Evolutionary Ordinal Extreme Learning Machine*

Javier Sánchez-Monedero,
Pedro Antonio Gutiérrez, and Cesar Hervás-Martínez

University of Córdoba, Dept. of Computer Science and Numerical Analysis
Rabanales Campus, Albert Einstein building, 14071 - Córdoba, Spain
{jsanchezm,pagutierrez,chervas}@uco.es

Abstract. Recently the ordinal extreme learning machine (ELMOR) algorithm has been proposed to adapt the extreme learning machine (ELM) algorithm to ordinal regression problems (problems where there is an order arrangement between categories). In addition, the ELM standard model has the drawback of needing many hidden layer nodes in order to achieve suitable performance. For this reason, several alternatives have been proposed, such as the evolutionary extreme learning machine (EELM). In this article we present an evolutionary ELMOR that improves the performance of ELMOR and EELM for ordinal regression. The model is integrated in the differential evolution algorithm of EELM, and it is extended to allow the use of a continuous weighted RMSE fitness function which is proposed to guide the optimization process. This favors classifiers which predict labels as close as possible (in the ordinal scale) to the real one. The experiments include eight datasets, five methods and three specific performance metrics. The results show the performance improvement of this type of neural networks for specific metrics which consider both the magnitude of errors and class imbalance.

Keywords: ordinal classification, ordinal regression, extreme learning machine, differential evolution, class imbalance.

1 Introduction

Ordinal regression, or ordinal classification, problems are classification problems where the problem nature suggests the presence of an order between labels. In addition, it is expected that this order would be reflected on the data distribution through the input space [1]. Compared to nominal classification, ordinal classification has not attracted much attention, nevertheless the number of algorithms and associated publications have grown in the late years [2].

In this work we propose an evolutionary extreme learning machine for ordinal regression. We modify the ELMOR model proposed by Deng et. al [3] with an

* This work has been partially subsidized by the TIN2011-22794 project of the Spanish Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P11-TIC-7508 project of the "Junta de Andalucía" (Spain).

J.-S. Pan et al. (Eds.): HAIS 2013, LNAI 8073, pp. 500–509, 2013.

extension to allow a probabilistic formulation of the neural network, for which we propose a fitness function that considers restrictions related to ordinal regression problems. We evaluate the proposal with eight datasets, five related methods and three specific performance metrics.

The rest of the paper is organized as follows. Section 2 introduces the ordinal regression problem and formulation. Section 3 presents the extreme learning machine and its evolutionary alternative, and Section 4 explains the proposed method. Experiments are covered at Section 5 and finally conclusions and future work are summarized in the last section.

2 Ordinal Regression

Ordinal regression is a type of supervised classification problem in which there is an order within categories [1,4]. This order is generally deduced from the problem nature by an expert or by simple assumptions about the data.

2.1 Problem Formulation

The ordinal regression problem can be mathematically formulated as a problem of learning a mapping ϕ from an input space $\mathbb{X}$ to a finite set $\mathcal{C} = \{C_1, C_2, \ldots, C_Q\}$ containing Q labels, where the label set has an order relation $C_1 \prec C_2 \prec \ldots \prec C_Q$ imposed on it (symbol $\prec$ denotes the ordering between different categories). The rank of an ordinal label can be defined as $\mathcal{O}(\mathcal{C}_q) = q$. Each pattern is represented by a K-dimensional feature vector $\mathbf{x} \in \mathbb{X} \subseteq \mathbb{R}^K$ and a class label $t \in \mathcal{C}$. The training dataset $\mathbf{D}$ is composed of N patterns $\mathbf{D} = \{(\mathbf{x}_i, t_i) \mid \mathbf{x}_i \in \mathbb{X},\ t_i \in \mathcal{C}, i = 1, \ldots, N\}$, with $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{iK})$.

For instance, bond rating can be considered as an ordinal regression problem where the purpose is to assign the right ordered category to bonds, being the category labels $\{C_1 = \text{AAA}, C_2 = \text{AA}, C_3 = \text{A}, C_4 = \text{BBB}, C_5 = \text{BB}\}$, where labels represent the bond quality assigned by credit rating agencies. Here there is a natural order between classes $\{\text{AAA} \prec \text{AA} \prec \text{A} \prec \text{BBB} \prec \text{BB}\}$, AAA being the highest quality one and BB the worst one.

