Molecular Crystals and Liquid Crystals

Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/gmcl20

Machine Learning Methods Used to Predict the Liquid-Crystalline Behavior of Some Copolyethers

Florin Leon [a] , Silvia Curteanu [b] , Cătălin Lisa [b] & Nicolae Hurduc [c]

[a] Department of Computer Science and Engineering, Gh. Asachi Technical University, Iaşi, Romania

[b] Department of Chemical Engineering, Gh. Asachi Technical University, Iaşi, Romania

[c] Department of Natural and Synthetic Polymers, Gh. Asachi Technical University, Iaşi, Romania

Version of record first published: 22 Sep 2010

To cite this article: Florin Leon, Silvia Curteanu, Cătălin Lisa & Nicolae Hurduc (2007): Machine Learning Methods Used to Predict the Liquid-Crystalline Behavior of Some Copolyethers, Molecular Crystals and Liquid Crystals, 469:1, 1-22

To link to this article: http://dx.doi.org/10.1080/15421400701431232

Mol. Cryst. Liq. Cryst., Vol. 469, pp. 1–22, 2007
Copyright © Taylor & Francis Group, LLC
ISSN: 1542-1406 print/1563-5287 online
DOI: 10.1080/15421400701431232

Machine Learning Methods Used to Predict the Liquid-Crystalline Behavior of Some Copolyethers

Florin Leon
Department of Computer Science and Engineering, Gh. Asachi Technical University, Iaşi, Romania

Silvia Curteanu
Cătălin Lisa
Department of Chemical Engineering, Gh. Asachi Technical University, Iaşi, Romania

Nicolae Hurduc
Department of Natural and Synthetic Polymers, Gh. Asachi Technical University, Iaşi, Romania

This article presents new methods of predicting the liquid-crystalline behavior of some copolyethers with mesogen groups in the main chain, using neural networks or different categorization algorithms. The prediction of properties is correlated with chemical structure, type and copolymer composition, molecular weight, and a series of structural characteristics estimated by mechanical molecular simulation. The importance of a suitable choice for the parameters that characterize the structure and the behavior of the copolyethers is emphasized, as well as the codification of the input and output variables of the algorithms. The procedures are simple and provide accurate results, especially neural-network-based techniques.

Keywords: Categorization algorithms; liquid-crystalline behavior; neural networks

INTRODUCTION

Because we living in an extremely complex environment, human beings are forced to reduce the number and diversity of stimuli in order to better control them. One of the strategies employed is *categorization*, that is, establishing classes that include a group of objects or

Address correspondence to Silvia Curteanu, Department of Chemical Engineering, Gh. Asachi Technical University, B-dul D. Mangeron No. 71A, 700050, Iaşi, Romania. E-mail: scurtean@ch.tuiasi.ro

stimuli that have some common physical or functional traits. For most practical uses, a supervised approach is necessary, especially when using a computer-based method to assist the human user. Automatic classification techniques developed by machine learning and statistics researchers achieve excellent performance, given sufficient structure in the underlying relationship between characteristics and categories and also sufficient data describing the relationship.

Inductive learning aims at formulating hypotheses, or general descriptions of concepts, from instances of these concepts. In inductive learning, the categories learned are considered to be described by a set of attributes, which can be either numeric or symbolic (nominal). The instances are depicted by different values for these domain attributes. Instances, especially if their attributes are numeric, can be represented as points in an n-dimensional space, where n is the number of attributes. Usually, to not bias a dimension against another, the values are normalized. The learning task is to find a general description for each category (the subset of attribute values that can uniquely identify that class). After concept descriptions have been established, it is important to properly classify previously unseen instances (to predict their membership to one of the existing classes).

One of the goals of *categorization algorithms* is to retain the nature of data distribution to have good classifications with low prediction errors. Depending on the underlying principle of the method, the following methods of categorization are found in literature: rule-based methods, prototype-based methods, and exemplar-based methods. They are implemented in different algorithms. Decision trees are represented by a tree-like structure, in which, based on successive tests from general to particular, a new instance is introduced into an existing class [1]. In nearest-neighbor algorithms, an instance is introduced in the same class as its nearest neighbor in feature space or as the majority's class of a group of neighbors [2]. Bayesian induction computes the membership probabilities of an instance to every class of the domains, selecting the class with the highest probability [3]. Bayesian networks are an extension of the naïve Bayes classifiers, because they take into account the fact that the events of the real world are not always independent [4]. Neural networks can also be used as classifiers [5].

Few approaches are known in literature about the application of the machine-learning models in chemistry. Different machine-learning algorithms, including hierarchical clustering, decision trees, k-nearest neighbors, support vector machines, and bagging were applied to construct models to predict the molecular weight of the polymers produced by a set of homogeneous catalysts [6]. In Ref. [7], a machine-learning approach for sequence-based prediction of protein–protein interaction sites is described. A support vector machine classifier was trained to predict whether a surface residue is an interface residue, based on the identity of the target residue and its ten sequence neighbors [7]. Different machine-learning methods are used in structure prediction; several examples are presented in Refs. [8–11].

The present approach is an opportunity to prove the utility and the efficiency of the categorization algorithms and neural networks for classification problems, particularly for quantifying the structure–properties relation for some copolyethers.

In the polymers' field, the efficient design of new materials requires the prediction of the properties of the candidate polymers and the selection of the best structure from all the potential possibilities. To solve this problem, a quantitative structure–property relationship is necessary and, as a function of the investigated property, some methods are given in the literature [12–14]. One of the most interesting properties of polymers is the liquid-crystalline (LC) behavior, because in this state, the materials combine two essential properties of the matter: the order and the mobility. However, because of the complexity of the liquid-crystalline phase, it is not at all easy to predict the occurrence of a mesophase. There are many methods of predicting the liquid-crystalline behavior, based on molecular, energetic, or structure–property relationship models [15–22].

In the present article, we report a new approach to predict the liquid-crystalline behavior for a series of copolyethers with mesogene groups in the main chain, using neural networks and categorization algorithms. The prediction of properties is correlated with the chemical structure, type, and copolymer composition (molar ratio), polymer molecular weight, and two geometric parameters: the length of the spacer and an asymmetry parameter. In a prediction problem, the substitution of the experiments means to save time and materials. It should be mentioned that the neural-network predictions cannot provide the exact result of the actual experiment. However, if the error rate is low both for the training and prediction/cross-validation phases, a neural network can provide acceptably accurate results, with the advantage of speed and without the use of any material resources.

