in to Microsoft Excel. A routine, callable from an Excel spreadsheet, can be produced for the NN model, similar to a formula in a spreadsheet cell. The GA generates different solutions in the spreadsheet and keeps track of their fitness, calculated by the NN routine. The trained NN model can immediately calculate the brightness corresponding to any solution generated by the GA. The spreadsheet can be used to calculate other parameters, such as the formulation cost. This information can also be used in a fitness function.

In general, the input variables were mapped to the GA as 16-bit floating-point numbers. The program also allowed the designation of chromosomes that could only take on integer values. Because brands of coating components were either present or absent, brand chromosomes were constrained to integer values of 0 or 1.

The software documentation does not explicitly state the equations used for fitness functions or how multiple fitness functions are combined. The user chooses whether a specific spreadsheet cell or cells is to be minimized, maximized, or brought to a specific value. Some information about detailed implementation was derived by inference and from preliminary experimentation. Apparently the final fitness function is a simple linear combination of the component fitness functions, with each component given equal weight. All fitness functions are mapped linearly onto the range 0−1.

The GA software recognizes two types of constraints. "Hard" constraints limit the search space, and they are an absolute limit on values a chromosome can assume. "Soft" constraints reduce the fitness of solutions by an unspecified penalty function. For each soft constraint, the GA program requires an equation or inequality describing the constraint, a tolerance value, and a priority of "high", "medium", or "low". The tolerance value indicates how closely the constraint must be met, and the priority ranks the order of importance of meeting the criteria. Program documentation is not specific on how the tolerance and priority are implemented.

The software generated initial solutions for the GA randomly.

Maximizing NN Performance. The performance of a NN can be greatly improved by finding suitable values for its three major design parameters: number of neurons in the hidden layer, learning rate, and momentum. These values were determined using two consecutive 3 × 3 factorial designs: a screening factorial design and refinement factorial design. The screening factorial design determined a range for these parameters in which the NN performed the best. The refinement factorial design was applied over this range to determine the most suitable values of the adjustable design parameters.

The hidden layer of a NN is between the input and output layers. The number of neurons in the hidden layer influences the performance of the network. A NN with too many hidden neurons may generalize poorly. A NN with too few hidden neurons may never converge. Decreasing the number of neurons also decreases the computer time required to train the network. Typically, setting the number of hidden neurons equal to half the number of total input and output neurons is a suitable starting point for most applications.[7]

During training, the network keeps changing the weights of the interconnections; the rate of change is a function of the original weights and a factor. This factor is the product of the learning rate set by the user and the error in prediction, which the network computes from the difference between the predicted output and the actual output. For a large number of inputs, a higher learning rate may lead to oscillation, and the learning may never end. Training with a low learning rate may never generate a NN model with acceptable prediction accuracy. The learning rate value ranges from 0 to 1.

Momentum provides a smoothing effect, by making the change in the weights a function of the previous weight change. This parameter helps in faster learning and protects against oscillation. The momentum value ranges from 0 to 1.

Maximizing GA Performance. As with NNs, the performance of GAs can be improved by optimizing the value of three design parameters: population size, crossover rate, and mutation rate. The population size is the number of solutions considered in each cycle. A larger population size treats the problem more exhaustively but requires more computer time, because more solutions are considered in each generation. A GA with a large population is less likely to converge in a local optimum, because of wider coverage of the search space. Computer resources limit the population size. In our case the upper limit was 2000.

The crossover rate is the probability that a crossover will be applied to any particular solution. Its value can range from 0 to 1.

The mutation rate is the probability that mutation will be applied to any particular solution. Its value can range from 0 to 1.

A 3 × 2 factorial design was used to determine the most suitable values for crossover and mutation rates. Based on preliminary studies, the population was kept at 2000, the maximum allowed by the software. The optimization process was stopped when 200 000 solutions were tried without an increase in the fitness. An elitist strategy was used throughout.

Prices. The prices for all ingredients of the coating formulation were included in the optimization. List prices provided by the suppliers were used to calculate the cost of the formulation proposed by the GA. The same prices were used to calculate the cost of any formulation in the database for the purpose of comparison. The cost of any formulation was calculated according to eq 1, which the GA attempted to minimize.

$$\text{total cost of the formulation} = \sum\{(\text{parts}_i/100 \text{ parts total}) \times \text{price}_i\ [\text{cents/lb}] \times \text{coat weight}\ [\text{lb/thousand ft}^2] \times 1000\ [\text{thousand ft}^2/\text{million ft}^2]\} \quad (1)$$

where parts_i is the number of parts of any coating component i and price_i is the price of component i. To minimize round-off errors during addition, the cost was calculated for 1 million ft^2 of the coated surface.