Considering the previous definitions, an ordinal classifier (and the associated training algorithm) has two challenges. First, since the nature of the problem implies that the class order is somehow related to the distribution of patterns in the space of attributes $\mathbb{X}$ as well as the topological distribution of the classes, the classifier must exploit this a priori knowledge about the input space [1,4]. Secondly, specific performance metrics are needed. Given the bond rating example, it is reasonable to conclude that predicting class BB when the real class is AA represents a more severe error than that associated with AAA prediction. Therefore, performance metrics must consider the order of the classes so that misclassifications between adjacent classes should be considered less important than the ones between non-adjacent classes, more separated in the class order [5,4].

2.2 Performance Metrics

As mentioned, ordinal regression needs specific performance metrics. In this work we will use the accuracy and the Mean Absolute Error (MAE), since those are the most used ones, and the recently proposed average MAE, which is a robust metric for imbalanced datasets. Let us suppose we want to evaluate the performance of N predicted ordinal labels for a given dataset $\{\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_N\}$, with respect to the actual targets $\{t_1, t_2, \ldots, t_N\}$. The accuracy, also known as Correct Classification Rate or Mean Zero-One Error (MZE) when expressed as an error, is the rate of correctly classified patterns.

However, the MZE does not reflect the magnitude of the prediction errors. For this reason, the MAE is commonly used together with MZE in the ordinal regression literature [2,5,6]. MAE is the average absolute deviation of the predicted labels from the true labels:

$$MAE = \frac{1}{N} \sum_{i=1}^{N} e(\mathbf{x}_i), \tag{1}$$

where $e(\mathbf{x}_i) = |\mathcal{O}(t_i) - \mathcal{O}(\hat{t}_i)|$. The MAE values range from 0 to $Q-1$. However, neither MZE, nor MAE are suitable for problems with imbalanced classes. To solve this issue, Baccianella et. al [7] proposed to use the average of the MAE across classes:

$$AMAE = \frac{1}{Q} \sum_{j=1}^{Q} MAE_j = \frac{1}{Q} \sum_{j=1}^{Q} \frac{1}{n_j} \sum_{i=1}^{n_j} e(\mathbf{x}_i), \tag{2}$$

where $AMAE$ values range from 0 to $Q-1$ and n_j is the number of patterns in class j.

3 Extreme Learning Machine

This section presents the ELM and ELMOR models, in order to establish the baseline for the article proposal.

3.1 ELM for Nominal Classification and Regression

This section presents the extreme learning machine (ELM) algorithm and the Evolutionary ELM. For a further review of ELM please refer to specific survey [8]. The ELM algorithm has been proposed in [9]. ELM and its extensions have been applied to several domains including multimedia Quality-of-Service (QoS) [10] or sales forecasting, among others.

The ELM model is a Single-Layer Feedforward Neural Network that is described as follows. Let us define a classification problem with a training set given by N samples $\mathbf{D} = \{(\mathbf{x}_i, \mathbf{y}_i) : \mathbf{x}_i \in \mathbb{R}^K, \mathbf{y}_i \in \mathbb{R}^Q, i = 1, 2, \ldots, N\}$, where $\mathbf{x}_i$ is

a $K \times 1$ input vector and $\mathbf{y}_i$ is a $Q \times 1$ target vector[1] Here, a target $\mathbf{y}$, associated to pattern $\mathbf{x}$, is defined so that $y_j = 1$ means that pattern $\mathbf{x}$ belong to class j and $y_k = 0 | j \neq k$ means the pattern does not belong to class k, this is generally known as a 1-of-Q coding scheme. Let us consider a multi-layer perceptron (MLP) with M nodes in the hidden layer and Q nodes in the output layer given by:

$$f(\mathbf{x}, \theta) = (f_1(\mathbf{x}, \boldsymbol{\theta}_1), f_2(\mathbf{x}, \boldsymbol{\theta}_2), \ldots, f_Q(\mathbf{x}, \boldsymbol{\theta}_Q)), \tag{3}$$

where:

$$f_q(\mathbf{x}, \boldsymbol{\theta}_q) = \beta_0^q + \sum_{j=1}^{M} \beta_j^q \sigma_j(\mathbf{x}, \mathbf{w}_j), q = 1, 2, \ldots, Q, \tag{4}$$

where $\theta = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_Q)^T$ is the transpose matrix containing all the neural net weights, $\boldsymbol{\theta}_q = (\boldsymbol{\beta}^q, \mathbf{w}_1, \ldots, \mathbf{w}_M)$ is the vector of weights of the q output node, $\boldsymbol{\beta}^q = \beta_0^q, \beta_1^q, \ldots, \beta_M^q$ is the vector of weights of the connections between the hidden layer and the qth output node, $\mathbf{w}_j = (w_{1j}, \ldots, w_{Kj})$ is the vector of weights of the connections between the input layer and the jth hidden node, Q is the number of classes in the problem, M is the number of sigmoidal units in the hidden layer and $\sigma_j(\mathbf{x}, \mathbf{w}_j)$ the sigmoidal function:

$$\sigma_j(\mathbf{x}, \mathbf{w}_j) = \frac{1}{1 + \exp\left(-\left(w_{0j} + \sum_{i=1}^{K} w_{ij} x_i\right)\right)}, \tag{5}$$

where w_{0j} is the bias of the jth hidden node.

The linear system $f(\mathbf{x}_j) = \mathbf{y}_j, j = 1, 2, \ldots, N$, can be written as the following matrix system $\mathbf{H}\boldsymbol{\beta} = \mathbf{Y}$, where $\mathbf{H}$ is the hidden layer output matrix of the network:

$$H(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{w}_1, \ldots, \mathbf{w}_M) = \begin{bmatrix} \sigma(\mathbf{x}_1, \mathbf{w}_1) & \cdots & \sigma(\mathbf{x}_1, \mathbf{w}_M) \\ \vdots & \ddots & \vdots \\ \sigma(\mathbf{x}_N, \mathbf{w}_1) & \cdots & \sigma(\mathbf{x}_N, \mathbf{w}_M) \end{bmatrix}_{N \times M},$$

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \vdots \\ \boldsymbol{\beta}_M \end{bmatrix}_{M \times Q} \text{ and } \mathbf{Y} = \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_N \end{bmatrix}_{N \times Q}.$$

The ELM algorithm randomly selects the $\mathbf{w}_j = (w_{1j}, \ldots, w_{Kj}), j = 1, \ldots, M$, weights and biases for hidden nodes, and analytically determines the output weights $\beta_0^q, \beta_1^q, \ldots, \beta_M^q$ for $q = 1 \ldots Q$ by finding the least square solution to the given linear system. The minimum norm least-square solution (LS) to the linear system is $\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{Y}$, where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse of matrix $\mathbf{H}$. The minimum norm LS solution is unique and has the smallest norm among all the LS solutions, which guarantees better generalization performance.

[1] Note we change the notation of the targets here from a scalar target (t) to a vector target ($\mathbf{y}$). This is due to the multi-class neural network outputs, since neural networks generally have Q or $Q - 1$ number of output neurons.

The evolutionary extreme learning machine (EELM) [11] improves the original ELM by using the original Differential Evolution (DE) algorithm proposed by Storn and Price [12]. The EELM uses DE to select the input weights $\mathbf{w}_j$, and the Moore-Penrose generalized inverse to analytically determine the output weights between hidden and output layers. Then, the population of the evolutionary algorithm is the set of input weights $\mathbf{w}_j$ which are evaluated completing the ELM training process.

3.2 ELM for Ordinal Regression

The ELM has been adapted to ordinal regression by Deng et. al [3] being the key of their approach the output coding strategies that impose the class ordering restriction. That work evaluates single multi-class and multi-model binary classifiers. The single ELM was found to obtain slightly better generalization results for benchmark datasets and also to report the lowest computational time for training. In the present work the single ELM alternative will be used. In the single ELMOR approach the output coding is a targets binary decomposition [13], an example of five classes ($Q = 5$) decomposition is shown in Table 1.

