Koniocortex-Like Network Application to Business Intelligence

J. Fombellida D. Andina

Abstract—Koniocortex-Like Network model is a Bio-Inspired Neural Network structure that tries to replicate the architecture and properties of the biological koniocortex section of the brain. The structure is composed by different kinds of artificial neurons that interplay between them to create a competitive model that can be used to classify patterns. The classification performance obtained is based on different properties like lateral inhibition, metaplasticity and intrinsic plasticity, that allows a natural evolution of the network until obtaining the desired results. This kind of network has been applied to synthetic and real data showing big potential, now the network capabilities are tested using other state-of-the-art real data application: the classification of credit data from the Australian Credit Approval Database.

Index Terms—Metaplasticity, Koniocortex, KLN, Business Intelligence, ACAD, Feature Extraction, Competition

I. INTRODUCTION

The human brain presents a region inside the cerebral cortex in the form of a granular layer (layer IV) that is called koniocortex. This brain area contains abundance of spiny stellae neurons that gives this layer its granular texture, these neurons are directly connected to the thalamus. The thalamus is the main relay station from the senses to the cortex and provide neural projections to the spiny neurons of the koniocortex.

The Koniocortex-Like Network (KLN) is an Artificial Neural Network (ANN) model based on an unsupervised learning paradigm, that is composed by two layers that try to replicate the behavior of its biological counterpart. The first layer contains neurons similar to the thalamocortical neurons of the thalamus, and the second layer include the neurons that resemble the spiny and inhibitory interneurons that form the biological koniocortex. This two-layered structure constitute a network capable of classifying patterns using its competition and auto-organization properties.

According to [1] the real koniocortex can be considered as a competitive network due to the fact that only a small number of spiny neurons are active when the thalamus connection provides sensory stimuli to this region. This is similar to the Winners Takes All (WTA) paradigm that is included in many unsupervised network models so the biological structure and properties have been used as inspiration for this new artificial model. The main difference between the classic ANN approach for the WTA and the KLN is that while conventional competitive neurons detects the most active neuron using external calculations, in the case of the KLN the winning neuron emerges naturally from the interaction between neurons inside the network dynamics. In addition the non-winning neurons are silenced due to the activity of its neighbor neurons, not because they are algorithmically reset.

Considering the KLN model the main properties involved in competitive learning are synaptic metaplasticity and intrinsic plasticity. Intrinsic plasticity adjusts the global excitability of the neuron so that highly excited neurons will be less excitable in the future, and vice versa. This article is in line with the works presented previously [2], [3], [4], [5], [6], [7]. The KLN is applied to the classification of the patterns included in the Australian Credit Approval Database (ACAD) [8]. The results are evaluated considering the following performance figures: sensitivity, specificity and accuracy, and the validation is made using the 10-fold cross validation method. The paper is organized as follows. Section 2 presents the Business Intelligence (BI) concept and the application of bio-inspired systems to its specific problems. Section 3 presents a detailed description of the database and the algorithms. In section 4 the experimental results obtained are shown, a brief discussion of these results is included and a comparison with other state-of-the-art algorithms is presented. Finally, section 5 summarizes the main conclusions.

II. BUSINESS INTELLIGENCE AND ANN APPLICATION

Business Intelligence (BI) solutions applied to business services and utilities is considered as one of the principal possibilities inside the development of what it is called Industry 4.0 revolution. System networks need powerful intelligence systems to process all the available information coming from the different users to study their behavior so services and business results can be improved based on the existing knowledge embedded on Business Data (BD).

The constant evolution on technology research provide new solutions that introduce new technical and organizational concepts. Companies are switching to IT-driven efficient management of complex information [9], [10], that performs an added value processing of the information coming from the different actors. This processing improves efficiency, reliability
and economy. New business processes are being established to accommodate these technologies and market evolutions.

BI allows collecting, correlating and analyzing events from multiple sources by taking advantage from existing tools while gaining end-to-end management of the entire network. BI is the practice of interpreting the data to make useful business-oriented decisions. BI decision support applications facilitate this multi-dimensional analysis like online analytical processing, click-stream analysis, balance score-card, preparation, visualization, querying, reporting, charting, data mining for text content and voice, forecasting, geospatial analysis, enterprise portal implementation, knowledge management, digital dashboard access and other cross-functional activities.