CATEGORIZATION METHODS

Neural networks are typical methods of categorization. A neural network consists of processing units called *neurons* and information flow channels between the neurons—*interconnections*. The way in which neurons are connected to form a network represents the *neural network topology* (*architecture*). A *multilayer neural network* has *input, hidden*, and *output layers* consisting of input, hidden, and output neurons, respectively. The most common neural network architecture is the *multilayer feedforward neural network* (often called *multilayer perceptron*, *MLP*) (Fig. 1). The inputs of a hidden neuron are combined in a weighted sum, to which a constant term (*bias*) is added. This sum is then passed through a nonlinear transformation called *activation function*. With each connection, a *weight* is associated, which is a weighted factor that reflects its importance. This weight is a scalar value, which can be positive (excitatory) or negative (inhibitory). The artificial neural networks have a nonlinear nature; therefore, the small values at the exit of one node can influence the final result. During the training phase, the algorithm used must ensure that these small fluctuations are attenuated so that they do not influence the final result.

In the *training phase*, the neural network learns the behavior of the process. The *training data set* contains both input patterns and the corresponding output patterns (also called *target patterns*). Neural training leads to finding values of connection weights that minimize differences between the network outputs and the target values. The most extensively adopted algorithm for the learning phase is the *back-propagation algorithm*. The weights of the connections are usually initialized with small, nonzero values. They are small, so they do not saturate the sigmoid function output, and nonzero, because in the training phase of the back-propagation algorithm, these values are used for adjusting the weights.

The purpose of developing a neural model is to devise a network (set of formulae) that captures the essential relationships in the data. These formulae are then applied to new sets of inputs to produce corresponding outputs. This is called *generalization* and represents a subsequent phase after training—*validation* or *testing phase*. A

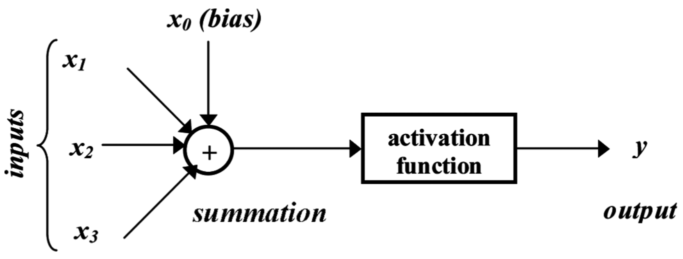

FIGURE 1 Unit of a multilayer perceptron.

network is said to generalize well when the input–output relationship, found by the network, is correct for input–output patterns of validation data that were never used in training the network (*unseen data*).

The multilayer perceptrons used nowadays have nonlinear, sigmoid activation functions, unlike the first models, which had a step function following the "all or nothing" perception rule. This gives them the advantage of a gradual propagation of signals (although bounded between limits such as [−1,1] or [0,1]) and also the possibility of using a derivative-based training algorithm such as back propagation.

However, despite their advantages, neural networks remain subsymbolic "black boxes" that cannot use a priori information about data and show difficulties in extracting structural knowledge. In this article, we attempt a comparison between several categorization methods, both subsymbolic and symbolic.

The algorithms used for solving the proposed problem are *neural networks* with different topologies, *decision trees* (the C4.5 algorithm, random tree, and random forest), the *Bayesian classifier*, which tries to estimate the probability distribution of data, and an instance-based algorithm, which can be viewed as an extension of the *classical nearest-neighbor approach*. Each method can be applied with several variants by changing the parameters involved in the computing procedure.

In the case of the simple inductive-learning structure of a decision tree, the general rule of membership of an instance to a class is given by traversing the tree branches from the root to the leaf corresponding to the instance or class. Each internal node of the tree represents a test of none or more properties, and the branches that go from that node are labeled with the possible results of the test. C4.5 is such a decision-tree induction algorithm that recursively partitions the data set and selects the test that leads to the highest information gain [23], that is, the resulting subclasses should be as homogenous as possible. To avoid overfitting, the resulting tree can be *pruned* at the end of the categorization process. In this way, the tree will be smaller, with more errors on the training set than the *unpruned version*, but supposedly with better generalization capabilities.

The random-tree (RT) [24] classifier builds a tree that considers k random features at each node and performs no pruning; therefore its error rate on the training set alone is rather small. A random forest (RF) [25] is composed of several classification trees. To classify a new object from an input vector, the input vector is run down each of the trees in the forest. Each tree gives a classification, and this is considered to be a "vote" for that class. The forest chooses the classification that has most of the votes (over all the trees in the forest).

Bayesian rule induction is based on Bayes's theorem about conditional probabilities and determines the posterior probability distribution of X (the conditional probability distribution of X given Y) by multiplying the prior probability density function by the likelihood function and then normalizing the result:

$$P(X|Y) = \frac{P(Y|X) \cdot P(X)}{P(Y)}. \quad (1)$$

Because there are usually more (n) attributes X_i, the formula applied is

$$P(X_i|Y) = \frac{P(Y|X_i) \cdot P(X_i)}{\sum_{j=1}^{n} P(Y|X_j) \cdot P(X_j)}. \quad (2)$$

In classification, one is interested in the membership probabilities of an instance to a class, based on the attribute values of that instance. The naïve Bayes classifier computes the conditional probabilities of the classes assuming that all attributes are independent. A Bayesian network is a graph associated with a set of probability tables. Nodes are variables, and arcs are causal relations between variables; therefore, in a Bayesian network, a variable is connected only to the variables it depends on. The advantage of the method is that estimating conditional probabilities for data distribution is the optimal classification method, from the theoretic point of view. The disadvantage lies in the rather complex computations needed, especially when the number of conditional variables is large.

Instance-based learning reduces the learning effort by simply storing the examples presented to the learning agent and classifying the new instances on the basis of closeness to their "neighbors", that is, previously encountered instances with similar attribute value. The nearest-neighbor (NN) algorithm classifies a new instance in the same class as the closest stored instance in the attribute space. A straightforward extension is the k-NN, where k neighbors (instead of 1) are taken into account when determining the membership of an instance to a class. This approach is especially useful when data are affected by noise and the nearest neighbor of an instance may indicate an erroneous class.

The advantage of NN algorithms is the very quick learning (simple storing of exemplars) and high prediction capability. The disadvantage is the slow process of searching the nearest instance by processing all training set.

Nonnested generalized exemplars (NNGE) is an NN-like algorithm using nonnested generalized exemplars [26,27] as a way to avoid all forms of overgeneralization by never allowing exemplars to nest or overlap. NNGE always tries to generalize new examples to their nearest neighbor of the same class, but if this is immediately impossible because of intervening negative examples, no generalization is performed. If a generalization later conflicts with a negative example, it is modified to maintain consistency.