Table 1. Example of nominal and ordinal output coding for five classes ($Q = 5$)

1-of-Q coding	Frank and Hall coding [13]
$\begin{pmatrix} +1 & -1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 & -1 \\ -1 & -1 & +1 & -1 & -1 \\ -1 & -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & -1 & +1 \end{pmatrix}$	$\begin{pmatrix} +1, & -1, & -1, & -1, & -1 \\ +1, & +1, & -1, & -1, & -1 \\ +1, & +1, & +1, & -1, & -1 \\ +1, & +1, & +1, & +1, & -1 \\ +1, & +1, & +1, & +1, & +1 \end{pmatrix}$

In this way, the solutions provided by the $\hat{\boldsymbol{\beta}} = \mathbf{H}^{\dagger}\mathbf{Y}$ expression tend to produce order aware models. For the generalization phase, the loss-based decoding approach [14] is applied, i.e. the chosen label is that which minimizes the exponential loss:

$$\hat{t} = \arg \min_{1 \leq q \leq Q} d_{\mathrm{L}} \left(\mathbf{M}_q, \mathbf{g}(\mathbf{x}) \right),$$

where $\hat{t}$ is the predicted class label, being $\hat{t} \in \mathcal{C} = \{C_1, C_2, \ldots, C_Q\}$ containing Q labels, $\mathbf{M}_q$ is the code associated to class C_q (i.e. each of the rows of coding at the right of Table 1), $\mathbf{g}(\mathbf{x}) = f\left(\mathbf{x}, \boldsymbol{\theta}\right)$ is the vector of predictions given by the model in Eq. (3), and $d_{\mathrm{L}}\left(\mathbf{M}_q, \mathbf{g}(\mathbf{x})\right)$ is the exponential loss function:

$$d_{\mathrm{L}} \left(\mathbf{M}_q, \mathbf{g}(\mathbf{x}) \right) = \sum_{i=1}^{Q} \exp \left(-\mathbf{M}_{iq} \cdot g_i(\mathbf{x}) \right). \tag{6}$$

4 Evolutionary Extreme Learning Machine for Ordinal Regression

This section presents our evolutionary extreme learning machine for ordinal regression (EELMOR) model and the associated training algorithm. First, the EELMOR extends the ELMOR model to obtain a probabilistic output. For doing that, the softmax transformation layer is added to the ELMOR model using the negative exponential losses of Eq. (6):

$$p_q = p_q(\mathbf{x}, \boldsymbol{\theta}_q) = \frac{\exp(-d_{\mathrm{L}}(\mathbf{M}_q, \mathbf{g}(\mathbf{x})))}{\sum_{i=1}^{Q} \exp(-d_{\mathrm{L}}(\mathbf{M}_i, \mathbf{g}(\mathbf{x})))}, \; 1 \leq q \leq Q, \tag{7}$$

where p_q is the posterior probability that a pattern $\mathbf{x}$ has of belonging to class C_q and this probability should be maximized for the actual class and minimized (or ideally be zero) for the rest of the classes. This formulation is used for evaluating the individuals in the evolutionary process but not for solving the ELMOR system of equations.

In the case of ordinal regression, the posterior probability must decrease from the true class to more distant classes. This has been pointed out in the work of Pinto da Costa et al. [5]. In that work an unimodal output function is imposed to the neural network model, and the probability function monotonically decreases as the classes are more distant from the true one.

According to the previous observation, we propose a fitness function for guiding the evolutionary optimization that simultaneously considers two features of a classifier:

1. Misclassification of non-adjacent classes should be more heavily penalized as the difference between classes labels grows.
2. The posterior probability should be unimodal and monotonically decrease for non-adjacent classes.

In this way, not only the right class output is considered, but also the posterior probabilities with respect to the wrong classes are reduced. In order to satisfy these restrictions, we propose the weighted root mean square error ($WRMSE$).

First, we design the type of cost associated with the errors. Let us define the absolute cost matrix as $\mathbf{A}$, where the element $a_{ij} = |i - j|$ is equal to the difference in the number of categories, $a_{ij} = |i - j|$. The absolute cost matrix is used, for instance, for calculating the MAE, being i the actual label and j the predicted label. An example of an absolute cost matrix for five classes is shown in Table 2. In the case of $WRMSE$, $\mathbf{A}$ cannot be directly applied because it would suppress information about the posterior probability of the correct class (see Eq. (9)). Then, we add a square matrix of ones $\mathbf{1}$ so that our final cost matrix is $\mathbf{C} = \mathbf{A} + \mathbf{1}$ (see an example in Table 2).