The relative cost of the formulation suggested by the GA is reported, where eq 2 gives the relative cost.

relative cost =
[cost of the formulation suggested by GA × 100]/
[cost of the cheapest formulation in the database with equal or higher brightness] (2)

Constraints on the Optimization. In this work hard constraints were applied to ensure that the GA only evaluated formulations that were within the range of NN training. The following were implemented as soft constraints.

1. The sum of all ingredients in any formulation equals 100 ± 0.3 parts.
2. The number of different top-coat pigments is ≤4. Zero tolerance was allowed on this constraint.
3. The binder level is between 13 ± 0.3 and 19 ± 0.3 parts per hundred parts total formulation. Note that this terminology is slightly different from typical practice in paper coating, where binder levels are often specified on the basis of 100 parts pigment.
4. Brands and amounts for a specific ingredient must be linked. Thus, the GA may choose only one brand for an ingredient, if and only if that particular ingredient has an amount greater than zero. For example, if the GA chooses calcined clay as one of the ingredients, then it has to choose one brand for calcined clay. If the GA does not choose calcined clay, it cannot choose any calcined clay brand. Zero tolerance was allowed on this constraint.
5. Only calendered papers may be brush polished. Zero tolerance was allowed on this constraint.
6. The number of brush polishing passes must be an integer between 0 and 4. Zero tolerance was allowed on this constraint.

Functions Optimized. In an effort to achieve target brightness at minimum cost, a linear combination of two functions was optimized in the GA. The brightness of the top coat was calculated by using the NN model. A GE brightness of 86% was targeted, because it was the highest brightness that occurred frequently in the database. The minimum cost of the top-coat formulation was targeted. This was calculated in the spreadsheet, using eq 1, from the known costs of ingredients and the amounts and brands specified by the GA solution.

Results

Maximizing NN Performance. The screening factorial design for NN design parameters was applied over a range of 0.1−0.5 for both the learning rate and momentum. The range for numbers of hidden neurons was 12−32. Figures 2−4 show the effects of the learning rate, momentum, and number of hidden neurons. The response was R^2, the coefficient of multiple determination for the NN predictions with the validation set.

The NN was most affected by changes in the learning rate, and a learning rate of 0.1 gave the highest values of R^2 for any number of hidden neurons (Figure 2) and any value of momentum (Figure 3). The interaction between the learning rate and the number of hidden neurons is evident in Figure 2. Increasing the learning rate caused a steeper decline in R^2 when the number of hidden neurons was increased. In fact, when both of these parameters were set to their highest values (32 hidden neurons and 0.5 learning rate), the network produced no useful predictions, no matter what the value of momentum. This is shown in Figure 2 as an R^2 of 0.

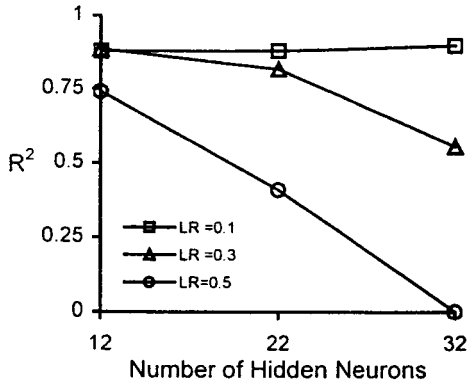

Figure 2. Effect of the number of hidden neurons on the prediction efficiency of a NN model at different values of the learning rate (LR). The prediction efficiency is the correlation coefficient between the actual and predicted values with the validation set.

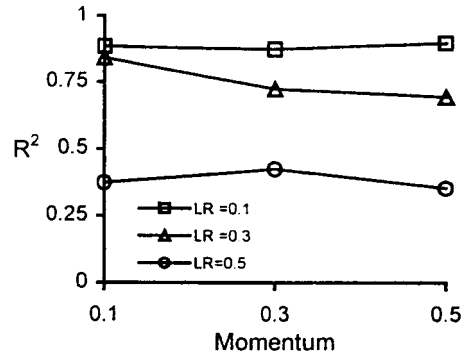

Figure 3. Effect of the momentum on the prediction efficiency of a NN model at different values of the learning rate (LR). The prediction efficiency is the correlation coefficient between the actual and predicted values with the validation set.