The function used to compute the distance from an instance to a hyperrectangle is

$$d(I,H) = w_H \cdot \sqrt{\sum_{i=1}^{m} \left(w_i \cdot \frac{d_i(I,H)}{maxval_i - minval_i} \right)^2}, \tag{3}$$

with

$$d_i(I,H) = \begin{cases} I_i - H_{upper_i}, & \text{if} \quad I_i > H_{upper_i} \\ H_{lower_i} - I_i, & \text{if} \quad I_i < H_{lower_i} \\ 0, & \text{otherwise} \end{cases}, \tag{4}$$

where w_H is the exemplar (hyperrectangle) weight, w_i are feature weight, I_i is the ith attribute value of I, $maxval_i$ and $minval_i$ are the upper and lower bounds of the domain on dimension i, and H_{upper} and H_{lower} are the upper and lower bounds of the hyperrectangle H.

For symbolic attributes,

$$d_i(I,H) = \begin{cases} 0, & \text{if} \quad I_i \in H_i \\ 1, & \text{if} \quad I_i \notin H_i \end{cases}. \tag{5}$$

RESULTS AND DISCUSSION

All the investigated polymers were synthesized by phase-transfer-catalysis reactions, starting from various bisphenols and α,ω-dihalogenated hydrocarbon compounds. Details concerning the synthesis and characterization procedures were previously reported [28–31].

A typical polymerization reaction, the bisphenols' chemical structures, and abbreviations are presented in Fig. 2.

The investigated liquid-crystalline polymers are systems with the mesogens' groups in the main chain. As a consequence, the most important parameters that influence the liquid-crystalline behavior are the geometric ones. The first step of the study was to establish the input parameters for the categorization algorithms. It must be emphasized that all the investigated polymers contain the azobenzene as the main mesogenic group. The role of the second bisphenol (with

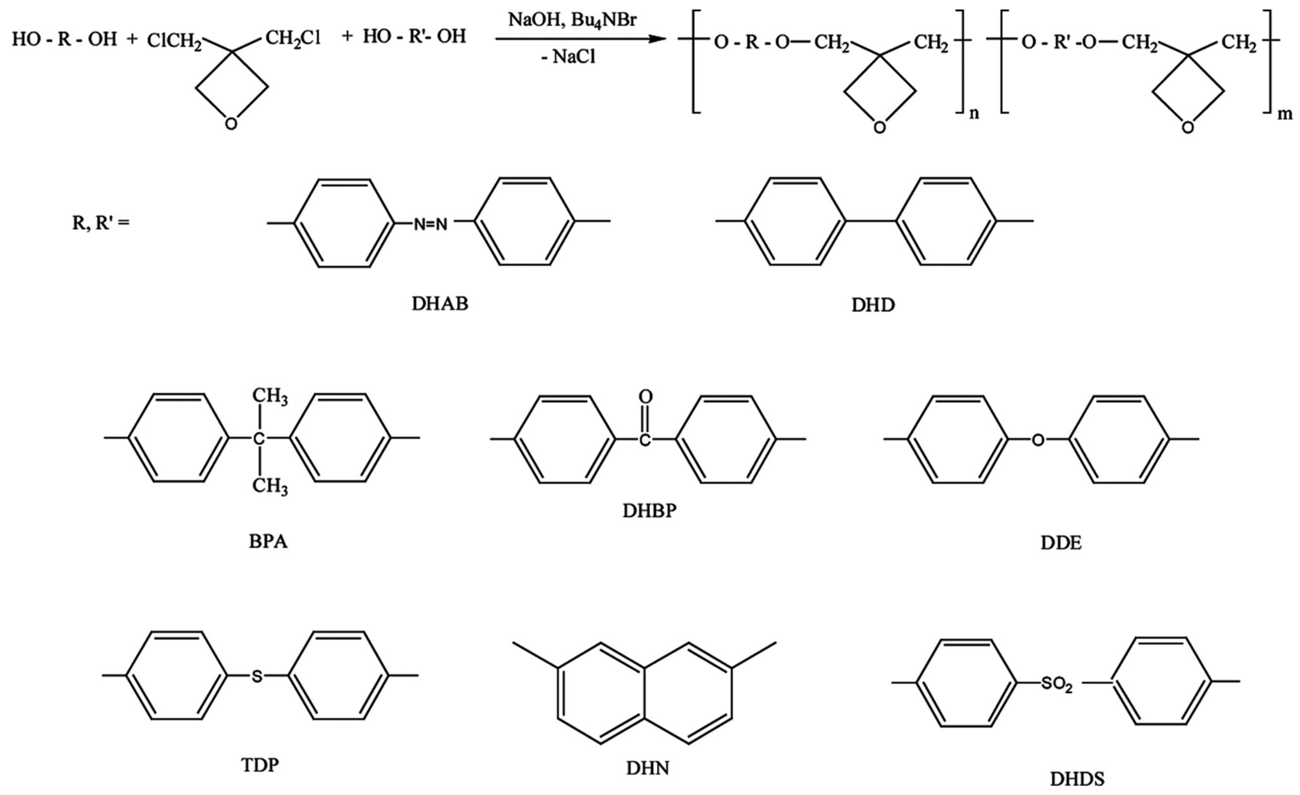

FIGURE 2 Reaction scheme for copolyether synthesis.

different geometries and flexibilities) from the copolymeric structures is to generate a certain disorder degree in the system. This second bisphenol can be 4,4′-dihydroxydiphenyl (DHD), bisphenol A (BPA), 4,4′-dihydroxybenzophenone (DHBP), 4,4′-dihydroxydiphenylether (DDE), 4-hydroxydiphenylsulfide (TDP), 2,7-dihydroxinaphtalene (DHN), or 4,4′-dihydroxydiphenylsulfone (DHFD). The type of bisphenol and the ratio between the 4,4′-dihydroxyazobenzene (DHAB) and the second bisphenol were considered as input parameters (seven symbolic inputs for the seven types of bisphenol and one numerical input for the molar ratio). The inputs corresponding to bisphenol type have 1 or 0 values to indicate the presence or absence, respectively, of the considered type. In the following, the tables with neural-network predictions show the codification of the inputs. An input parameter was also the polymer molecular weight (M_n). Because of the high values of the polymers' temperature transitions and the low solubility, we have synthesized products situated between oligomeric and polymeric domains. In these circumstances, the molecular weight value can help to "discover" the mesophase [32] and can play an important role concerning the liquid-crystalline behavior. Another input parameter was the length of the spacer, as a measure of the soft–hard control. Instead of molecular weight, one can use the asymmetry parameter, s, evaluated using molecular modeling simulation, which represents the ratio between the length and the diameter of the polymer single chain. All these enumerated parameters are considered as inputs of the neural networks because they are the most important structural characteristics that impose a liquid-crystalline behavior.