Second, according to the model output defined in Eq. (7), we define the weighted root mean square error ($WRMSE$) associated to a pattern as:

$$e = \frac{\sum_{q=1}^{Q} (c_{iq} \sqrt{(y_q - p_q)^2})}{Q}, \tag{8}$$

Table 2. Example of an absolute cost matrix ($\mathbf{A}$) and an absolute cost matrix plus the matrix of ones ($\mathbf{C} = \mathbf{A} + \mathbf{1}$) for five classes ($Q = 5$)

$\mathbf{A}$	$\mathbf{C} = \mathbf{A} + \mathbf{1}$
$\begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{pmatrix}$	$\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 2 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 2 & 3 \\ 4 & 3 & 2 & 1 & 2 \\ 5 & 4 & 3 & 2 & 1 \end{pmatrix}$

where i is index of the true target and c_{iq} represents the cost of errors associated to the q output of the neural network coded in matrix $\mathbf{C}$ (see Table 2). Finally, the total error of the prediction is defined as:

$$WRMSE = \frac{\sum_{i=1}^{N}(e_i)}{N}. \tag{9}$$

For ending this section, it should be noticed that in a single model multiclass classifier the $RMSE$ has the interesting property of selecting solutions that consider good classification performance of all classes simultaneously [15]. In the case of MZE, only one network output (the one with maximum value) contributes to the error function, and it does not contribute with the output's value. However, for $RMSE$ it is straightforward to check that each model output (posterior probabilities) contributes to the error function. Then, the model's decision thresholds and posteriors will tend to be more discriminative. This implicit pressure over the posteriors is even more severe in the case of $WRMSE$.

5 Experimental Section

This section presents experiments comparing the present approach with several alternatives, with special attention to the EELM and the ELMOR as reference methods.

5.1 Datasets and Related Methods

Table 3 shows the characteristics of the 8 datasets included in the experiments. The publicly available real ordinal regression datasets were extracted from benchmark repositories (UCI [16] and `mldata.org` [17]). The experimental design includes 30 stratified random splits (with 75% of patterns for training and the remainder for generalization).

In addition to the EELM, ELMOR and the proposed method (EELMOR), we include the following alternatives in the experimental section:

- The POM algorithm [18], with the *logit* link function.
- The GPOR method [6] including automatic relevance determination, as proposed by the authors.

Table 3. Characteristics of the benchmark datasets

Dataset	#Pat.	#Attr.	#Classes	Class distribution
automobile (AU)	205	71	6	(3, 22, 67, 54, 32, 27)
balance-scale (BS)	625	4	3	(288, 49, 288)
bondrate (BO)	57	37	5	(6, 33, 12, 5, 1)
contact-lenses (CL)	24	6	3	(15, 5, 4)
eucalyptus (EU)	736	91	5	(180, 107, 130, 214, 105)
LEV (LE)	1000	4	5	(93, 280, 403, 197, 27)
newthyroid (NT)	215	5	3	(30, 150, 35)
pasture (PA)	36	25	3	(12, 12, 12)

– NNOR [19] Neural Network with decomposition scheme by Frank and Hall in [13].

The algorithms' hyper-parameters were adjusted by a grid search using MAE as parameter selection criteria. For NNOR, the number of hidden neurons, M, was selected by considering the following values, $M \in \{5, 10, 20, 30, 40\}$. The sigmoidal activation function was considered for the hidden neurons. For ELMOR, EELM and EELMOR, higher numbers of hidden neurons are considered, $M \in \{5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, given that it relies on sufficiently informative random projections [9]. With regards to the GPOR algorithm, the hyperparameters are determined by part of the optimization process. For EELM and EELMOR the evolutionary parameters' values are the same as used at [11]. The number of iterations was 50 and the population size 40.