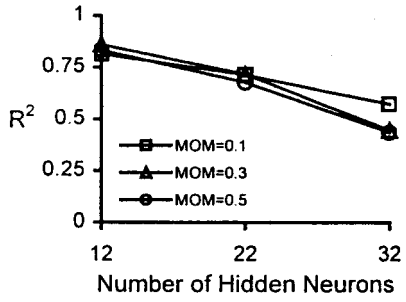

Figure 4. Effect of the number of hidden neurons on the prediction efficiency of a NN model at different values of the momentum (MOM). The prediction efficiency is the correlation coefficient between the actual and predicted values with the validation set.

Momentum had little effect on the R^2 (Figures 3 and 4). What fluctuations exist in these two figures are mostly due to random variations. Examination of three-dimensional response surface plots (not shown) bears this out.

The importance of a low learning rate is consistent with the fact that a more gradual change in the weights of NN is required to model a database with a complex response surface and many input variables. In addition, unnoticed outliers in a large historical database like ours will add noise and complexity. The gradual change in the weights is induced by a low learning rate, leading the NN to attain an acceptable prediction accuracy even when trained with a noisy database. As the NN goes over one record after another, several records might

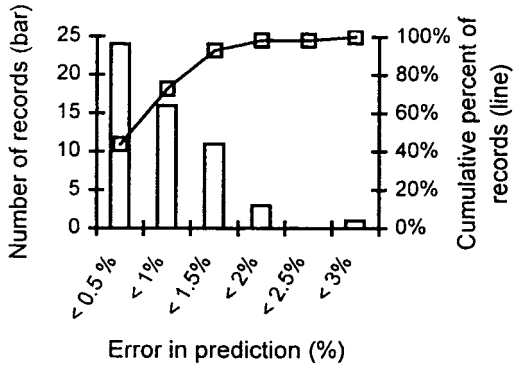

Figure 5. Distribution of the relative error in NN prediction for the 55 records in the validation set. The error is |predicted brightness − actual brightness|.

have different brightness values for a similar coating formulation and machine settings. If the learning rate is kept low, there will not be a drastic change in the weights of the NN when it encounters records with similar input values but different output values. Thus, keeping the learning rate low protects the NN from oscillating and helps it to converge to acceptable prediction accuracy.

The design of the refinement factorial design was based on the results of the screening factorial. The learning rate for the refinement factorial design centered on the lowest value used in the screening factorial design, or 0.1. Values of 0.05, 0.1, and 0.2 were chosen for the learning rate. Because low learning rates (0.05, 0.1, and 0.2) were chosen for the refinement factorial design, the number of hidden neurons was kept at 10, 20, and 30. This helped us to further explore the effect of the number of hidden neurons on the NN performance at low learning rates. There is a hint in the screening results that a high value of momentum improves results when the learning rate is low, especially for more hidden neurons. This possibility was explored further in the refinement design by using 0.4, 0.5, and 0.6 for values of momentum.

No definite effect of the learning rate, momentum, or number of hidden neurons was observed over the range used in the refinement factorial experiment. The R^2 values for the refinement factorial design ranged from 0.85 to 0.93. This implies that the range of the learning rate, momentum, and number of hidden neurons used in the refinement factorial design, derived from the screening factorial design, was well suited for our application. The best performing NN (20 hidden neurons, 0.1 learning rate, and 0.5 momentum) was retrained with the range of input parameters expanded by 10% to improve the ability to extrapolate. This also helped in optimization, because it allowed the search for an optimum to extend slightly beyond the ranges used in the historical trials.

The capability of the final NN to predict top-coat brightness was tested by using the trained network to predict the brightness for the 55 records in the validation set that were not used in any part of the training. Excellent performance of the NN model was achieved for the validation set. The R^2 value was 0.91, and the mean-squared error was 0.54 brightness units. All predictions were within 3% of the true value. A total of 24 of the 55 records in the validation data set (44%) were within 0.5% of the true value (Figure 5). There was no correlation between the magnitude of the prediction error and the actual brightness.

Normally a neural network took 4−5 min to train on a Pentium II 233 MHz computer with 64 MB of memory. However, if both the learning rate and momentum were kept very high (0.9 each), the network oscillated for 2 h without ever converging to an acceptable predicting accuracy.