Concerning the liquid-crystalline behavior, we have coded the possibility to generate a mesophase as "1" and the crystalline or amorphous phases as "0". Two outputs are considered for the neural model, a pair 1–0 for liquid-crystalline behavior and 0–1 as the contrary, and the notation used for these outputs was LC_1 and LC_2. From a conceptual point of view, the neural network could be split into two networks, one each for every class (0 or 1). This formulation increases the reliability degree of the results provided by the neural model.

Table 1 presents the investigated polymers' structures and details concerning the composition, molecular weight, spacer length, asymmetry, and liquid-crystalline or SC (semicrystalline) structure.

A special comment is necessary concerning the spacers. For the first 15 samples (Table 1), an oxetanic structure was used. The oxetane does not permit a complete rotation around the simple bond, because of the sterical hindrance generated between the oxetanic cycle and the bisphenols o-hydrogen. Previous molecular simulations have evidenced rotational barriers of more than 1000 kcal/mol (corresponding

TABLE 1 Main Characteristics of the Investigated Copolymers

Sample code	Copolymer composition (molar ratio)	Molecular weight M_n	Spacer length (Å)	Asymmetry parameter (s)	LC or SC behavior
1	DHAB/DHD = 3.7/1	3000	0	11	LC
2	DHAB/DHD = 1.5/1	2900	0	11	LC
3	DHAB/DHD = 1/2.0	3000	0	10	SC
4	DHAB/BPA = 2.9/1	3100	0	10.88	LC
5	DHAB/BPA = 1.1/1	3000	0	10	SC
6	DHAB/BPA = 1/2.3	3100	0	9.6	SC
7	DHAB/DHD = 3.1/1	3000	0	11	LC
8	DHAB/DHD = 1/1	3000	0	11	SC
9	DHAB/DHBP = 3.3/1	2200	0	11	LC
10	DHAB/DHBP = 2.8/1	2900	0	11	SC
11	DHAB/DHBP = 1.3/1	3500	0	10.38	SC
12	DHAB/DHBP = 1/2.3	2900	0	9.6	SC
13	DHAB/DDE = 1.9/1	3100	0		SC
14	DHAB/DDE = 1.3/1	3300	0		SC
15	DHAB/DDE = 1/1.5	3400	0		SC
16	DHAB/DHD = 2.3/1	1400	2.5	11	LC
17	DHAB/DHD = 1/1.1	1550	2.5	11	LC
18	DHAB/DHD = 1/3.2	1500	2.5	10	SC
19	DHAB/BPA = 3.1/1	1350	2.5	11	SC
20	DHAB/BPA = 1/1.2	1500	2.5	10	SC
21	DHAB/BPA = 1/3.2	1550	2.5	9.46	SC
22	DHAB/TDP = 3.1/1	1650	2.5		SC
23	DHAB/TDP = 1.1/1	1550	2.5		SC
24	DHAB/TDP = 1/2.8	1600	2.5		SC
25	DHAB/DHN = 2.9/1	1500	2.5	10	LC
26	DHAB/DHN = 1.1/1	1550	2.5	9.5	SC
27	DHAB/DHN = 1/2.8	1700	2.5	8.4	SC
28	DHAB/DHD = 2.5/1	2400	3.9	11	LC
29	DHAB/DHD = 1/1.1	2550	3.9	11	LC
30	DHAB/DHD = 1/3.1	2700	3.9	10	SC
31	DHAB/BPA = 2.5/1	2800	3.9	11	LC
32	DHAB/BPA = 1/1.5	3000	3.9	9.9	LC
33	DHAB/BPA = 1/3.5	2900	3.9	9.4	SC
34	DHAB/DHN = 2.5/1	2100	3.9	10.34	SC
35	DHAB/DHN = 1/1.2	2000	3.9	9.2	SC
36	DHAB/DHN = 1/2.9	1800	3.9	8.3	SC
37	DHAB/DHD = 2.5/1	2300	5.1	11.11	LC
38	DHAB/DHD = 1/1.1	2200	5.1	11	LC
39	DHAB/DHD = 1/2.6	2400	5.1	10	LC
40	DHAB/BPA = 2.6/1	3200	5.1	11	SC
41	DHAB/BPA = 1/1.2	2700	5.1	10	SC
42	DHAB/BPA = 1/3.5	3400	5.1	9.4	SC

(*Continued*)

TABLE 1 Continued

Sample code	Copolymer composition (molar ratio)	Molecular weight M_n	Spacer length (Å)	Asymmetry parameter (s)	LC or SC behavior
43	DHAB/DHN = 2.2/1	2700	5.1	10	SC
44	DHAB/DHN = 1/1.2	2000	5.1	9.2	SC
45	DHAB/DHN = 1/1.2	2200	5.1	9.2	SC
46	DHAB/DHD = 2.5/1	2100	6.4	11	LC
47	DHAB/DHD = 1/1.2	2300	6.4	11	LC
48	DHAB/DHD = 1/3.1	2300	6.4	10	LC
49	DHAB/BPA = 2.5/1	2200	6.4	11	SC
50	DHAB/BPA = 1/1.1	2500	6.4	10	LC
51	DHAB/BPA = 1/3.2	3400	6.4	9.5	SC
52	DHAB/DHN = 2.9/1	1800	6.4	10	SC
53	DHAB/DHN = 1.1/1	1900	6.4	9.5	SC
54	DHAB/DHN = 1/2.6	1800	6.4	8.4	SC
55	DHAB/DHDS = 2.6/1	2100	6.4		SC
56	DHAB/DHDS = 1/1.2	2100	6.4		SC
57	DHAB/DHDS = 1/3.2	3000	6.4		SC
58	DHAB/DHD = 2.8/1	3200	7.7	11	LC
59	DHAB/DHD = 1/1	3100	7.7	11	LC
60	DHAB/DHD = 1/2.9	2900	7.7	10	LC
61	DHAB/DHD = 2.8/1	2600	8.9	11	LC
62	DHAB/DHD = 1/1.2	2400	8.9	10.67	LC
63	DHAB/DHD = 1/2.6	2300	8.9	10	LC
64	DHAB/BPA = 2.4/1	2600	8.9	11	LC
65	DHAB/BPA = 1/1.7	2500	8.9	9.8	SC
66	DHAB/BPA = 1/3.7	3000	8.9	9.4	SC
67	DHAB/DHN = 2.4/1	3200	8.9	10	SC
68	DHAB/DHN = 1/1.2	2600	8.9	9.2	SC
69	DHAB/DHN = 1/2.9	2700	8.9	8.3	SC
70	DHAB/DHBP = 2.6/1	2600	8.9	11	LC
71	DHAB/DHBP = 1.3/1	2300	8.9	10	LC
72	DHAB/DHBP = 1/2.2	2350	8.9	9.7	SC
73	DHAB/DHD = 2.6/1	2600	10.2	11	LC
74	DHAB/DHD = 1/1	2500	10.2	11	LC
75	DHAB/DHD = 1/2.4	2850	10.2	10	LC
76	DHAB/BPA = 2.7/1	3100	10.2	11	LC
77	DHAB/BPA = 1/1.4	2800	10.2	10	SC
78	DHAB/BPA = 1/3.3	2800	10.2	9.5	SC
79	DHAB/DHN = 2.8/1	2800	10.2	10	SC
80	DHAB/DHN = 1.2/1	2600	10.2	9.6	SC
81	DHAB/DHN = 1/3	2700	10.2	8.3	SC
82	DHAB/DHBP = 2.9/1	2450	10.2	11	SC
83	DHAB/DHBP = 1.1/1	2300	10.2	10	SC
84	DHAB/DHBP = 1/2.7	2350	10.2	9.6	SC

