Label Dependent Evolutionary Feature Weighting for Remote Sensing Data

Daniel Mateos-García, Jorge García-Gutiérrez, and José C. Riquelme-Santos

Department of Computer Science,
Avda. Reina Mercedes S/N, 41012 Seville (Spain)
{mateosg,jgarcia,riquelme}@lsi.us.es
http://www.lsi.us.es

Abstract. Nearest neighbour (NN) is a very common classifier used to develop important remote sensing products like land use and land cover (LULC) maps. Evolutive computation has often been used to obtain feature weighting in order to improve the results of the NN. In this paper, a new algorithm based on evolutionary computation which has been called Label Dependent Feature Weighting (LDFW) is proposed. The LDFW method transforms the feature space assigning different weights to every feature depending on each class. This multilevel feature weighting algorithm is tested on remote sensing data from fusion of sensors (LIDAR and orthophotography). The results show an improvement on the NN and resemble the results obtained with a neural network which is the best classifier for the study area.

Keywords: remote sensing, feature weighting, evolutionary computation, label dependence.

1 Introduction

Remote sensing is a very important discipline for many tasks like resource management, environmental monitoring, disaster response, etc. Since long time ago, machine learning techniques have been used to improve remote sensing performance and applicability. In addition, the use of active sensors like LIDAR (Light Detection and Ranging) has recently spread to improve the classical remote sensing products [1] which were mainly based on images. This fact involves a data complexity increase and makes machine learning even more important in order to extract meaningful information from remote sensing data.

Remote sensing knowledge can be gathered in several products where land use and land cover (LULC) maps can be found as one of the most important. This product is based on a classification of the terrain by means of its own morphologic or functional characteristics and it is a main tool to develop policies to manage the natural environment. An automatic pixel classification which is generally supervised is usually the first step to extract LULC maps from remote sensing data. Several techniques from machine learning have been used to develop LULC maps with satisfactory results, e.g., k-NN [2], Naive Bayes [3], SVM [4], etc.

E.S. Corchado Rodriguez et al. (Eds.): HAIS 2010, Part II, LNAI 6077, pp. 272–279, 2010.

Although machine learning validity has widely been proved in the remote sensing context, more research is needed in order to fulfill the standard requirements of many products from remote sensing and specially for LULC map development [5]. In this way, some researchers [6] have started to exploit optimization techniques (genetic algorithms) on their approaches showing that a weighted execution produces an improvement on the results.

In addition, machine learning often applies evolutionary computation to search optimal weighting on both structural and functional aspects in order to improve the predictive models. From the standpoint of unsupervised learning, we can see some works that focus on the determination of weights for clustering algorithms. Generally, the considered model is the k-means algorithm and traditional evolutionary techniques [7] with differences in the fitness function, which can be distance-based or even based on information given by a combination of different algorithms. Additionally, there are basically three main areas of weighting application in supervised machine learning: support vector machines optimization, artificial neural networks (training and topology) and feature weighting. Thus, SVM kernel [8] or artificial neural networks [9] parameters can be optimized by means of genetic algorithms or genetic programming with good results. In this context, evolutionary algorithms are usually employed to find a set of weights for the feature space, allowing greater accuracy in the classification process [10]. A common individual encoding is a set of real values that represent the weights of each feature. The fitness is defined by the classification process itself. Therefore, the search process can be viewed as a global task in which the optimal weights are considered with respect to the features regardless of the label that each instance belongs.

In this work, a novel proposal of applying evolutionary algorithms to search optimal weights for each feature depending on the label is shown. Existing methods in the literature usually work in a global way, i.e., the same weights are applied to all features. In opposition, this work shows that the importance of each feature can depend on the class to predict. Thus, for the LULC maps development, the features provided by orthophotos may have more leverage to distinguish vegetation textures, and the features provided by LIDAR can discriminate better different structures like buildings and roads since they include height measures. To the best of our knowledge, the application of this multiple weighting level has not been exploited enough and it can improve the results when a classical classifier is applied on remote sensing data. In this way, a new evolutionary method based on distances and a double weighting level is described with three main objectives:

- Improve the general quality of a well-known machine learning technique like the k-NN classifier when it is applied on remote sensing data.
- Obtain new information about what features are more important to classify each class by means of the study of the resulted weights per label.
- Provide a new tool to develop high accuracy LULC maps from fusion of sensors (LIDAR and imagery).