Maximizing GA Performance. The trained NN was integrated with a GA to determine values for all of the input variables required to achieve target brightness at minimum formulation cost. In preliminary trials the GA was set to maximize the brightness and minimize the cost, but this led to difficulty in choosing the single best solution. Because the brightness is not related to the cost of the top-coat formulation by any simple function, the brightness and cost could not be compared to each other. This also led to a difficulty when trying to improve the efficiency of the GA by varying crossover and mutation rates. Each of the solutions generated differed in both brightness and cost. It was not possible to decide whether a solution that gave high brightness at high cost (e.g., 83% brightness at 94% relative cost) or one that gave lower brightness but at lower cost (e.g., 80% brightness at 90% relative cost) was better. These issues are well-known in multiobjective optimization.[9] Sette and co-workers[13] encountered a similar problem during their attempt to maximize tensile and elongation of fiber yarn simultaneously using NN and GA. Software limited us to exploring only the simplest solutions to this problem.

To overcome this problem in subsequent trials, a fixed brightness target was chosen and cost was minimized. Solutions could then be compared primarily on the basis of cost. A value of 86% GE brightness was targeted, because the database contained a number of trials that attained 86% GE brightness but very few that attained higher brightness.

The population was kept at 2000, the maximum allowed by the software, because it was observed during the preliminary work that the larger the population, the better the solution proposed by the GA. With a larger population, more solutions were considered in each cycle and more solutions were subjected to mutation, reducing the chances of a local optimum. A larger population provided for a more exhaustive and detailed search of the solution space. In addition, constraints were sometimes not met during optimization using smaller populations. For example, the sum of all of the ingredients in the final solution might be just 95 instead of 100 parts. On the other hand, smaller populations ran more quickly. An optimization with a population of 2000 took around 30 h on a Pentium II 233 MHz computer, as compared to 5−10 min for a population of 50.

The GA with a mutation rate of 0.15 and a crossover rate of 1.0 came out to be most suitable for our application. Solutions proposed by GA with all other combinations of mutation and crossover rates were poorer in one respect or another, with no definite trend, as listed in Table 2.

Three out of nine combinations tried came up with solutions having reduced cost, but none quite met the target of 86% GE brightness. The best solution (Table 2, trial 9) approached target brightness at 14% less cost. The population and crossover rates were already at their maximum values in this trial. Increasing the mutation rate did not improve the results.

The results of the optimization, or the final solution proposed by the GA, are listed in Tables 3 and 4. Note

Table 2. Brightness and Relative Cost for Solutions Proposed by GA with Different Crossover and Mutation Rates

trial no.	crossover rate	mutation rate	relative cost (%)	brightness achieved (% GE brightness)
1	0.85	0.15	164	84
2	0.9	0.1	118	82
3	0.9	0.15	120	75
4	0.95	0.05	82	82.5
5	0.95	0.1	73	80
6	0.95	0.15	118	81.8
7	1.0	0.05	121	84
8[a]	1.0	0.1	178	82
9	1.0	0.15	86	85

[a] Constraints were not satisfied in trial 8.

Table 3. Values of Operational Variables in the Optimum Solution Proposed by GA

parameter	value	units
base sheet brightness	15	% GE brightness
bottom-coat weight	6.8	g/m^2
top-coat weight	16	g/m^2
calendered	yes	
brush polished	1	pass
top-coat brightness	85	% GE brightness
relative cost	86	%

Table 4. Values of Formulation Variables in the Optimum Solution Proposed by GA[a]

formulation	top coat (parts per 100)	bottom coat (parts per 100)
amount of titanium dioxide	27	–
amount of no. 2 clay	25	18
amount of calcined clay	2	4
amount of high-gloss clay	23	–
amount of structured clay	–	40
amount of calcium carbonate	–	18
amount of plastic pigment	8	0
amount of protein binder	9	1
amount of latex	5	19
% solids	46	–

[a] Entries with a dash were not included in the optimization.

that operational and formulation variables are separated into two tables for clarity of presentation, but they are interdependent and were optimized simultaneously. The GA optimization also proposed specific brands for each component, but those are not reported here because of proprietary and other commercial concerns.