to the simple bonds that connect the oxetanic cycle with the main chain) that permit us to consider them as semirigid spacers [33,34]. We have estimated that for the oxetanic polymers, the ordering process takes place at a polymeric chain level (a rod-like behavior), and as a consequence, the length of the spacer was considered to be 0 Å.

The spacer length and the asymmetry parameters were estimated by mechanical molecular simulation using the Hyperchem program.

Consequently, 10 input parameters for the neural model are considered: 7 columns with 1 and 0 values for the type of bisphenol, molar ratio between the DHAB and the second bisphenol, polymer molecular weight and the length of the spacer. The output parameters are the pair 1–0 for liquid-crystalline behavior and 0–1 for SC structure. This is referred to as problem 1. The second approach, problem 2, supposes the same outputs and the following 10 input variables: 7 for the type of bisphenol, molar ratio between the DHAB and the second bisphenol, the length of the spacer, and the asymmetry parameter. In this way, two variants for combining different parameters as categorization algorithm inputs are considered.

The neural-network modeling implies the following stages: collecting the training data by experiments, making up the training and testing data sets, developing the neural-network topology, training, and finally establishing the performance of the neural-network model by comparing the network prediction to unseen (validation) data.

The success in obtaining a reliable and a robust network depends strongly on the choice of the process variable involved, the available set of data, and its domain used for training, as well as the training method.

After the establishment of modeling purpose (input and output variables), one important problem in developing a layered neural network is to determine the network architecture, that is, the number of hidden layers and the number of neurons in each hidden layer. A feedforward neural network (MLP, multilayer perceptron) was chosen because it is a simple type that is able to model systems with different degrees of complexity and nonlinearity. It is generally accepted that a large number of hidden layers do not necessarily improve the performance and increases the difficulties in training. Determining the number of hidden nodes depends on the nonlinearity of the problem and the error tolerance. Too many hidden nodes (an oversized network) cause the network to memorize the training set (*i.e.*, overfitting), leading to poor generalization performance. Too few hidden nodes may not achieve the required error tolerance (*i.e.*, underfitting) and create difficulties in representing the nonlinear processes.

In our work, the number of hidden layers and units was established by training many networks and selecting the one that best balanced generalization performance against network size. The networks were trained using the back-propagation algorithm. Once the data have been fed into the neural networks, the weights were updated continuously based upon the back-propagation learning rule.

Before training, the data are split into training and validation data sets because it is more important to evaluate the performance of the neural networks on unseen data than on training data. In this way, we can estimate the most important feature of a neural model—the generalization capability.

The best network topology was determined based upon the mean squared errors (MSE) of the training data. The MSE was computed using the following formula:

$$\mathrm{MSE} = \frac{\sum_{j=1}^{P}\sum_{i=1}^{N}(d_{ij} - y_{ij})^2}{N \cdot P}, \qquad (6)$$

where P is the number of output processing elements (in this case, $P = 1$), N is the number of exemplars in the data set, y_{ij} is the network output for exemplar i at processing element j, and d_{ij} is the desired output for exemplar i at processing element j.

We developed and trained many networks, changing these options, and then we selected the best that balances the size and the performance. Table 2 presents several examples. The column "Network topology" contains the number of neurons in input, hidden, and output layers, "MSE" is the mean squared error, and "r" represents the correlation between training data and the answers of the networks.

All the neural networks in Table 2 present good performances and relatively simple topologies—one or two intermediate layers and a small number of hidden neurons.

Table 3 presents some predictions of the neural models to previously unseen data (not used in the training phase, so "unseen" data for the networks). In Table 3, one can also see the nominal and

TABLE 2 Different Neural Network Topologies

		Training performance	
No.	Network topology	MSE	r
1	MLP(10:5:2)	0.000674	0.9996
2	MLP(10:10:2)	0.000496	0.9997
3	MLP(10:50:2)	0.000052	0.9998
4	MLP(10:12:4:2)	0.000024	0.9999

TABLE 3 Validation of the Neural Models for the First Problem

	Experimental data												Neural networks predictions					
													MLP (10:5:2)		MLP (10:50:2)		MLP (10:12:4:2)	
Sample code	DHD (I_1)	BPA (I_2)	DHBP (I_3)	DDE (I_4)	TDP (I_5)	DHN (I_6)	DHDS (I_7)	Molar ratio (I_8)	M_n (I_9)	Spacer length (I_{10})	LC_1 (O_1)	LC_2 (O_2)	LC_1	LC_2	LC_1	LC_2	LC_1	LC_2
6	0	1	0	0	0	0	0	0.43	3100	0	0	1	0	1	0	1	0	1
10	0	0	1	0	0	0	0	2.80	2900	0	0	1	0	1	0	1	0	1
22	0	0	0	0	1	0	0	3.10	1650	2.5	0	1	1	1	0	1	1	0
23	0	0	0	0	1	0	0	1.10	1550	2.5	0	1	0	1	0	1	0	1
32	0	1	0	0	0	0	0	0.67	3000	3.9	1	0	0	0	0	1	0	1
37	1	0	0	0	0	0	0	2.50	2300	5.1	1	0	1	0	1	0	1	0
42	0	1	0	0	0	0	0	0.29	3400	5.1	0	1	0	1	0	1	0	1
46	1	0	0	0	0	0	0	2.50	2100	6.4	1	0	1	0	1	0	1	0
51	0	1	0	0	0	0	0	0.31	3400	6.4	0	1	0	1	0	1	0	1
56	0	0	0	0	0	0	1	0.83	2100	6.4	0	1	0	1	0	1	0	1
60	1	0	0	0	0	0	0	0.34	2900	7.7	1	0	1	0	1	0	1	0
64	0	1	0	0	0	0	0	2.40	2600	8.9	1	0	1	0	0	0	1	0
68	0	0	0	0	0	1	0	0.83	2600	8.9	0	1	0	1	0	1	0	1
72	0	0	1	0	0	0	0	0.45	2350	8.9	0	1	0	1	0	1	0	1
75	1	0	0	0	0	0	0	0.42	2850	10	1	0	1	0	1	0	1	0
79	0	0	0	0	0	1	0	2.80	2800	10	0	1	0	1	0	1	0	1
83	0	0	1	0	0	0	0	1.10	2300	10	0	1	1	1	1	0	0	1