According to [11] BI requires tools and technologies focused on enhanced decision making. The results obtained are normally used in supply chains, sales, finances and marking. BI generates reliable decisions that reduces the risks associated to the inconsistencies in the classic decision-making process [12]. So BI systems are well recognized as important contributors for decision-making. BI systems are most commonly identified as technological solutions holding quality information in well-designed data stores, enabling them to make the right decisions or take the right actions.

BD and BI systems complexity make their evaluation difficult. Due to the multi-disciplinary aspects involved in these systems, a classic engineering approach may not provide a successful adequate solution [13]. To tackle the complex problem of providing BI solutions based on BD, bio-inspired ANNs have to be considered, because these systems have the potential to provide a more efficient and effective use of explicit and implicit knowledge present in the BD.

This is the case that we are presenting in this article, a novel structure and learning algorithm inspired in different biological properties like metaplasticity and intrinsic plasticity. These basic characteristics of the neural connections are believe to be crucial in achieving the biological Deep Learning that allows biological brains to successfully deal with real-world complex problems. In the experiments that have been performed in the frame of this investigation, the KLN method is applied to the classification of the ACAD and the resulting system achieve performance that is comparable with the most powerful methods of the state-of-the-art with the additional advantage that being an unsupervised learning algorithm there is no need of previous information about the classification result of the patterns.

III. MATERIAL AND METHODS

A. Credit Scoring Method

The financial crisis has put the focus of investigation in banking issues, specially in the ones related to the approval of credits. Until recently, the decisions related to credit loans were based on individual perceptions and the human capacity to assess the risk. The growing demand for credit has led to the use of a statistician method, known as Credit Scoring, to decide whether to or not to grant credit.

This method is widely used for consumer loans, and it is getting more used for commercial loans. The credit score is a binary classification task of basic finance. An advantage of the credit score method is the reduction of the costs of credit analysis: faster credit decisions, greater control and reduced potential risks.

B. ACAD Data Preparation

The database contains 690 cases, divided into two classes, 307 applicants “accepted” and 383 applicants “rejected”. Each applicant has 15 features, including 6 nominal, 8 numeric attributes and the last one is the label of each class (accepted or rejected). This dataset is interesting because there is a good mix of attributes: continuous and nominal, nominal ratings with small and large values. Another important characteristic is that few values are missing.

Normally the classifiers based on neural networks produce better results if the training sets are balanced, presenting the same number of patterns belonging to each one of the possible classes. Considering this an adaptation of the data set is needed so some accepted patterns have been repeated instead of eliminating rejected patterns to avoid a potential loss of information.

Depending on the concrete inputs used for training and for performance evaluation it is possible to have a numerical influence on the results. To obtain results statistically independent of the distribution of the patterns a 10 fold cross validation evaluation method has been considered. Using this method the possible dependence of the results with the distribution of the samples in the training or performance evaluation sets is eliminated: all the samples are used to train the networks and all the samples are used to evaluate the performance of the results. This is applied to different executions of the experiment for the same initial neural networks, and finally mean values are calculated to establish the final performance results.

For this experiment we have created ten data sets from the ACAD with the following distribution of patterns:

- G1: 78 total patterns: 39 rejected and 39 accepted
- G2: 78 total patterns: 39 rejected and 39 accepted
- G3: 78 total patterns: 39 rejected and 39 accepted
- G4: 76 total patterns: 38 rejected and 38 accepted
- G5: 76 total patterns: 38 rejected and 38 accepted
- G6: 76 total patterns: 38 rejected and 38 accepted
- G7: 76 total patterns: 38 rejected and 38 accepted
- G8: 76 total patterns: 38 rejected and 38 accepted
- G9: 76 total patterns: 38 rejected and 38 accepted
- G10: 76 total patterns: 38 rejected and 38 accepted

Using these 10 initial sets we will create 10 different data groups. The training data groups that will be used as inputs to the networks for training the system and evaluating the evolution of the error will consist in the union of 9 of the previous 10 sets. The final evaluation that calculates the performance of the performance of the network will use the other initial set. The 10 folders will be created with the variation of the initial set that is used for evaluation and
not for training. The networks are trained from the same initial aleatory weights presenting the data corresponding to each of the 10 final folders created. Finally the mean values of the results will be calculated to eliminate the statistical influence of the fixed selection of some patterns for training and evaluation.