Discussion

The NN successfully modeled the brightness of a complex coated paper product to the extent that 100% of the predictions were within 3% of the actual value. The capability of the NN model is further emphasized by the fact that these predictions were made on a set of data not used in the training phase. We intentionally did not try to reduce the number of variables to achieve accurate predictions. We included all of the available variables that current knowledge of the coating process suggested could be relevant. The NN learned for itself what variables are most important in predicting brightness accurately. An examination of the sensitivity of the NN to input variables[15] was consistent with current knowledge. That is, factors such as the coat weight and amounts of high-brightness materials such as titanium dioxide had the greatest effect on the NN predictions. It is certainly possible that equal prediction accuracy could have been achieved with fewer input variables. Our approach reduced the time needed to achieve a satisfactory model as compared to a careful search for the most relevant input variables to provide to the NN. Of course, more selectivity would be required if fewer records were available for training.

The solution proposed by the integrated NN−GA model (Tables 3 and 4) is in line with our understanding of the coating process. The solution kept the expensive, high-brightness titanium dioxide content at a moderate 27% and included a low amount of binder (14%) to minimize the brightness reduction. To reduce cost while maintaining the brightness, the formulation incorporated high percentages of number 2 clay (25%) and high gloss clay (23%) in the top coat. The percent solids would be feasible for air knife coating. The solution proposed by the GA should be feasible, because all of the values are within the ranges used in the trials and recorded in the database. The cost of the formulation suggested by the GA is 86% of the cost of the cheapest formulation in the database that has identical or higher brightness. Because the pilot coating equipment was not available to us, we could not test the solution experimentally.

While 86% GE brightness was targeted, the GA was only able to achieve 85% brightness cost effectively. The final solution was obtained with the maximum values of two out of three GA design parameters: The crossover rate was kept at 1.0 and the population size at 2000, the maximum allowed by the software. One reason for not achieving target brightness might be that the large number of variables and constraints exceeded the capability of the available computing resources. Improved GA software with the ability to handle even larger population sizes and a faster computer capable of running the software might be able to find a solution that achieves target brightness at reduced cost. On the other hand, as suggested by one of the reviewers, the problem may arise from the simple linear combination of brightness and cost required by the software. Just giving more weight to brightness than cost might have allowed the GA to investigate high-brightness regions of the response surface more thoroughly.

Initially, to limit the number of variables, the bottom-coat brightness was chosen to represent the effect of the bottom-coat properties on the top-coat brightness. This approach highlighted a pitfall in such a complex optimization. The NN trained successfully, achieving a high R^2 value. When this NN was combined with the GA to achieve target top-coat brightness at minimum cost, the GA suggested solutions with very high bottom-coat brightness. There was no cost associated with bottom-coat brightness and high bottom-coat brightness enhanced top-coat brightness. As a result, solutions proposed by the GA appeared to achieve target brightness at a very low relative cost. In actual manufacturing conditions, it would be economically infeasible to achieve high top-coat brightness by increasing the bottom-coat brightness. Careful tradeoffs are required between simplifying the problem to increase speed and maintaining a model that represents reality.

To remedy this problem, the bottom-coat formulation was included in subsequent modeling and optimization. The final solution suggested high amounts of calcium carbonate and structured clay with no titanium dioxide in the bottom-coat formulation. This formulation would give sufficient coverage and brightness to the base sheet while maintaining low cost.

This work apparently is the first use of integrated NN

Table 5. Advantages and Disadvantages of NN and GA

NN	GA
Advantages	
no mathematical model required	no explicit model needed
utility with large database	no first guess required
combined numeric and nonnumeric data	wide search of response surface
self-teaching and checking	
Disadvantages	
uncertain how solution was achieved	uncertain how solution was achieved
risk of including too many variables in modeling	randomly generated first guesses

and GA for optimization of paper manufacturing. The work illustrates advantages and disadvantages of the integrated technique, the use of a systematic method for finding the best parameters for the NN and GA, and a method for extracting knowledge from a large process database.

The quality of the predictions from a NN depends on parameters such as the number of hidden neurons, learning rate, and momentum. While a number of authors have reported the use of NNs in the paper industry and elsewhere, few, if any, have discussed the selection of these parameters. We have shown that simple experiments in factorial design can substantially improve the performance of NN and GA. This work is the first step in "data mining" in a large historical database. Advantages and disadvantages of NN and GA are summarized in Table 5. The integrated techniques complement one another and provide the capability of modeling, predicting, and optimizing without requiring exhaustive knowledge of the process. Numerical and nonnumerical data can be combined readily into both NN and GA.

The NNs are self-teaching, require few assumptions, and can be readily checked. The self-teaching aspect of the NN leads to the most prominent disadvantage of this method, namely, the impossibility of determining exactly how the NN arrives at a model.