The rest of the paper is organized as follows. Section 2 describes the general process to select the feature weighting, highlighting the most interesting features of the applied evolutionary algorithm. The results achieved are shown in Section 3. Finally, Section 4 shows a summary of the conclusions and the future lines of work.

2 Method

2.1 Data Description

The data for this study belongs to a geographical area in the north of Galizia (Spain) and it was obtained from the fusion of LIDAR and orthophotography information. LIDAR is an active sensor technology that measures properties of light (usually laser) to register distant targets. After a LIDAR flight, a cloud point database is available in which for every point, it is possible to find: spatial position(i.e., x, y and z coordinates), intensity of return, number of the return in a sequence (if a pulse caused multiple impacts), etc. This features and the RGB values in an orthophoto are used in this work to obtain statistics on which the instances for the model are based.

A Digital Elevation Model (DEM) is needed to correct the height of objects. In this case, a DEM was extracted from the LIDAR data to make the correction. The orthophoto is used to extract features from the visible spectrum band. It was taken from the same area with similar weather conditions at the time of the LIDAR flight acquisition.

From the original data set, 500 instances are classified manually to build the training set. Every instance from the training set has a total of 61 basic statistics (average, variance, minimum, maximum, standard deviation, etc.) from five different bands of the LIDAR and the image data: height, intensity, red band, green band and blue band; and 5 different classes, one for each land type: road, farming land, middle vegetation, high vegetation and buildings.

2.2 Preprocess

Before the generation of the model, a preprocess has to be carried out. Thus, three different filters are executed. First, every attribute missing value is replaced with the corresponding averaged value. Then, the data are standardized. Finally, a Correlation Feature Selection (CFS) method is applied in order to reduce the search space. With the 18 selected features already generated, the next phase is the execution of the evolutionary algorithm which is characterized in the next subsections.

2.3 Initial Population

The goal of the proposed evolutionary algorithm is to find an optimal set of weights in order to apply a lineal transformation to the feature space depending on each label and to improve the overall classification process. Thus, after the

evolutionary execution (see Fig. 1), a weight is obtained for each label and feature which is used to complete the classification process in two steps:

1. The weights are applied to the training instances according to its label.
2. Given a test instance, the transformed nearest neighbour label is chosen to classify the test instance.

To carry out this idea, the population representation is as follows: An individual is a matrix which represents the weights per label for every feature. Hence, there is a row for each label which has as many columns as features. The initial population is then a matrix of weights where every value is randomly chosen.

2.4 Fitness Function

As previously said, the training data consist of a matrix P with n rows (each one represents a pixel) and f columns (one per feature). A class label is assigned by using the *label* function to each point of P. For simplicity, we assume that the label is an integer between 1 and b. Thus, a point p_i is a row of P and a vector of $\mathbb{R}^f$ such that $label(p_i) = l \in \{1..b\}$.

A transformation is given by a matrix of weights W (w_{ij}), with b rows (number of different labels), and f columns (number of features). Thus, p_i is transformed through W in p'_i so that each feature is "weighted" with a value depending on the class to which it belongs as follows:

$$\forall j = 1...f,\ p'_{ij} = w_{label(p_i)j} * p_{ij} \tag{1}$$

As seen in Fig. 1, each training set P, is divided into n bags (3), so that the weights of the individual which is being evaluated are applied to $n-1$ bags (5), and the remaining is used as initial test (6 et seq.). The transformation of each label is applied to each pixel of the test bag (6-8) and then, the nearest pixel from P' is calculated (9). Once the point has been tested, it becomes part of P' reinforcing the training (10). The label that makes the process return the shorter distance is chosen (12). If this label does not match the point test label, the fitness will be increase (13, 14). Therefore, to calculate the fitness function, the input parameters are a matrix P, the *label* function and a matrix W. The output is a function to measure the classification error rate which is the objective function to be minimized.

