CLASSIFICATION OF REMOTE SENSING DATA USING K-NN METHOD

Miss. MAYANKA B. KHUMAN

M.E.E.C. Student, Department Of Electronics & Communication Engineering, Kalol Institute of Technology & Research centre, Kalol.

mbkhuman@gmail.com

ABSTRACT: *Remote sensing image processing is nowadays a mature research area. The techniques developed in the field allow many real-life applications with great societal value. For instance, urban monitoring, fire detection or flood prediction, deforestation and crop monitoring, weather prediction, land use mapping, land cover mapping can have a great impact on economical and environmental issues. From acquisition to the final product delivered to the user, a remotely sensed image goes through a series of image processing steps, Classification maps are probably the main product of remote sensing image processing. Among the various remote sensing methods that can be used to map areas, the K Nearest Neighbor (KNN) supervised classification method is becoming increasingly popular for creating forest inventories in some countries.*

Keywords — Remote sensing, Image processing, Supervised-Classification, KNN.

I: INTRODUCTION

Remote sensing is defined as the measurement of object properties on the Earth's surface using data acquired from aircraft and satellites. Sensors on board satellites or aircrafts measure the amounts of energy reflected from or emitted by the Earth surface targets in different wavelength intervals (Fig. 1.1). Remote sensing has proven useful for a range of applications including the detection of earthquakes, faulting, volcanic activity, landslides, flooding, wildfire, and the damages associated with each.

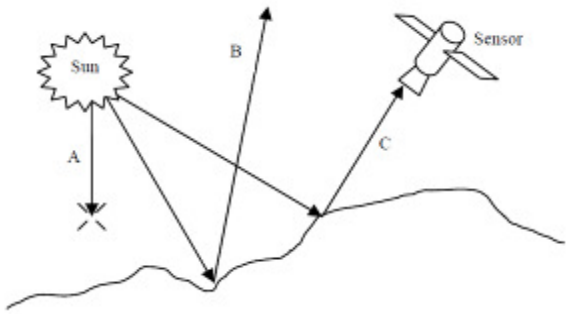

Figure 1 - Sensors measure the amounts of energy reflected from or emitted by the Earth surface targets: Absorbed (A), Scattered (B), and Reflected (C) energy.

A basic assumption in remote Sensing is that individual land covers (soil, water bodies, and vegetation of various species) have a characteristic manner of interacting with incident radiation which is described by the spectral response (spectral signature) of that target. The spectral response is represented by a curve that describes the amount of reflected energy by the target in function of the wavelength.

II: IMAGE PROCESSING FOR REMOTE SENSING DATA

For each remote sensing application a specific processing methodology must be developed. From acquisition to the final product delivered to the user, a remotely sensed image goes through a series of image processing steps, starting with efficient compression strategies and ending with accurate classification routines. In spite of the application complexity, some basic techniques are common in most of the remote sensing applications named as image registration, image fusion, image segmentation and classification

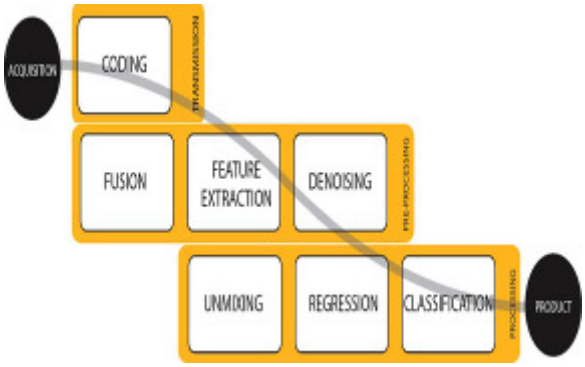

Figure 2 - Remote sensing image processing chain [3]

III: Classification of image

Classification maps are probably the main product of remote sensing image processing. Important applications are urban monitoring, catastrophe assessment, change or target detection. Image classification is the process used to produce thematic maps from remote sensing imagery. A thematic map represents the earth surface objects (soil, vegetation, roof, road, buildings), and its construction implies that the themes or categories selected for the map are distinguishable in the image. Many factors can difficult this task, including topography, shadowing,

atmospheric effect, similar spectral signature, and others. In order to facilitate the discrimination among classes, regions obtained in the segmentation process are described by attributes (spectral, geometric, texture) that attempt to describe the objects of interest in the image. Different methods used for image classification for remote sensing data are k-NN, LDA, neural nets and kernel methods[2].

