Fashion and Textiles

RESEARCH

Open Access

The use of extreme learning machines (ELM) algorithms to prediction strength for cotton ring spun yarn

Josphat Igadwa Mwasiagi*

*Correspondence:
igadwa@gmail.com
Moi University, Eldoret, Kenya

Abstract

The increasing use of artificial neural network in the prediction of yarn quality properties calls for constant improvement of the models. This research work reports the use of a novel training algorithm christened extreme learning machines (ELM) to prediction yarn tensile strength (strength). ELM was compared to the Backpropagation (BP) and a hybrid algorithm composed of differential evolution and ELM and named DE-ELM. The three yarn strength prediction models were trained up to a mean squared error (mse) of 0.001. This is an arbitrary level of mse that was selected to enable a comparative study of the performance of the three algorithms. According to the results obtained in this research work, the BP model needed more time for training, while the ELM model recorded the shortest training time. The DE-ELM model was in between the two models. The correlation coefficient (R^2) of the BP model was lower than that of ELM model. In comparison to the other two models the DE-ELM model gave the highest R^2 value.

Keywords: Cotton Yarn, Backpropagation (BP), extreme learning machines (ELM), Prediction, Strength

Introduction

Yarn quality properties can be predicted by modelling selected inputs and outputs of the cotton spinning system. This approach is widely applied in the study of the fiber to yarn process, where yarn properties can be predicted using mathematical, statistical and artificial neural network (ANN) models just to mention a few. A study of the comparison of ANN and other models (mathematical and statistical) has also been undertaken, and reports indicated that the ANN models have comparatively higher prediction efficiency (Guha et al. 2001; Ureyen and Gurkan 2008a, b; Majumdar and Majumdar 2004). The efficiency of the ANN algorithms has enabled the design of yarn quality prediction models, which can be used in the spinning industry (Furferi and Gelli 2010). The increasing use of the ANN models in the textiles industry warrants more attention to ensure that necessary improvements are made. This is expected to improve the cotton spinning process. It is with the afore-mentioned reasons that the objectives of studying the improvement of the ANN models used in the prediction of cotton yarn tensile strength (strength) were envisioned. This was done by comparing the working of Backpropagation (BP) models with the Extreme Learning Machines (ELM) algorithm during

SpringerOpen

© 2016 The Author(s). This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

the prediction of yarn tensile strength. Further improvement of the ELM algorithms was undertaken to produce more efficient prediction models made from a hybrid of differential evolution (DE) and ELM algorithms christened DE-ELM algorithm.

Methods

Yarn quality prediction models

The design of ANN models suitable for the prediction of yarn quality properties could take a variety of forms. As explained by Cybenko (1989) an ANN model with one hidden layer is robust enough and can be used to design yarn properties prediction models. The three layers of the ANN model can be designed using a multi-layer perception (MLP). In this research work a one hidden layer MLP was designed and used for the prediction of yarn tensile properties. The architecture of an MLP as explained by several researchers (Ham and Kostanic 2003; Huang et al. 2006a), with one hidden layer consists of several elements which include, input to hidden layer weights, hidden layer biases, hidden layer transfer function, hidden layer to output layer weights, output layer biases and output layer transfer function. The training of the MLP used to predict yarn properties in this research work was initially implemented using BP based algorithms, namely Levenberg Marquart. The BP based algorithms select the initial weights and biases using a random process which is likely to search for the weights and biases in the local area of the vector space hence leading to the problem of local minima. The selected weights and biases are normally updated using an iterative process. The iterative process could however slow down the working of the algorithms.

Attempts taken to improve the efficiency and speed of the BP algorithms were reported by Huang et al. (2006a, b) who suggested that by randomly selecting the input weights and hidden layer biases the efficiency and speed of the ANN model can be improved. This is due to the fact that the weights and biases of an MLP need not be iteratively updated. If the output layer function is eliminated then the logarithm becomes a linear system, and hence the hidden layer to output layer weights can be analytically determined. The aforementioned modifications introduced a new training algorithm christened ELM. ELM has been tested by Huang et al. (2006a) in several fields which include medical and forestry studies and proved to be faster and more efficient than the BP algorithm. Up to date there are no reports available in public domain for the study of the use of ELM in the cotton spinning process. This paper attempts to fill this void.