GAs share several disadvantages with many optimization techniques. The final optimum reported often depends on the starting point of the optimization process, and it cannot be proven that a global optimum has been achieved. In most traditional optimization techniques, the operator chooses a starting point based on prior experience. GA, in contrast, generates a random starting point. This disadvantage is compensated by the fact that many starting points are generated simultaneously. In fact, GA considers multiple solutions (2000) simultaneously at each iteration. Thus, a wide area of the search space is covered simultaneously. This fact, along with the effects of crossover and mutation, reduces the chance that GA will end at a local optimum. GAs also share a disadvantage with NNs, namely, that it is not clear how a GA arrives at an optimal solution.

This work was accomplished by combining commercially available software that did not require a programming language and yet allowed integration of NN and GA. The simplicity of implementation was offset by the inflexibility in choosing the strategy for multiple objective optimization or having full control over the operation of constraints. On balance this commercial software has proved useful for solving a practical problem using a historical database.

This work represents only a beginning and illustrates the feasibility of combining NN and GA. Of course, it is unrealistic to optimize for brightness alone, ignoring requirements for surface strength, gluability, and other properties that may require trade-off with the brightness. These issues are the subject of ongoing work on NN−GA modeling and optimization.

Acknowledgment

We are grateful to Mead Corp. Central Research for providing us the data used in this project and for initial funding. We are also grateful to Jennifer Gerber, Brian Casteel, and Susan E. Smith, Miami University students who spent many hours entering, checking, and organizing the original data. Finally, we appreciate the helpful comments of two anonymous reviewers. A preliminary version of this work was presented at the TAPPI99 conference in March 1999.

Literature Cited

(1) Beaverstock, M.; Wolchina, K. Neural Network helps G−P Ashdown Mill improve brownstock washer operation. *Pulp Paper* **1992**, *66* (9), 134−136.

(2) Dayal, B. S.; Macgregor, J. F.; Taylor, P. A.; Kildaw, R.; Marcikic, S. Application of Feed Forward Neural Networks and Partial Least Square Regression for Modeling Kappa Number in a Continuous Kamyr Digester. *Pulp Paper Can.* ***1994***, *95* (1), 26−32.

(3) Qian, Y.; Tessier, P.; Dumont, G. A. Modeling a wood-chip refiner using artificial neural networks. *Tappi J.* ***1995***, *78* (6), 167−174.

(4) Baines, G. H.; Hayes, R. L.; Stabell, J. L. Predicting boiler emissions with neural networks. *Tappi J.* **1997**, *80* (5), 57−61.

(5) Novak, G.; Gornik, M.; Malesic, I.; Goveker, E. Neural network modeling for the optimization of the properties of coated paper. *Das Papier* **1995**, *49* (5), 222−223.

(6) Wasserman, P. D. *Neural Computing, Theory and Practice*; Van Nostrand Reinhold: New York, 1989.

(7) Lawrence, J. *Introduction To Neural Networks: Theory, Design and Practice*; California Scientific Software Press: Nevada City, CA, 1994.

(8) Gen, M.; Cheng, R. *Genetic algorithms and engineering design*; Wiley: New York, 1997.

(9) Gen, M.; Cheng, R. *Genetic algorithms and engineering optimization*; Wiley: New York, 2000.

(10) Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*; Addison-Wesley: New York, 1989.

(11) Haupt, R. L.; Haupt, S. E. *Practical Genetic Algorithms*; Wiley: New York, 1998.

(12) Rooij, A. J. F. V.; Jain, L. C.; Johnson, R. P. *Neural Network Training Using Genetic Algorithms*; World Scientific: River Edge, NJ, 1996.

(13) Sette, S.; Boullart, L.; Langenhove, L. V.; Kiekens, P. Optimizing the fiber-to-yarn production process with a combined neural network/genetic algorithm approach. *Textile Res. J.* **1997**, *67* (2), 84−92.

(14) Rogers, L. L.; Dowla, F. U.; Johnson, V. M. Optimal field-scale groundwater remediation using neural networks and the genetic algorithm. *Environ. Sci. Technol.* ***1995***, *29* (5), 1145−55.

(15) Kumar, A. The Feasibility of Using Neural Networks and Genetic Algorithms to Predict and Optimize Coated Paper and Paper Board Brightness. M.S. Thesis, Miami University, Oxford, OH, 1999.

Received for review March 20, 2000
Accepted September 8, 2000

IE000346I