*Cells marked in grey represent incorrect predictions of the networks.

numerical codifications of the inputs ($I_1 \ldots I_{10}$) and outputs (O_1, O_2) for the networks. Cells marked in grey represent incorrect predictions of the networks. For MLP(10:5:2) and MLP(10:50:2), the probability of a correct answer was 82.35%, and for MLP(10:12:4:2), it was 88.23%, that is, a good performance of the designed networks. Consequently, a feedforward network MLP(10:12:4:2) can predict satisfactory the liquid-crystalline behavior of the copolyethers.

For the second problem, MLP(10:12:4:2) with MSE $= 0.000014$ and $r = 0.9999$ was determined as the best neural network, by trial and error. Data in Table 4 show the efficiency of the neural model, which has a probability of a correct answer of 92.86% compared to validation data set. We cannot consider as an absolute conclusion the fact that the choice of *s* parameter as input variable leads to the best results, but an important step is the choice of the properties and structural parameters in such a combination that allows us to evaluate the liquid-crystalline behavior with high probability.

NeuroSolutions, a software application dedicated to the study of neural networks, was used to design and obtain predictions of neural networks. In this program, the following specifications are necessary: the network type, the input and desired output values, the stop condition of the training, the number of processing elements in hidden layers, the activation functions, the learning rule, the maximum number of epochs, and some configuration parameters to display the neural model development.

Because classification is an important problem in machine learning, several classification methods have been tried to analyze and compare their results on two benchmark problems.

In the following, we present a performance comparison between the presented algorithms: C4.5 (with pruned and unpruned tree, respectively), random tree, random forest, naïve Bayes, NN, and *k*-NN (with $k = 4$), and nonnested generalized exemplars.

We considered both the error on the whole training set and the error found by dividing the data into training and testing sets. In this way, one can estimate the generalization capabilities of the models built by the classifiers.

Special attention must be paid to the data representation for the algorithms that are essentially numerical. In the case of neural networks, for each discrete value of a symbolic attribute a different input or output must be considered that will take the value of 1 or 0 depending on the activation of the corresponding attribute value. For the other numerical algorithms, such as those based on the NN paradigm, the only problem is to express the "distance" between two discrete values as Eq. (5) shows. A major issue for this class of algorithms is

TABLE 4 Validation of the Neural Model for the Second Problem

	Experimental data									Neural networks predictions	
										MLP (10:12:4:2)	
Sample code	DHD (I_1)	BPA (I_2)	DHBP (I_3)	DHN (I_6)	Molar ratio (I_8)	Spacer length (I_9)	Asymmetry parameter (I_{10})	LC_1 (O_1)	LC_2 (O_2)	LC_1	LC_2
1	1	0	0	0	3.70	0	11	1	0	1	0
6	0	1	0	0	0.43	0	9.6	0	1	0	1
10	0	0	1	0	2.80	0	11	0	1	0	1
26	0	0	0	1	1.10	2.5	9.5	0	1	0	1
40	0	1	0	0	2.60	5.1	11	0	1	0	1
43	0	0	0	1	2.20	5.1	10	0	1	0	1
47	1	0	0	0	0.83	6.4	11	1	0	1	0
60	1	0	0	0	0.34	7.7	10	1	0	1	0
65	0	1	0	0	0.59	8.9	9.8	0	1	0	1
69	0	0	0	1	0.34	8.9	8.3	0	1	0	1
73	1	0	0	0	2.60	10.2	11	1	0	1	0
77	0	1	0	0	0.71	10.2	10	0	1	1	0
81	0	0	0	1	0.33	10.2	8.3	0	1	0	1
84	0	0	1	0	0.37	10.2	9.6	0	1	0	1

*Cells marked in grey represent incorrect predictions of the networks.

the distance metric used to compute the influence of the neighbors. For the NN and *k*-NN algorithms, we chose the inverse of distance in a normalized Euclidian space to represent the strength of a particular neighbor to the instance being classified.

For categorization algorithms, the different types for the second bisphenol were codified by nominal values as follows: 1, DHD; 2, BPA; 3, DHBP; 4, DDE; 5, TDP; 6, DHN; and 7, DHDS.

Table 5 shows the performances of the neural networks and classification algorithms for the two considered problems. Problem 1 refers to the input parameters: molar ratio, type of bisphenol, molecular weight, and spacer length, and problem 2 is formulated for the molar ratio, type of bisphenol, spacer length, and asymmetry parameter as input parameters.

Regarding the actual results, the pruned decision tree for the first problem is shown:

```
Copolyethers = 1
  Spacer length ≤ 3.9
    Molar ratio ≤ 0.67 → class = 0
    Molar ratio > 0.67 → class = 1
  Spacer length > 3.9 → class = 1
Copolyethers = 2,4,5,6,7 → class = 0
Copolyethers = 3
  Spacer length ≤ 8.9
    Molar ratio ≤ 2700 → class = 1
    Molar ratio > 2700 → class = 0
  Spacer length > 8.9 → class = 0
```

TABLE 5 Error Rates of the Considered Classification Algorithms

	Problem 1		Problem 2	
Algorithm	Training %	Validation %	Training %	Validation %
Neural network (1H-5)	0	17.65	0	10.71
Neural network (2H-12:4)	0	11.76	0	7.14
C4.5 pruned	10.71	20.69	12	32.0
C4.5 unpruned	8.33	17.24	12	34.61
Random tree	0	28.57	0	32
Random forest (100 trees)	0	17.85	0	32
Naïve Bayes	13.09	17.24	16	30
Nearest neighbor	0	13.79	0	26.92
4-Nearest neighbor	0	13.79	0	23.07
NNGE	0	17.24	0	26.92

The tree should be interpreted as “if `Copolyethers` is 1, then if `Spacer length` is less than or equal to 3.9, and `Molar ratio` is less than or equal to 0.67, then the `class` is 0. Otherwise, if `Molar ratio` is greater than 0.67, the `class` is 1. If `Copolyethers` is 1, and `Spacer length` is greater than 3.9, the `class` is 1,” and so on.