C. Koniocortex-Like Network Model

KLN model is composed by rate code neurons whose outputs $O_j$ are limited between the values 0 and 1. These values represent the probability of occurrence of an action potential. Considering the normalized input pattern $\vec{I} = \frac{I}{\|\vec{I}\|}$ (lower case notation meaning vector normalization) as the external inputs to neuron j. Normalization is performed with the $l_1$-norm in which:

$$\|\vec{I}\| = \sum_{i=1}^{n} |I_i|$$  \hspace{1cm} (1)

And the neuron’s j weights as the components of a vector prototype $\vec{T}^j$, so that $\vec{T}^j = [W_{j1}, W_{j2}, ..., W_{jn}]$. The inner product of weights and the input pattern, the net-input of neuron j is calculated as $\text{net}_j = \|\vec{W}^j \cdot \vec{I}\| = \|\vec{T}^j \|$, the modulus of the projection of prototype $\vec{T}^j$ over input pattern $\vec{I}$.

The weights are modified as a consequence of the training method using the incremental version of the pre-synaptic rule:

$$\Delta \omega = \xi f(O - \omega)$$  \hspace{1cm} (2)

Where $O$ and $I$ are the post-synaptic and pre-synaptic action potential probabilities, respectively, and $\xi$, a learning factor.

This rule is based on the plasticity curves [14] that relate the post-synaptic voltage and the weights. This curve is also influence by metaplasticity, an homeostatic neural property that changes the elongation of the plasticity curves depending on the initial synaptic weights [15], [16].

There exists a relation between the net-input of neuron $O^j$ and its firing probability $O_j$, using a conventional sigmoidal activation function.

$$O_j = \frac{1}{1 + e^{-\frac{\text{net}_j}{k}(0.5 - 2s^j)}}$$  \hspace{1cm} (3)

Where $k$ is a curve-compressing factor and $s^j$ the horizontal shift of the activation function ranging from zero to one, $0 < s^j < 1$. Real neuron exhibits intrinsic plasticity [17], [18] as shown in Fig. 1, the homeostatic property that makes very active neurons to be moderated and inactive neurons to increment its firing rate. The activation function shifts leftwards or rightwards regulating the activation of scarcely or highly activated neurons, respectively.

Parameter $s^j$ is mathematically incorporated to the model in the neuron’s activation function $f(\cdot)$ relating the net-input of the neuron to its spiking probability $O_j$:

$$O_j = f(\|\vec{T}^j\|, s^j)$$  \hspace{1cm} (4)

The following equation calculates the shift of the activation function, $s$ at time $t$ in terms of the shift and output probability of the neuron at time $t - 1$.

$$s_t^j = \frac{v_t O_{t-1} + s_{t-1}^j}{v + 1}$$  \hspace{1cm} (5)

Where $v$ is the shifting velocity parameter. It is a small arbitrary factor for adjusting the shifting rate of the activation function.

Fig. 2 is the complete version of the KLN model used in the experiment. In the KLN structure, B labeled neurons are inhibitory neurons including intrinsic plasticity capabilities. S labeled neurons are the main neurons engaged in competition and also present intrinsic plasticity. Since each S contacts a single B, intrinsic plasticity is regulated in both types of neurons. So if S is highly activated it is the same for associated B. This implies that S reduces its excitability and B the inhibitory field surrounding S, affecting the final activated neuron in future classification performances. TC neurons can use intrinsic plasticity to remove the mean of a series of input values. When removing the average value, patterns become more uncorrelated and easier to classify.

Fig. 2 shows that each S neuron has a recurrent connection on itself that was initially intended for allowing a sustained activation over time in simple rate-code neurons. Recurrent connections are extremely rare in real neurons. Despite of this, this kind of recurrent connection was indeed present in the koniocortex. Finally SB neuron is incorporated to the model to be used in pattern normalization. Similarly to real shunting/dividing inter-neurons, SB neurons perform the arithmetical summation of its inputs ($TC$ outputs), dividing the activation of its target neurons (the S neurons) by this quantity.
D. Network Characteristics

In this experiment 50 different initial networks have been trained, all the models used in this study were trained and tested with the same data and validated using the 10-folder cross-validation.