While ELM may be faster than BP algorithms there is still room for improvement. Given that ELM computes the output weights based on prefixed input weights and hidden layer biases, there is a possibility of a set of non-optimal or unnecessary input weights and hidden layer biases being selected. Furthermore the problem of local minima which is common in BP algorithms may also exist in ELM, albeit to a lower degree. As suggested by Zhu et al. (2005) the problems experienced while using ELM as a network training algorithm can be minimized by using the DE algorithm for the initial weights and biases selection process. This idea can be implemented by combining the DE and ELM algorithm to form a hybrid training algorithm. The hybrid algorithm will thereafter be referred to as DE-ELM for lack of a better name.

Designing and training of prediction models

Input factors and data pre-processing

The cotton fiber-to-yarn process involves processing cotton lint through a set of machines to produce yarn. The manufactured yarn is expected to meet some quality standards so that it can be produced at optimum productivity and perform within set standards in the subsequent processes. The quality of cotton yarn can be evaluated using yarn quality properties which include yarn elongation, strength, work of rupture, hairiness, unevenness etc. The aim of this research work was to study the prediction of yarn strength. While the selection of input factors was based on published works (Mwasiagi et al. 2008, 2012) (see Table 1), data pre-processing was also undertaken to ensure better performance of the models. The cotton lint and yarn samples were collected from Kenyan factories, and the cotton and yarn samples tested according to testing standards, in fiber and yarn laboratory. After collection and testing of the samples the data used in this research work was prepared. The collected data consisted of 144 samples each made up of 19 input factors. The date was subdivided into three sets: training, validation and testing sets in the ratio of 4:1:1, respectively. This was done in a random manner. The use of validation data will ensure that the BP network minimizes overfitting.

To further ensure a high quality of the input data, the 19 input factors were pre-processed using principal component analysis (PCA) as discussed by Chattopadhyay et al. (2001) and Bernstein et al. (1988). PCA was designed to come up with a new set of variables that have as little correlation with one another as possible. The level of correlation allowable can be determined based on the percentage confidence limit selected by the researcher. In this research work a 95 % confidence limit was selected which reduced the

Table 1 Characteristics of the input factors

No.	Input factors	Input value			PCA	
		Minimum	Maximum	Mean	Factor contribution (%)	Cumulative factor contribution (%)
1	Elongation (%)	3.98	8.84	6.13	10.15	10.2
2	Spindle speed (Rpm)	8000	12,000	10851	8.77	18.9
3	Ring diameter (mm)	42	50	43	8.7	27.6
4	Trash grade	1	4	2	8.53	36.2
5	SCI	108	187	157	7.87	44
6	Maturity	0.82	0.93	0.86	6.99	51
7	SFI (%)	5.9	9.7	7.39	6.96	58
8	Yellowness (+b)	8.9	12.3	10.69	6.81	64.8
9	Length (mm)	24.77	33.45	29.3	6.53	71.3
10	Micronaire	3.27	5.89	3.9	6.21	77.5
11	Length uniformity (%)	78.3	87.3	83.44	4.53	82.1
12	Twist (Tpi)	17.81	23.79	21.55	4.36	86.4
13	Strength (g/tex)	21.5	36.5	29.66	4.32	90.7
14	Count (tex)	19.41	31.56	23.59	2.86	93.6
15	Ringframe draft	20	30	25	2.17	95.8
16	Reflectance (Rd)	71	85	78	1.32	97.1
17	Trash area (%)	0.04	0.57	0.16	1.09	98.2
18	Trash count	1	36	12.36	0.93	99.1
19	Traveler weight (mg)	43	75	67	0.89	100

data set to 14 inputs. As given in Table 1, the five inputs removed from the initial data set were ring frame draft, traveller weight, fibre reflectance, fibre trash weight and fibre trash area.

Data pre-processing undertaken in this research work also included data normalization, which is a process of scaling the input factors in a data set so as to improve the accuracy of the subsequent numeric computations. One way to normalize the input factors is to subtract the mean of the input factor from each input factor unit and then divide the results with the standard deviation of the input factor (Demuth et al. 2005; MathWorks et al. 2004). Data normalization is necessary to ensure that the operation of the network is optimized. Long training times can be caused by the presence of an input vector whose length is much larger or smaller than the other input factors. By normalizing the data as described above all the values of the data will fall within a given range, and the impact of the input factors can be judged based on the pattern shown by the input factors but not on the numeric magnitude. Data normalization therefore has two main advantages: It reduces the scale of the network and ensures that input factors with large numeric values do not overshadow those with smaller numeric values. After the network has been trained, these vectors were transformed back to the original values by reversing the normalization process so that the outputs are presented as they were originally. This was done to ensure that the results can be interpreted with ease.