Remote sensing research has developed many methods to classify digital images by the spectral properties of the objects present in the image. Broadly speaking, classification methods can be divided in three families.

1. ***Unsupervised*** *methods*
2. ***Supervised*** *methods*
3. ***Semi-supervised*** *methods*

In *unsupervised classification*, the computer is allowed to analyze all of the spectral signatures of all of the image's pixels and to determine their natural groupings, that is to say, to group the pixels on the basis of their similar spectral signatures. The main advantage of this method is its great speed, for it requires practically no intervention from the user. Its main flaw is to be based exclusively on spectral differences, which do not always correspond to natural land cover categories. For example, unsupervised classification often yields several classes corresponding to grassy vegetation but only one class encompassing the entire urban fabric, roadways, and tilled fields, which does not usually meet the interpreter's needs.

Supervised classification is usually appropriate when you want to identify relatively few classes, when you have selected training sites that can be verified with ground truth data, or when you can identify distinct, homogeneous regions that represent each class. In order to describe the classification procedures used in an intuitive manner, the supervised classification techniques are divided into two standard groups called non-parametric and parametric procedures. Supervised classification is the process of using samples of known identity to classify pixels of unknown identity. At present, this field is probably the most active in remote sensing image processing.

Finally, *semi-supervised methods* join the (typically few) labeled data and the information about the wealth of unlabeled samples. In remote sensing, the data manifold has been modeled with either graphs or cluster kernels algorithms.

IV: K-NN ALGORITHM

K Nearest Neighbour (KNN from now on) is one of those algorithms that are very simple to understand but works incredibly well in practice. KNN is a *non-parametric lazy learning* algorithm. When you say a technique is non-parametric, it means that it does not make any assumptions on the underlying data distribution. This is pretty useful, as in the real world, most of the practical data does not obey the typical theoretical assumptions made (eg gaussian mixtures, linearly separable etc) . Non parametric algorithms like KNN come to the rescue here.

It is also a lazy algorithm. What this means is that it does not use the training data points to do any *generalization*. In other words, there is *no explicit training phase* or it is very minimal. This means the training phase is pretty fast. Lack of generalization means that KNN keeps all the training data. More exactly, all the training data is needed during the testing phase.

The dichotomy is pretty obvious here – There is a non-existent or minimal training phase but a costly testing phase. The cost is in terms of both time and memory. More time might be needed as in the worst case, all data points might take point in decision. More memory is needed as we need to store all training data.

The KNN algorithm is a method for classifying objects based on the closest or most similar training samples in the feature space. It is a form of instance-based learning. An object is classified by a majority vote of its neighbors. This so-called nearest neighbor is determined by the use of distance functions. Eventually, the unknown object is assigned to the class most similar amongst its k nearest neighbours (figure 4.1).

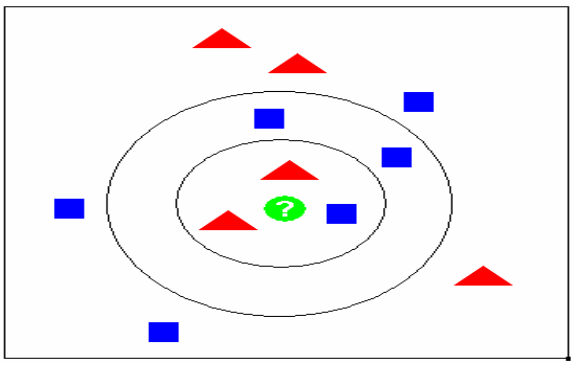

Figure 4.1 Example of kNN classification. The test sample (green circle) should be classified either to the first class of blue squares or to the second class of red triangles. If k = 3, it is classified to the second class, because there are 2 triangles and only 1 square inside the inner circle. If k = 5, it is classified to the first class (3 squares vs. 2 triangles inside the outer circle).