Considering the concrete experiment characteristics:

1) Structure of the network:
   - Number of inputs: This KLN has 14 neurons in its input layer corresponding to the number of elements that form each input pattern.
   - Number of TC neurons: There are 14 neurons in the TC layer (similar number as input layer to be coherent with the KLN structure).
   - Number of S neurons: There are 2 neurons in the S layer as two classes are considered in the experiment.
   - Number of B neurons: There are 2 neurons in the upper B layer associated to the S output neurons.
   - Activation function: Sigmoidal according to (3) and (5).

2) Conditions considered to finalize the network training:
   - Reach a defined number of epochs in the network training, the number of epochs will vary during the experiment.

When an input is presented to the network and the processing is complete, one of the output neurons will present a higher activation value than the other, which output will be inhibited by the winning node. The classification of the pattern will be made considering the comparison between the output values, and the performance of the network will be evaluated in comparison with the ideal output of the network.

This KLN prototype has demonstrated to be extremely sensitive to the concrete values of the training mathematical parameters, so very small changes from the used values cause big differences in the results, even in some cases lead to non-convergence of the learning algorithm. In this case the values obtained for the parameters have been obtained using a Montecarlo approach with many simulations until adequate results have been obtained. The final values used in this simulation are \( \nu = 0.025, \xi = 0.001 \), compression factor for the curve \( k = 20.13 \), the initial sigmoid shift \( s = 0.5 \), initial weights from TC to S neurons are negligible and random, and non-modifiable weights are set to \( W_{Ss} = 0.85, W_{Sn} = 0.98, W_{ITC} = 1.0 \) and \( W_{Bs} = 0.5 \).

E. Evaluation Method

50 different networks have been trained using the 10 fold cross validation method. Using 50 different initial networks and calculating mean values we assure that the results are independent of the initial random values in the creation of the networks. From the results obtained for the same network with each one of the folders the mean confusion matrix is obtained for each network. Once we have these 50 mean values an additional calculation is made and the final mean value is obtained as the final result of the experiment.

The following hypothesis are defined to build a confusion matrix model (as presented in table I):

- True Positive (TP) \( H(1/1) \): The pattern is accepted and has been classified as accepted.
- False Positive (FP) \( H(1/0) \): The pattern is rejected and has been classified as accepted.
- False Negative (FN) \( H(0/1) \): The pattern is accepted and has been classified as rejected.
- True Negative (TN) \( H(0/0) \): The pattern is rejected and has been classified as rejected.

TABLE I
CONFUSION MATRIX MODEL

| True Positive \( H(1/1) \) | False Positive \( H(1/0) \) |
| True Negative \( H(0/0) \) | False Negative \( H(0/1) \) |

To evaluate the performance two measures are used and defined as: \( \text{Sensitivity (SE)} = \frac{TP}{TP+FN} \text{ (%)} \) that evaluates the performance of the network identifying the accepted patterns, and \( \text{Accuracy (AC)} = \frac{TP+TN}{TP+TN+FP+FN} \text{ (%)} \) that evaluates the performance of the network classifying both kinds of patterns. TP, TN, FP, and FN stand for true positive, true negative, false positive and false negative, respectively.

We have considered sensitivity as the driver figure in these experiments.

IV. RESULTS

A. ACAD Patterns Classification

Several variations of the experiment have been performed with different number of epochs in each experiment (considering one epoch like presenting the full set of input patterns once to the network). The output of the network is integrated by two neurons, depending on which one presents the higher level at the output we have considered that one of the classes
in the worst situation so KLN is considered to perform successful classification task without the need of previous information about the ideal result of the each pattern of the data set.

C. State of the Art Study

This section present a comparison of the results obtained with the KLN structure with the results obtained by other authors using different methods over the same ACAD dataset. Table IV includes the name of the researchers, the algorithm or method used in the study and the values for sensitivity (if available) and accuracy obtained.