5.2 Experimental Results

Table 4 shows mean generalization performance of all the algorithms including metrics described at Section 2.2. The mean rankings of MZE, MAE and $AMAE$ are obtained to compare the different methods. A Friedman's non-parametric test for a significance level of $\alpha = 0.05$ has been carried out to determine the statistical significance of the differences in rank in each method. The test rejected the null-hypothesis stating that all algorithms performed equally in the mean ranking of the three metrics. Because of space restrictions, we will only examine $AMAE$ metric, since it is the most robust one. For this purpose, we have applied the Holm post-hoc test to compare EELMOR to all the other classifiers in order to justify our proposal. The Holm test is a multiple comparison procedure that works with a control algorithm (EELMOR) and compares it to the remaining methods [20]. Results of the test are shown in Table 5, which shows that our proposal improves on all the methods' performance except NNOR for $\alpha = 0.10$, and there are only statistical differences with EELM for $\alpha = 0.05$. The second best performance in $AMAE$ was for NNOR.

Table 4. Experimental generalization results comparing the proposed method to other nominal and ordinal classification methods. The mean and standard deviation of the results are reported for each dataset, as well as the mean ranking. The best result is in bold face and the second best result in italics.

	MZE Mean								Mean MZE rank
Method/DataSet	AU	BS	BO	CL	EU	LE	NT	PA	
EELM	0.453	0.152	0.544	0.344	0.507	0.393	0.152	0.389	4.94
ELMOR	0.384	0.082	*0.476*	0.383	0.440	**0.371**	0.051	0.389	3.31
GPOR	0.389	**0.034**	**0.422**	0.394	**0.315**	0.388	*0.034*	0.478	3.13
NNOR	*0.376*	*0.039*	0.500	**0.294**	0.418	0.373	0.035	**0.237**	**2.31**
POM	0.533	0.092	0.656	0.378	0.841	0.380	**0.028**	0.504	4.69
EELMOR	**0.360**	0.092	0.533	*0.306*	*0.394*	*0.372*	0.035	*0.333*	*2.63*
	MAE Mean								Mean MAE rank
Method/DataSet	AU	BS	BO	CL	EU	LE	NT	PA	
EELM	0.688	0.216	0.722	0.517	0.718	0.439	0.154	0.404	5.06
ELMOR	0.542	0.089	0.649	0.522	0.531	**0.406**	0.052	0.404	3.44
GPOR	0.594	**0.034**	**0.624**	0.511	**0.331**	0.422	*0.034*	0.489	2.75
NNOR	**0.503**	*0.044*	0.671	*0.456*	0.476	0.408	0.035	**0.241**	*2.44*
POM	0.953	0.111	0.947	0.533	2.029	0.415	**0.028**	0.585	5.00
EELMOR	*0.510*	0.108	*0.644*	**0.433**	*0.447*	*0.407*	0.035	*0.344*	**2.31**
	AMAE Mean								Mean AMAE rank
Method/DataSet	AU	BS	BO	CL	EU	LE	NT	PA	
EELM	0.813	0.426	1.119	0.545	0.778	0.632	0.212	0.404	4.75
ELMOR	0.649	0.176	1.168	0.531	0.575	0.611	0.114	0.404	3.94
GPOR	0.792	**0.051**	1.360	0.651	**0.362**	0.654	0.062	0.489	4.13
NNOR	**0.566**	*0.066*	1.135	*0.493*	0.506	**0.608**	0.059	**0.241**	*2.19*
POM	1.026	0.107	*1.103*	0.535	1.990	0.632	**0.050**	0.585	4.06
EELMOR	*0.592*	0.172	**1.041**	**0.463**	*0.489*	**0.608**	*0.052*	*0.344*	**1.94**

Table 5. Table with the different algorithms compared with EELMOR using the Holm procedure ($\alpha = 0.10$) in terms of $AMAE$. The horizontal line shows the division between methods significantly different from EELMOR.

i	Algorithm	z	p	α'_{Holm}
1	EELM	3.0067	0.0026	0.0200
2	GPOR	2.3385	0.0194	0.0250
3	POM	2.2717	0.0231	0.0333
4	ELMOR	2.1381	0.0325	0.0500
5	NNOR	0.2673	0.7893	0.1000

6 Conclusions and Future Work

In this work, we have adapted the ELMOR model to the Evolutionary ELM. We have proposed the weighed RMSE error function to guide the algorithm. Based on theoretical analysis and experimental results, we justify the proposal compared to the reference methods and other ordinal regression techniques.