Designing and training of prediction models

The architecture of the BP strength prediction model was designed using Cybenko theorem (Cybenko 1989). Using the network design procedure reported by Mwasiagi et al. (2012), the final BP network had 14 inputs, one input layer, one hidden layer and one output (yarn strength). The ELM and DE-ELM were designed according to the reports of Huang et al. (2006a) and Zhu et al. (2005) respectively.

The BP yarn strength prediction models were trained using the Levenberg Marquart Backpropagation algorithm, which is one of the faster BP training algorithms used in training of prediction models (Hagan and Menhaj 1994; Demuth et al. 2005), until the set target error of 0.001 was attained. This is an arbitrary level that was selected for the purpose of comparing the three algorithms used in this research work. The performance of the strength prediction model, trained using the BP algorithm was monitored as the number of neurons was varied from 2 in steps of 1 until the set target error of 0.001 was attained.

To study the predictive power of the algorithm, three performance factors were considered. These were mean square error (mse), training time and correlation coefficient (R^2). These are the factors commonly reported by many researchers (Cheng and Adams 1995; Chattopadhyay et al. 2004; Desai et al. 2004; Majumdar and Majumdar 2004; Majumdar et al. 2005; Huang et al. 2006a; Ureyen and Gurkan 2008a, b; Mehment 2009). The ELM trained algorithms were trained in a similar manner like the LMBP algorithm, however the training algorithm used was ELM. Finally the yarn strength prediction model was trained using the DE-ELM algorithm. The performance of the DE-ELM yarn strength prediction models was monitored as the number of generations were varied from 1 to 10 in steps of 1. For every generation number the performance of the DE-ELM model was also monitored as the number of neurons were varied from 1 to 10.

Results and discussions

Prediction of yarn strength using BP algorithms

The yarn strength prediction model, trained using the BP algorithms, was used to predict yarn strength using the 14 inputs as discussed earlier. The performance of the strength prediction model, trained using the BP algorithm as the number of neurons was varied from 2 in steps of 1 until the set target error of 0.001 was attained, is given in Table 2. The strength model was able to attain the set target error when the number of neurons in the hidden layer reached 10.

The BP trained strength model exhibited a typical network behavior whereby the mse showed a steady improvement as the number of neurons in the hidden layer increased. The mse reached the set target (0.001) when the number of hidden neurons was 10. The R^2 value measured using the testing data was 0.917.

Prediction of yarn strength using ELM algorithm

The prediction of yarn strength using the ELM model was carried out in the same manner like the BP trained model, and the results are given in Table 3.

The performance of the ELM strength prediction model improved rapidly especially when the number of neurons was varied from 2 to 15 in steps of 1. Thereafter the change in the mse value was relatively smaller, with the set target error of 0.001 being attained when the number of neurons was 41.

The time needed for network training kept on increasing as the number of neurons was increased. When the number of neurons was 41, and the set target error (0.001) had

Table 2 Performance of strength model trained using BP algorithm

Neurons	2	3	4	5	6	7	8	9	10
mse	0.05200	0.02510	0.01450	0.00850	0.00648	0.00443	0.00240	0.00140	0.00083
Time(s)	1.281	1.287	1.291	1.297	1.312	1.344	1.375	1.39	1.438
Iteration	19	18	17	20	22	24	25	23	26

Table 3 Performance of ELM strength prediction model

No. of neurons	$(mse)_{tr}$	Time (s)	No. of neurons	$(mse)_{tr}$	Time (s)	No. of neurons	$(mse)_{tr}$	Time (s)
2	0.102144	0.0156	16	0.004507	0.0210	30	0.002025	0.0267
3	0.094864	0.0159	17	0.003640	0.0215	31	0.001849	0.0273
4	0.066444	0.0160	18	0.003088	0.0217	32	0.001806	0.0275
5	0.053161	0.0163	19	0.002770	0.0221	33	0.001764	0.0278
6	0.038704	0.0167	20	0.002601	0.0228	34	0.001722	0.0283
7	0.032556	0.0171	21	0.002450	0.0231	35	0.001681	0.0285
8	0.029447	0.0178	22	0.002401	0.0237	36	0.001560	0.0290
9	0.019386	0.0183	23	0.002352	0.0239	37	0.001444	0.0296
10	0.014851	0.0189	24	0.002304	0.0240	38	0.001225	0.0297
11	0.010053	0.0192	25	0.002256	0.0246	39	0.001089	0.0303
12	0.008711	0.0196	26	0.002209	0.0251	40	0.001018	0.0305
13	0.007299	0.0199	27	0.002162	0.0255	41	0.000949	0.0311
14	0.006724	0.0201	28	0.002116	0.0261			
15	0.005242	0.0208	29	0.002070	0.0263			

been attained the time needed for training was 0.0311 sec. This is much lower than the time of 1.438 needed by the BP trained algorithm. The much lower training time for the ELM models could be due to the saving of time that may be needed to iteratively update the weights and biases in the BP trained models. AS reported by Huang et al. (2006a), this is one of main advantages of the ELM trained models, when compared to the BP models.