The nearest neighbor algorithm is one of the simplest machine learning algorithms.

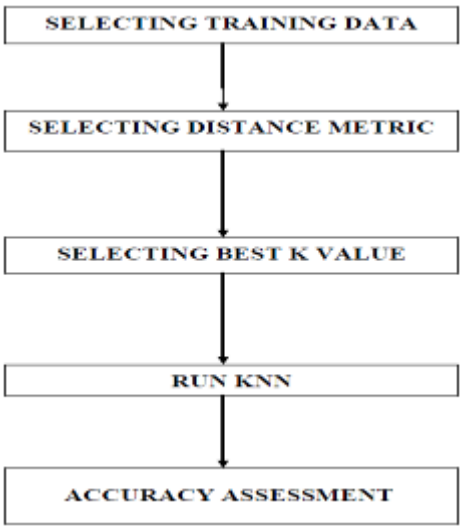

Figure 4.2 – KNN algorithm steps.

The nearest neighbor algorithm steps can be described as follows:

1) Training phase
   a) A human being classifies a number of objects manually. This is the training set. The feature vectors and class labels of these samples are stored.
   b) The computer reads in this set of objects. The correct classification for these objects is known.

2) Classification phase
   a) A new, unclassified input object (test sample) is classified by a majority vote of its neighbors:

   - The neighbors are taken from the training set.
   - Distances from the test sample object to all stored sample objects are calculated, and the k nearest neighbors of the object are selected. k is a small integer.
   - There are different ways to assign a particular class to the object. Usually, the most common class among these k neighbors is assigned to the object. In other words, an object is assigned to the class c if it is the most frequent class label among the k nearest training samples. If k = 1, then the class of the nearest neighbor is assigned to the object. This special case (k = 1) is called the "nearest neighbor" algorithm.

Normally, the training phase is executed once, and the classification phase is executed any number of times afterwards.

V: DIFFERENT PARAMETER FOR KNN

There are different parameters from which we can make change in classification of data n improve efficiency.

Choosing k value

If there are only two different classes, an even number of k can cause a tie. Choosing an odd value for k prevents this problem. The size of k value is important. Small and large values of k value have different characteristics. The small and large k values are compared in Table 5. Heuristic techniques such as cross-validation can help in selecting a good value for k. ultimately, of course, the best value of k depends on the data at hand.

Table 5. Features of small and large values of k

Small values of k	Large values of k
Cause over-fit	Cause over-generalization
Increase negative effect of noise	Reduce negative effect of noise
Create distinct class boundaries	Create indistinct class boundaries

Distance functions

Different distance metrics can be used when calculating distances for the KNN algorithm. First of all, it is helpful to explain a general class of metric called as Minkowski metric which is given in Equation 5.1.

$$d(\vec{x},\vec{y}) = \left(\sum_{i=1}^{p} |x_i - y_i|^k \right)^{1/k} \quad (5.1)$$

where different values of $k \geq 1$ result in different commonly-used metrics. Here are the most common metrics used for calculating distances in KNN:

Euclidean: This is a special case of the Minkowski metric (Equation 5.2) where k = 2, This metric should be used when the different features are not strongly correlated.

$$d(\vec{x},\vec{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \ldots + (x_p - y_p)^2} \quad (5.2)$$

Mahalanobis: Let the vectors *x* and *y* be two input samples of the same distribution with the covariance matrix P. The Mahalanobis distance between sample *x* and sample *y* is defined as

$$d(\vec{x},\vec{y}) = \sqrt{(\vec{x} - \vec{y})^T \Sigma^{-1} (\vec{x} - \vec{y})} \quad (5.3)$$

This metric should be used when the different features are strongly correlated. The covariance matrix $\sum$ represents this correlation.