Future work involves the design and experiments with new output codes and associated error functions. In addition, as a future work, a comparison can be performed taking into account the run time of the algorithms. Also the exploration of limitations of the proposal should be part of future research.

References

1. Hühn, J.C., Hüllermeier, E.: Is an ordinal class structure useful in classifier learning? Int. J. of Data Mining, Modelling and Management 1(1), 45–67 (2008)
2. Gutiérrez, P.A., Pérez-Ortiz, M., Fernández-Navarro, F., Sánchez-Monedero, J., Hervás-Martínez, C.: An Experimental Study of Different Ordinal Regression Methods and Measures. In: Corchado, E., Snášel, V., Abraham, A., Woźniak, M., Graña, M., Cho, S.-B. (eds.) HAIS 2012, Part II. LNCS, vol. 7209, pp. 296–307. Springer, Heidelberg (2012)
3. Deng, W.Y., Zheng, Q.H., Lian, S., Chen, L., Wang, X.: Ordinal extreme learning machine. Neurocomputing 74(1-3), 447–456 (2010)
4. Sánchez-Monedero, J., Gutiérrez, P.A., Tiňo, P., Hervás-Martínez, C.: Exploitation of Pairwise Class Distances for Ordinal Classification. Neural Computation 25(9), 2450–2485 (2013)
5. Pinto da Costa, J.F., Alonso, H., Cardoso, J.S.: The unimodal model for the classification of ordinal data. Neural Networks 21, 78–91 (2008)
6. Chu, W., Ghahramani, Z.: Gaussian processes for ordinal regression. Journal of Machine Learning Research 6, 1019–1041 (2005)
7. Baccianella, S., Esuli, A., Sebastiani, F.: Evaluation measures for ordinal regression. In: Proceedings of the Ninth International Conference on Intelligent Systems Design and Applications, ISDA 2009, San Mateo, CA, pp. 283–287 (2009)
8. Huang, G.B., Wang, D., Lan, Y.: Extreme learning machines: a survey. International Journal of Machine Learning and Cybernetics 2(2), 107–122 (2011)
9. Huang, G.B., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 42(2), 513–529 (2012)
10. Chen, L., Zhou, L., Pung, H.: Universal Approximation and QoS Violation Application of Extreme Learning Machine. Neural Processing Letters 28, 81–95 (2008)
11. Zhu, Q.Y., Qin, A., Suganthan, P., Huang, G.B.: Evolutionary extreme learning machine. Pattern Recognition 38(10), 1759–1763 (2005)
12. Storn, R., Price, K.: Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. Journal of Global Optimization 11(4), 341–359 (1997)
13. Frank, E., Hall, M.: A simple approach to ordinal classification. In: Flach, P.A., De Raedt, L. (eds.) ECML 2001. LNCS (LNAI), vol. 2167, pp. 145–156. Springer, Heidelberg (2001)
14. Allwein, E.L., Schapire, R.E., Singer, Y.: Reducing multiclass to binary: a unifying approach for margin classifiers. J. of Machine Learning Research 1, 113–141 (2001)
15. Sánchez-Monedero, J., Gutiérrez, P.A., Fernández-Navarro, F., Hervás-Martínez, C.: Weighting efficient accuracy and minimum sensitivity for evolving multi-class classifiers. Neural Processing Letters 34(2), 101–116 (2011)
16. Asuncion, A., Newman, D.: UCI machine learning repository (2007)
17. PASCAL: Pascal (Pattern Analysis, Statistical Modelling and Computational Learning) machine learning benchmarks repository (2011), `http://mldata.org/`
18. McCullagh, P., Nelder, J.A.: Generalized Linear Models, 2nd edn. Monographs on Statistics and Applied Probability. Chapman & Hall/CRC (1989)
19. Cheng, J., Wang, Z., Pollastri, G.: A neural network approach to ordinal regression. In: Proceedings of the IEEE International Joint Conference on Neural Networks, IJCNN 2008, pp. 1279–1284. IEEE Press (2008)
20. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. J. Mach. Learn. Res. 7, 1–30 (2006)