The 41 neurons strength prediction model recorded an R^2 value of 0.988, when exposed to the testing data. This is an improvement of 7.7 % when compared to the BP trained model, and it could be an indication of better generalization and ability to avoid local minima trap.

The ELM yarn strength prediction therefore recorded faster training time and better correlation coefficient (R^2) but much higher neurons in the hidden layer when compared to the BP trained yarn strength model.

Prediction of yarn strength using DE-ELM algorithm

Having established that the ELM model needed 41 neurons in the hidden layer to attain the set target error of 0.001, DE-ELM was used to train the yarn strength prediction model with an aim of reducing the number of neurons. In the DE-ELM hybrid training algorithm the initial selection of the weights and biases was done using the differential evolution algorithm which is a global search algorithm, and thereafter they were updated analytically using the ELM algorithm. Using the DE algorithm the number of generations (G) was increased from 1 to 10 in steps of 1 and the number of neurons was varied from 2 to 10 in steps of 1. The results of the above-mentioned experiments are given in Table 4.

The results of using DE-ELM to train the yarn prediction model as given in Table 4 indicated that the set target mse value of 0.001 could be attained by all the neurons (2–10) experimented. The only difference could be seen in the number of generations needed. The 2 neurons algorithm need more generations (5), the 10 neurons model needed 3 generations while all the other neurons were in between. It is therefore clear that while the ELM model needed 41 neurons in the hidden layer to attain the set target mse value (0.001) the DE-ELM strength prediction model could attain the set target error with even 2 neurons in the hidden layer and 5 generations of the DE-ELM

Table 4 Variations of $(\mathrm{mse})_{tr}$ for DE-ELM strength model

G	Number of neurons in the hidden layer								
	10	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**
1	0.003158	0.003481	0.003881	0.004720	0.007073	0.010100	0.018036	0.028866	0.034820
2	0.001552	0.001459	0.001936	0.001632	0.001521	0.005170	0.008855	0.010201	0.020192
3	0.000365	0.000410	0.000404	0.000640	0.000713	0.004330	0.004679	0.006529	0.010878
4	8.1×10^{-5}	5.3×10^{-5}	0.00025	0.00031	0.00015	0.000262	0.000515	0.003856	0.007500
5	5.8×10^{-6}	6.1×10^{-7}	6.6×10^{-5}	1.2×10^{-5}	1.4×10^{-6}	1.9×10^{-5}	2.4×10^{-5}	0.000625	0.000339
6	4.4×10^{-11}	4×10^{-9}	4×10^{-8}	6.9×10^{-7}	3.2×10^{-8}	6.8×10^{-8}	1.6×10^{-6}	2.9×10^{-5}	2.8×10^{-5}
7	3.1×10^{-14}	4.5×10^{-11}	1.2×10^{-13}	1×10^{-11}	2×10^{-11}	4×10^{-10}	2.5×10^{-7}	2.3×10^{-6}	1.4×10^{-6}
8	3.9×10^{-18}	1.7×10^{-16}	2.2×10^{-20}	4.8×10^{-13}	1.7×10^{-14}	8.8×10^{-11}	3.1×10^{-9}	4×10^{-7}	1.6×10^{-8}
9	7.8×10^{-26}	7.8×10^{-22}	4.9×10^{-21}	9.6×10^{-17}	2.6×10^{-16}	7.8×10^{-14}	1.7×10^{-12}	1.6×10^{-9}	2.6×10^{-9}
10	1.2×10^{-30}	4.4×10^{-26}	2.8×10^{-23}	4.8×10^{-20}	4.6×10^{-17}	1.9×10^{-15}	1.6×10^{-13}	1.0×10^{-12}	2.8×10^{-10}

algorithms. This is much lower than the 10 neurons needed by the BP trained models. The use of the DE for the initial selection of the weights and biases seem to be able to drastically improve the performance of the ELM model.