The prior probabilities of the two classes, as given by the naïve Bayes method, are 0.62 for class 0 and 0.38 for class 1. In the one-third/two-thirds split method (2/3 of data for training and 1/3 for validation phase) for the first problem, 3 instances actually belonging to class 0 are incorrectly classified as class 1, and 2 instances actually belonging to class 1 are incorrectly classified as class 0, out of 29 instances in the prediction third of the dataset.

The NN methods do not provide explicit results in form of rules. The training instances are all memorized, and the new instances are classified by computing its conceptual “closeness” to one or more instances in memory.

The NNGE results for the first problem are presented as follows in form of rules (in fact hyperrectangles with one or more member instances):

`class 0`

- `Copolyethers in {3,5,6,7} and 0.31<=Molar ratio≤1.2 and 1550.0<=Molecular weight<= 3000.0 and 2.5<=Spacer length<=10.2 (19 instances)`
- `Copolyethers in {3} and Molar ratio = 2.9 and Molecular weight = 2450.0 and Spacer length = 10.2 (1 instance)`
- `Copolyethers in {6} and 2.4<=Molar ratio≤ 2.8 and 2800.0<=Molecular weight<=3200.0 and 8.9<=Spacer length<=10.2 (2 instances)`
- `Copolyethers in {2} and Molar ratio = 2.5 and Molecular weight = 2200.0 and Spacer length = 6.4 (1 instance)`
- `Copolyethers in {1,2,3,4} and 0.43<=Molar ratio≤1.3 and 2900.0<=Molecular weight<= 3500.0 and Spacer length = 0.0 (8 instances)`
- `Copolyethers in {3,4,5,6,7} and 1.9<=Molar ratio≤3.1 and 1650.0<=Molecular weight<= 3100.0 and 0.0<=Spacer length<=6.4 (7 instances)`
- `Copolyethers in {1,2} and 0.27<=Molar ratio≤0.32 and 2700.0<=Molecular weight<=`

3400.0 and 3.9<=Spacer length<=10.2 (6 instances)
- Copolyethers in {2} and 0.71<=Molar ratio $\leq$ 0.83 and 2700.0<=Molecular weight <=2800.0 and 5.1<=Spacer length<=10.2 (2 instances)
- Copolyethers in {2} and Molar ratio = 0.59 and Molecular weight = 2500.0 and Spacer length = 8.9 (1 instance)
- Copolyethers in {2} and Molar ratio = 2.6 and Molecular weight = 3200.0 and Spacer length = 5.1 (1 instance)
- Copolyethers in {2} and Molar ratio = 3.1 and Molecular weight = 1350.0 and Spacer length = 2.5 (1 instance)
- Copolyethers in {1,2} and 0.31<=Molar ratio $\leq$ 0.83 and 1500.0<=Molecular weight<= 1550.0 and Spacer length = 2.5 (3 instances)

class 1

- Copolyethers in {2,3} and 1.3<=Molar ratio $\leq$ 2.7 and 2300.0<=Molecular weight<=3100.0 and 8.9<=Spacer length<=10.2 (4 instances)
- Copolyethers in {1,2} and 0.34<=Molar ratio $\leq$ 0.67 and 2400.0<=Molecular weight<= 3000.0 and 3.9<=Spacer length<=7.7 (3 instances)
- Copolyethers in {1,2} and 1.5<=Molar ratio $\leq$ 3.7 and 1400.0<=Molecular weight<=3100.0 and 0.0<=Spacer length<=5.1 (8 instances)
- Copolyethers in {1} and 0.38<=Molar ratio $\leq$ 0.83 and 2300.0<=Molecular weight<=2850.0 and 8.9<=Spacer length<=10.2 (3 instances)
- Copolyethers in {1,2} and Molar ratio = 0.91 and 1550.0<=Molecular weight<=2550.0 and 2.5<=Spacer length<=6.4 (4 instances)
- Copolyethers in {1} and 1.0<=Molar ratio $\leq$ 2.8 and 2100.0<=Molecular weight<=3200.0 and 6.4<=Spacer length<=10.2 (6 instances)
- Copolyethers in {3} and Molar ratio = 3.3 and Molecular weight = 2200.0 and Spacer length = 0.0 (1 instance)

- `Copolyethers in {1} and 0.32<=Molar ratio`$\leq$ `0.83 and Molecular weight=2300.0 and Spacer length=6.4 (2 instances)`
- `Copolyethers in {6} and Molar ratio = 2.9 and Molecular weight=1500.0 and Spacer length = 2.5 (1 instance)`

Similar results are found for the second problem. These rules are interpreted in the following manner. For example, the rule for class 0, "`Copolyethers` in {3,5,6,7} and $0.31 \leq$ `Molar ratio` ≤ 1.2 and $1550.0 \leq$ `Molecular weight` ≤ 3000.0 and $2.5 \leq$ `Spacer length` ≤ 10.2 (19 instances)" means that if an instance has the `Copolyethers` value 3, 5, 6, or 7, and `Molar ratio` $\in [0.31, 1.2]$, and so on, then the class of that instance is 0. This rule is formed by a set of 19 training instances.

Although the symbolic methods (such as decision trees) are less effective, they have the advantage of an explicit representation of the classification model. C4.5 generally has good error rates over the training set (especially in the unpruned version) and acceptable generalization capabilities.

Neural networks have the best performance. However, the structure of the categorization model is implicit, as it is given by the connection weights. Also, there are many free parameters involved in neural networks, such as choosing the best topology, the learning rate, and the momentum factor when trying to accelerate the learning process, which is rather slow.

The Bayesian classifier generalizes well, because it is based on the estimation of the probability distribution of data. Unfortunately, it has high error rates in both situations.

An interesting approach is the NN paradigm, and especially k-NN, which proves to be a good choice for both problems. Like the neural networks, it does not provide a structure for the data. The speed and simplicity of the learning process is counterweighted by the prediction phase, which must employ a search process through the already learned instances to find the desired class label.

Our future research will focus on the adaptation of the categorization algorithms for different particular problems by combining different principles that characterize the algorithms.

CONCLUSIONS

The prediction of the mesophase occurrence with machine learning methods, as well as the choice and the codification (numerical and nominal) of different sets of parameters that characterize the structure and the behavior of the studied copolyeters, represents a new approach in the field.