<table>
<thead>
<tr>
<th>Author (Year)</th>
<th>Method</th>
<th>Sensitivity</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>This study</td>
<td>KLN</td>
<td>87.66%</td>
<td>83.10%</td>
</tr>
<tr>
<td>West (2005)</td>
<td>MOIE</td>
<td>86.70%</td>
<td>86.68%</td>
</tr>
<tr>
<td>Ong (2005)</td>
<td>GP</td>
<td>-</td>
<td>88.27%</td>
</tr>
<tr>
<td>Huang (2005)</td>
<td>2SGP</td>
<td>-</td>
<td>89.17%</td>
</tr>
<tr>
<td>Martens (2007)</td>
<td>SVM</td>
<td>-</td>
<td>85.70%</td>
</tr>
<tr>
<td>Hoffmann (2007)</td>
<td>Bayes</td>
<td>-</td>
<td>86.70%</td>
</tr>
<tr>
<td>Huang (2007)</td>
<td>GA-SVM</td>
<td>-</td>
<td>86.90%</td>
</tr>
<tr>
<td>Peng (2008)</td>
<td>MCQP</td>
<td>87.00%</td>
<td>86.58%</td>
</tr>
<tr>
<td>Tsai (2008)</td>
<td>Multi-Classifiers</td>
<td>-</td>
<td>87.25%</td>
</tr>
<tr>
<td>Nanni (2009)</td>
<td>LCRPSC</td>
<td>-</td>
<td>87.44%</td>
</tr>
<tr>
<td>Xu (2009)</td>
<td>HARA</td>
<td>-</td>
<td>89.28%</td>
</tr>
<tr>
<td>Liao (2009)</td>
<td>CLC</td>
<td>-</td>
<td>86.52%</td>
</tr>
<tr>
<td>Tsai (2009)</td>
<td>PCA</td>
<td>-</td>
<td>89.93%</td>
</tr>
<tr>
<td>Ping (2009)</td>
<td>RBF-SVM</td>
<td>-</td>
<td>87.52%</td>
</tr>
<tr>
<td>Chen (2009)</td>
<td>LDA+SVN</td>
<td>-</td>
<td>86.52%</td>
</tr>
<tr>
<td>Wang (2011)</td>
<td>Boosting</td>
<td>-</td>
<td>86.56%</td>
</tr>
<tr>
<td>Wang (2012)</td>
<td>RS-Bagging DT</td>
<td>-</td>
<td>88.17%</td>
</tr>
<tr>
<td>Vukovic (2012)</td>
<td>Pure CBRPFGA</td>
<td>-</td>
<td>88.12%</td>
</tr>
<tr>
<td>Tsai (2014)</td>
<td>SOM + MLP</td>
<td>-</td>
<td>91.61%</td>
</tr>
<tr>
<td>Semenkina (2014)</td>
<td>SeekCEA</td>
<td>-</td>
<td>90.22%</td>
</tr>
<tr>
<td>Zhao (2015)</td>
<td>MLP Scoring</td>
<td>-</td>
<td>87.00%</td>
</tr>
<tr>
<td>Liang (2015)</td>
<td>Naïve Bayes</td>
<td>-</td>
<td>86.09%</td>
</tr>
<tr>
<td>Xiao (2016)</td>
<td>ECSC</td>
<td>-</td>
<td>87.62%</td>
</tr>
<tr>
<td>Chen (2016)</td>
<td>PBIL-AIS</td>
<td>-</td>
<td>86.23%</td>
</tr>
</tbody>
</table>

V. Conclusions

In this work we have presented the application of the theoretical basis of the Koniocortex-Like Network to a real complex data set. It is observed that the results are very similar to the ones obtained in previous supervised training experiments. The figures obtained for accuracy and sensitivity are almost the same with independence of the number of iterations used in the unsupervised training, so it seems that the linear independent information present in the data is identified and processed with a very small number of iterations which allows to reduce the computational load of the method. KLN model is still in a non optimized prototyping phase, but even in this situation the system is capable of obtaining accurate results able to compete with many state-of-the-art ANN models. These model is based in different plasticity concepts that can be widely applied to different areas inside BI to improve the BD extraction and classification methods.

B. Discussion of the Results

- This experiment related to the use of KLN with real data shows that even the system is very sensitive to the concrete values of the network parameters it is able to obtain good classifications results from the data set.
- Even with a small amount of epochs the system is able to start learning and the results don't present big variations neither from the sensitivity nor for the accuracy values once the network has been able to obtain the linear independent information from the data set. Once that this situation is reached it is not possible to improve the results with a higher number of epochs in the training.
- Comparing the KLN unsupervised results, they are worse than the results of advanced supervised methods as those presented in section [19], but with differences under 5%