2.5 Crossover and Mutation

The crossover operation for two individuals is applied to every corresponding row (i^{th} row of an individual is crossed with the i^{th} row of the other) since they have the same label. The roulette-wheel method is selected as the method to obtain the individuals to cross. Besides, two techniques have been selected for the generation of the new individuals: the uniform crossover and the BLX-α crossover [11]. The uniform crossover consists of the pick out of a gene from one parent at random. The BLX-α crossover is described as follows: if g_1 and g_2 are

W is the matrix $\begin{bmatrix} w_{11} & \cdots & w_{1f} \\ \vdots & \ddots & \vdots \\ w_{b1} & \cdots & w_{bf} \end{bmatrix}$

```
fitness=0
for i = 1 to m do
    We divide P into n bags: B_1, ..., B_n
    for all bag B_k do
        According to Equation 1, we apply the W transformation to every point from the
        remaining n − 1 bags, obtaining the set of points P'
        for all point p_i in B_k do
            for all label l ∈ {1..b} do
                We construct the tranformed point p_i^l so that p_ij^l = w_lj * p_ij
                We calculate d_l = minimum distance from p_i^l to the points of P'
                We apply the W transformation to p_i according to its label, and we add it
                to P'
            end for
            We calculate the minimum from the distances d_l. Let h ∈ {1..b}, the label of
            the point of P' which makes d_l.
            if the label of p_i ≠ h then
                fitness = fitness + 1
            end if
        end for
    end for
end for
```

Fig. 1. Fitness function

the i^{th} gene from each parent, the new gene is a real number randomly selected in the interval $[G_{min} - I\alpha, G_{max} + I\alpha]$, where:

α=positive real number,
$G_{max} = max(g_1, g_2)$,
$G_{min} = min(g_1, g_2)$,
$I = G_{max} - G_{min}$

The mutation operator has been defined to increase or decrease the value of a weight according to a probability p. The increase or decrease is a random value Δ that satisfies:

$\Delta = r/10^z$, $where:$
$r \in \mathbb{R} : (0 \leq r \leq 1)$ and
$z \in \mathbb{Z} : (0 \leq z \leq n)$

3 Results

To assess the quality of our approach, a comparison among several classifiers is carried out. The classifiers Naive Bayes, Support Vector Machines (obtained

Table 1. Averaged error rate for each studied algorithm

Algorithm	**Accuracy**
Naive Bayes	0.15
SMO	0.14
Nearest Neighbour	0.13
Neural Network	0.10
Nearest Neighbour LDFW	0.10

by Sequential Minimal Optimization), Artificial Neural Networks (Multilayer Perceptron) and Nearest Neighbour (NN) with and without LDFW are chosen to compare their performance. Every model is built using the WEKA software [12]. For the experiments, the LDFW evolutionary algorithm is set up with the following parameters: a population of 20 individuals, 100 generations, a 10% of elitism and a 20% of mutation probability.

To establish a fair comparison among the performances of the different algorithms, a *stratified n-fold cross-validation* method is used. Concretely, three 10-fold cross-validation with different random seeds are executed and the results for each fold are then registered. In Tab. 1, the overall error rate for each algorithm can be seen.

In order to evaluate the statistical significance of the measured differences in algorithm ranks, we use a method for comparing classifiers across multiple data sets. In this case, there is only one data set since remote sensing data has high costs to be obtained. Thus, the set of measures are the partial results of the previous 10-fold cross-validation (30 measures for each classifier) and the Friedman test is selected to analyze those measures. The Friedman test is a non-parametric statistical test which evaluates the differences among more than two related sample means. The null hypothesis is that every classifier performs the same, regardless the differences among the registered results. In Equation 2, the statistic used can be seen.

$$X_F^2 = \frac{12n}{k(k+1)}\left(\sum_j {r_j}^2 - \frac{k(k+1)^2}{4}\right) \tag{2}$$

The Friedman test checks whether the average ranks are significantly different from the mean rank $r = 2.5$ expected under the null hypothesis. Leaning on a statistical package (MATLAB), p value for the Friedman test has resulted on a value of $7.0351E-7$ so the null hypothesis is rejected and the measured average ranks are significantly different (at $\alpha = 0.05$).