The general trend of the DE-ELM model was such that the mse value improved as the number of neurons and generations were increased. In Table 5, a boundary between the models which did not attain the set target error (0.001) and those which attained the set target is shown. It is worth noting that the R^2 values for the models are all higher (0.990 and above) than the R^2 value depicted by the BP trained models (R^2 value of 0.917) and the ELM trained models (R^2 value of 0.988).

The ability of the DE-ELM algorithm to reach the set mse target (0.001) while using fewer number of neurons in the hidden could be attributed to that fact that it was able to optimize the selection of the initial weights and biases. The better performance of the DE-ELM model needed more time when compared to the ELM model. However the time taken by the DE-ELM model is much lower when compared to the time needed by BP model.

In summary it is clear that the DE-ELM model inherited the advantages of the ELM models discussed earlier on. The only disadvantage is the higher training time, which could be due to the fact that the DE algorithm needs time to first select the weights and biases.

Comparison of BP, ELM and DE-ELM strength prediction models

Three types of models have been designed and trained to predict yarn strength in this research work. The performance of the yarn strength prediction models can be compared by using Table 6.

The ELM model needed many neurons in the hidden layer (41) to reach the set target error of 0.001. This is one of the disadvantages of the ELM algorithm as reported by Zhu et al. (2005). The ELM algorithm needed less time to train, when compared to the other models. The DE-ELM model gives very good performance with a reduced number of neurons and a higher R^2 value. Its training speed is slower than that of the ELM model but still much faster than that of the BP model.

Table 5 Performance of strength prediction model with DE-ELM algorithm

Neuron	10	9	8	7	6	5	4	3	2
Generation	3	3	3	3	3	4	4	5	5
$(mse)_{tr}$	0.00037	0.00041	0.00046	0.00064	0.000713	0.000262	0.000515	0.00063	0.000339
R-value	0.995	0.995	0.994	0.994	0.992	0.991	0.991	0.990	0.992
Time(s)	0.8681	0.8594	0.8394	0.7944	0.7825	0.7644	0.75	0.7263	0.7188

Table 6 Comparison of strength prediction models

Model	Input factors	No. of neurons	No. of generations	$(mse)_{tr}$	R-value	Iteration	Time(s)
LM	14	10	N/A	0.00083	0.917	26	1.438
ELM	14	41	N/A	0.000949	0.988	N/A	0.0311
DE-ELM	14	2	5	0.000339	0.992	N/A	0.7188

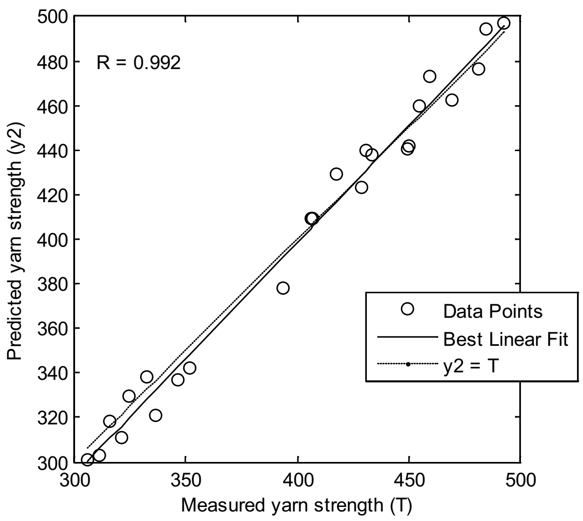

Fig. 1 Prediction of yarn strength using DE-ELM algorithm

The predicted and measured values for the 2 neuron model with 5 generation are given in Fig. 1. The predicted strength values traced the measured values so closely such that the success rate was at 99.2 %. This implies that the error rate is <1 %. This could be a sign of very good network generalization.

Conclusions

Yarn strength prediction models using BP, ELM and DE-ELM models were designed and trained up to a mse of 0.001. The performance of the BP algorithms was compared to two non-BP algorithms namely ELM and DE-ELM during the prediction of yarn strength.

The BP trained model needed 10 neurons in the hidden layer and was the slowest among the three algorithms. The ELM models exhibited the shortest training time (0.0311 s) but needed 41 neurons in the hidden layer. The DE-ELM hybrid models needed 2 neurons in the hidden layer and 5 generations, and its training time (0.7188 s) was shorter than the BP model but much slower than the ELM model. The BP yarn strength prediction model needed 1.438 s to attain the set mse target of 0.001. The hybrid model (DE-ELM) gave the highest prediction efficiency (R^2 of 0.992), while the BP and ELM models recorded R^2 values of 0.917 and 0.988 respectively.