The most important parameters that influence the liquid-crystalline behavior and constitute the inputs of categorization algorithms are the qualitative and quantitative composition of the copolyethers, the polymer molecular weight, the length of the spacer as a measure of the soft–hard control, and an asymmetry parameter.

Machine learning methods used in this approach to predict the liquid-crystalline behavior of some copolyethers are based on feed-forward neural networks and different categorization algorithms: decision trees, NN, and Bayesian induction, implemented in some variants: C4.5 pruned, C4.5 unpruned, random tree, random forest, naïve Bayes, NN, 4-NN, and NNGE.

Neural networks and *k*-NN prove to have the best performance (7.14% and 23% error, respectively, in the validation phase, for the second problem). However, the structure of the neural categorization model is implicit, as it is given by the connection weights. Also, there are many free parameters involved in neural networks, such as choosing the best topology, the learning rate, and the momentum factor to accelerate the learning process, which is rather slow. Although the neural network method yields the best results and the prediction/pro-production phase is very fast, the process of finding the best network may be difficult. On the other hand, the *k*-NN algorithm is very simple, with very few parameters to be taken into account. Its main disadvantage is that it is slow for a large number of training instances (which is not the case here).

REFERENCES

[1] Osei-Bryson, K. M. & Giles, K. (2004). *An Exploration of a Set of Entropy-Based Hybrid Splitting Methods for Decision Tree Induction*, Idea Group Publishing Hershey, PA, USA.
[2] Papadopoulos, A. N. & Manolopoulos, Y. (2004). *Nearest Neighbor Search: A Database Perspective*, Springer-Verlag, New York.
[3] Lee, P. M. (2004). *Bayesian Statistics: An Introduction*, Arnold Publishers, London.
[4] Jensen, F. V. (2001). *Bayesian Networks and Decision Graphs*, Springer, New York.
[5] Mira, J. & Álvarez, J. R. (Ed.). (2003). *Computational Methods in Neural Modeling*, Springer, New York.
[6] Landrum, J., Penzotti, J., & Putta, S. (2005). *Meas. Sci. Technol.*, *16*, 270–277.
[7] Yan, C, Honavar, V., & Dobbs, D. (2004). *Neural Comput. Applic.*, *13*, 123–129.
[8] Landrum, G. A. & Genin, H. (2003). *J. Solid State Chem.*, *176*, 587–593.
[9] Villars, P. (2000). *Eng. Appl. Artif. Intell.*, *13*, 497–505.
[10] Cundari, T. R., Deng, J., Pop, H. F., & Sarbu, C. (2000). *J. Chem. Inf. Comput. Sci.*, *40*, 1052–1061.

[11] Cundari, T. R., Sarbu, C., & Pop, H. F. (2002). *J. Chem. Inf. Comput. Sci.*, *42*, 1363–1369.

[12] Van Krevelen, D. W. (1990). *Properties of Polymers*, 3rd ed., Elsevier: Amsterdam.

[13] Bicerano, J. (2002). *Prediction of Polymers Properties*, 3rd ed., Marcel Dekker: New York.

[14] Kier, L. B. & Hall, L. H. (1986). *Molecular Connectivity in Structure–Activity Analysis*, John Wiley & Sons: New York.

[15] Maier, W. A. & Saupe, Z. (1959). *Naturforsch*, *A14*, 882–900.

[16] de Gennes, P. G. & Prost, J. (1993). *The Physics of Liquid Crystals*, Clarendon: Oxford.

[17] Sen, A. K. & Sullivan, D. E. (1987). *Phys. Rev. A*, *35*, 1391–1403.

[18] de Gennes, P. G. (1979). *Scaling Concepts in Polymer Physics*, Cornell University Press: Ithaca, NY.

[19] Doi, M. & Edwards, S. F. (1986). *The Theory of Polymer Dynamics*, Clarendon Press: Oxford.

[20] Jain, S. & Nelson, D. (1996). *Macromolecules*, *29(26)*, 8523–8529.

[21] *LiqCryst 3.4—Database of liquid crystalline compounds*. http://dwb.unl.edu/Teacher/NSF/C01/C01Links/liqcryst.chemie.uni-hamburg.de/lc/

[22] Piotto, S. P. (2002). *CURVIS, bending energy calculator*, 2nd release. ETH Zürich. Piotto, S. P. (2002). *FRACTALS, numerical curvature evaluation*. ETH Zürich. http://www.polymers.unisa.it/members/piotto/about.htm.

[23] Quinlan, R. (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers: San Mateo, CA.

[24] Witten, I. H. & Frank, E. (2000). *Data Mining: Practical Machine Learning Tools with Java Implementations*, Morgan Kaufmann: San Francisco.

[25] Breiman, L. (2001). *Machine Learning*, *45*(1), 5–32.

[26] Martin, B. (1995). *Instance-based learning: nearest neighbour with generalisation*. Master of Science Thesis, University of Waikato, Hamilton, New Zealand.

[27] Leon, F., Zaharia, M. H., Gâlea, D. (2004). "Performance analysis of categorization algorithms," in *Proceedings of the Eighth International Symposium on Automatic Control and Computer Science*, Politechnium, Iaşi, Romania.

[28] Alazaroaie, S., Catanescu, O., Toader, V., Taran, E., Pavel, V., Hurduc, N., Scutaru, D., & Simionescu, C. I. (2005). *High Performance Polymers*, *17*(1), 149–160.

[29] Alazaroaie, S., Toader, V., Carlescu, I., Kazmierski, K., Scutaru, D., Hurduc, N., & Simionescu, C. I. (2003). *Eur. Polym. Journal*, *39*, 1333.

[30] Catanescu, O., Hurduc, N., Scutaru, D., Toader, V., Stoleru, A., & Simionescu, C. I. (1999). *Die Angew. Macrom. Chemie*, *273*, 91–95.

[31] Hurduc, N., Daoudi, A., Buisine, J. M., Barboiu, V., & Simionescu, C. I. (1998). *Eur. Polym. J.*, *34*, 123.

[32] Keller, A., Ungar, G., & Percec, V. (1990). Liquid crystal polymers: A unifying thermodynamics based scheme. In: *Advances in Liquid Crystalline Polymers*, Weiss, R. A. & Ober, C. K. (Eds.), ACS Symposium Series 435, ACS: Washington, DC.

[33] Hurduc, N., Surpateanu, G., & Bulacovschi, V. (1992). *Eur. Polym J.*, *28*, 1589–1591.

[34] Hurduc, N., Bulacovschi, V., & Surpateanu, G. (1992). *Polym Bull.*, *28*, 639–644.