With this in mind, the results show that the performance of the Nearest Neighbour with LDFW is very similar to the resulted from the neural network which is the best classifier for the study area. However, as will be seen later, the

Table 2. Most important features according to its weight for the study zone

Class	**Features**			
Road	MINSNDVI	PEC	IMAX	HCV
Farming Land	IMEAN	IGMEAN	HMAX	HCV
Middle Vegetation	HSTD	IGKURT	MINSNDVI	IGVAR
High Vegetation	IMAX	IRVAR	IGVAR	IGMEAN
Buildings	IGKURT	PCT32	EMP	MINSNDVI

H*: height statistic, I*: Intensity statistic, IG*: Intensity green band stat., IR*: Intensity red band stat., *SNDVI: Simulated Normalized Difference Vegetation Index stat., PEC: Penetration coef., PCT32: Percentage third or later returns over second returns.

LDFW technique provides a descriptive information about the most important features per class. Neural networks supply an approximation about the features importance too, but in a much less explicit manner. The rest of the classifiers show a lower accuracy. If the LDFW-NN is compared with the classical NN, it results in a 3% of improvement.

In Tab. 2, the importance of the feature weighting according to the label can be seen. After the application of the LDFW, every class has its own set of features that determines its label the best. This information provides a very important feature selection tool and allows us to establish a more accurate class separation; e.g., vegetation classes are principally determined by the orthophoto, specially by the features that correspond to the green band (IG features) whilst roads are characterized better by LIDAR features.

4 Conclusions

In this paper, a new algorithm based on evolutionary computation which was called Label Dependent Feature Weighting (LDFW) was proposed. The LDFW method transforms the feature space assigning different weights to every feature depending on each class. This multilevel feature weighting algorithm was tested on remote sensing data from fusion of sensors (LIDAR and orthophotography) in order to improve a NN which is a very used classifier in the context of the LULC map development. The results showed an improvement of the 3% on the NN and resemble the results obtained with a neural network which was the best classifier for the study area. Additionally, the LDFW was able to provide qualitative and quantitative information about the importance of each feature in order to distinguish among the different classes.

In future work, the use of other measures like entropy in lieu of distance will be a very interesting way to improve the results and should be taken into account. In addition, different transformation functions on the attributes which, at the

moment, are limited to linear kernels should be explored. Finally, the definition of this algorithm as an independent preprocess method is a primary objective so that more complex classifiers like ensembles could be tested.

References

1. Erdody, T., Moskal, L.: Fusion of LIDAR and imagery for estimating forest canopy fuels. Remote Sensing of Environment (to appear, 2010)
2. Atkinson, P.M.: Spatially weighted supervised classification for remote sensing. International Journal of Applied Earth Observation and Geoinformation 5(4), 277–291 (2004)
3. Bork, E.W., Su, J.G.: Integrating lidar data and multispectral imagery for enhanced classification of rangeland vegetation: A meta analysis. Remote Sensing of Environment 111(1), 11–24 (2007)
4. Mazzoni, D., Garay, M.J., Davies, R., Nelson, D.: An operational misr pixel classifier using support vector machines. Remote Sensing of Environment 107(1-2), 149–158 (2007)
5. Shao, G., Wu, J.: On the accuracy of landscape pattern analysis using remote sensing data. Landscape Ecology (23), 505–511 (2008)
6. Tomppo, E.O., Gagliano, C., Natale, F.D., Katila, M., McRoberts, R.E.: Predicting categorical forest variables using an improved k-nearest neighbour estimator and landsat imagery. Remote Sensing of Environment (113), 500–517 (2009)
7. Krishna, K., Narasimha Murty, M.: Genetic K-means algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 29(3), 433–439 (2002)
8. Howley, T., Madden, M.G.: The genetic kernel support vector machine: Description and evaluation. Artificial Intelligence Review 24, 379–395 (2005)
9. Hervás-Martínez, C., Martínez-Estudillo, F., Carbonero-Ruz, M.: Multilogistic regression by means of evolutionary product-unit neural networks. Neural Networks 21(7), 951–961 (2008)
10. Komosinski, M., Krawiec, K.: Evolutionary weighting of image features for diagnosing of CNS tumors. Artificial Intelligence in Medicine 19(1), 25–38 (2000)
11. Eshelman, L.J., Schaffer, J.D.: Real-coded genetic algorithms and interval-schemata. In: Whitley, D.L. (ed.) Foundation of Genetic Algorithms 2, pp. 187–202. Morgan Kaufmann, San Mateo (1993)
12. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: An update. SIGKDD Explorations 11(1) (2009)