Competing interests
The author declares that he has no competing interests.

Received: 10 May 2016 Accepted: 25 August 2016
Published online: 28 November 2016

References
Bernstein, I. R., Garbin, C. P., & Teng, G. K. (1988). *Applied multivariate analysis* (pp. 157–197). Berlin: Springer.
Chattopadhyay, R., Guha, A., & Jayadeva, G. (2004). Performance of neural networks for predicting yarn properties using principal component analysis. *Journal of Applied Polymer Science, 91*, 1746–1751.
Cheng, L., & Adams, D. L. (1995). Yarn strength prediction using neural networks: part I: fiber properties and yarn strength relationship. *Textile Research Journal, 65*(9), 495–500.
Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematical Control Signal System, 1989*(2), 303–314.

Demuth, H. B., Beale, M., & Hagan, M. T. (2005). Neural network toolbox. *The Math Works, 5*, 25.

Desai, J. V., Kane, C. D., & Bandyopadhayay, B. (2004). Neural Networks: An alternative solution for statistically based parameter prediction. *Textile Research Journal, 74*(3), 227–230.

Furferi, R., & Gelli, M. (2010). Yarn strength prediction: A practical model based on artificial neural networks. *Advances in Mechanical Engineering, 8*, 1–10.

Guha, A., Chattopadhyay, R., & Jayadeva, B. (2001). A comparison of mechanistic statistical, and neural network models. *Journal of Textile Institute, 92*(1), 139–145.

Hagan, M. T., & Menhaj, M. B. (1994). Training feedforward networks with the Marquardt algorithm. *IEEE Transactions on Neural Networks, 5*(6), 989–993.

Ham, F. M., & Kostanic, I. (2003). *Principles of neurocomputing for science and engineering* (pp. 24–135). Beijing: China Machine Press.

Huang, G. B., Chen, L., & Siew, C. K. (2006a). Universal approximation using incremental constructive feedforward networks with random hidden nodes. *IEEE Transactions on Neural Networks, 17*(4), 879–892.

Huang, G. B., Zhu, Q. Y., & Siew, C. K. (2006b). Extreme learning machine: Theory and applications. *Neurocomputing, 70*, 489–501.

Majumdar, P. K., & Majumdar, A. (2004). Predicting the breaking elongation of ring spun yarns using mathematical statistical and artificial neural models. *Textile Research Journal, 74*(7), 652–655.

Majumdar, A., Majumdar, P. K., & Sarkar, B. (2005). Application of an adaptive neuro-fuzzy system for the prediction of cotton yarn strength from HVI fibre properties. *Journal of Textile Institute, 96*(1), 55–60.

MathWorks. (2004). *Matlab -The Language of technical computing*. Natick, MA: The MathWorks Inc.

Mehment, D. (2009). Prediction of yarn properties using evaluation programing. *Textile Research Journal, 79*(11), 963–972.

Mwasiagi, J. I., Huang, X. B., & XinHou, W. (2008). Performance of neural network algorithms during the prediction of yarn elongation. *Fibers and Polymers Korea, 9*(1), 80–86.

Mwasiagi, J. I., Huang, X. B., & XinHou, W. (2012). The use of hybrid algorithms to improve the performance of yarn parameters prediction models. *Fibers and Polymers, 13*(9), 1201–1208.

Ureyen, M. E., & Gurkan, P. (2008a). Comparison of artificial neural network and linear regression models for prediction of ring spun yarn properties. i: prediction of yarn tensile properties. *Fibers and Polymers, 9*(1), 87–91.

Ureyen, M. E., & Gurkan, P. (2008b). Comparison of artificial neural network and linear regression models for prediction of ring spun yarn properties ii: prediction of yarn hairiness and unevenness. *Fibers and Polymers, 9*(1), 92–96.

Zhu, Q. Y., Qin, A. K., Suganthan, P. N., & Huang, G. B. (2005). evolutionary extreme learning, machine. *Pattern Recognition, 38*, 1759–1763.

Submit your manuscript to a SpringerOpen journal and benefit from:

- ► Convenient online submission
- ► Rigorous peer review
- ► Immediate publication on acceptance
- ► Open access: articles freely available online
- ► High visibility within the field
- ► Retaining the copyright to your article

Submit your next manuscript at ► springeropen.com