Diagonal (Class-Dependent) Mahalanobis: The Diagonal Mahalanobis distance between sample *x* and sample *y* is defined as in Equation 5.4

$$d(\vec{x},\vec{y}) = \sqrt{\sum_{i=1}^{p} \frac{(x_i - y_i)^2}{\sigma_i^2}} \quad (5.4)$$

As with the Euclidean distance, the correlation of different features is not taken into account here.

Manhattan: This is a special case of the Minkowski metric (Equation 5.1) where k = 1. Let the vectors *x* and y be two input samples (objects) with *p* features (x_1, x_2,…., x_p). The Manhattan distance between sample *x* and sample *y* is defined as

$$d(\vec{x},\vec{y}) = |x_1 - y_1| + |x_2 - y_2| + \ldots + |x_p - y_p| \quad (5.5)$$

As with the Euclidean distance, the correlation of different features is not taken into account here.

Weight functions

After the k nearest neighbors of a test sample is found, these can be evaluated using different weighting methods. For each neighboring pixel, the pixel's weight is added to the total weight of that pixel's class. At the end, the class with the largest total weight wins. The goal of weight functions is to cause distant neighbors to have less effect on the majority vote than the closer neighbors. Here are the most common weight functions:

i. None: All neighbors have equal weight.

ii. Fraction: Let i be the order of the neighbor in the list of k neighbors, i = 1..k. The weight function is 1/i. Therefore, the weight of the pixel is inversely proportional to its rank in the neighbor list. The fraction weights decrease steeply as the order (i) of nearest neighbor increases (Figure 5.1).

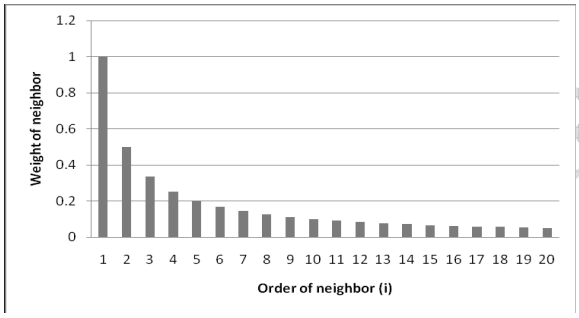

Figure 5.1. The Fraction weights for Weighted KNN

iii. Stairs: Let i be the order of the neighbor in the list of k neighbors, i = 1..k. The weight function is (k – i + 1) / k. Again, the weight of the pixel is inversely proportional to its rank in the neighbor list. The stairs weights slightly decrease as the order (i) of nearest neighbor increases (Figure 5.2).

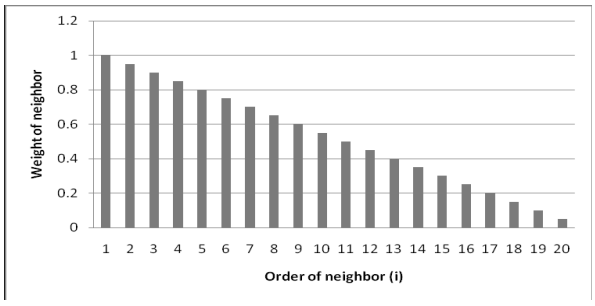

Figure 5.2. The Stairs weights for Weighted KNN

iv. Inverse Distance: Let d be the distance of the neighbor from the test sample. The weight function is 1/d. Therefore, the weight of the pixel is inversely proportional to its distance from the test sample.

v. Inverse Square Distance: Let d be the distance of the neighbor from the test sample. The weight function is $1/d^2$. Again, the weight of the pixel is inversely proportional to its distance from the test sample.

VI: COMPARISON OF DIFFERENT PARAMETER

Distance Functions

The distance function in knn algorithm can implement one of the following distance measurements:

- Euclidean,
- Manhattan,
- Diagonal mahalanobis,
- Mahalanobis,

by using leave one out cross validation, error values were calculated, for different values of k and for different distance metrics. The values were plotted in figure 6.1. the best result was obtained when the distance metric was selected as euclidean.

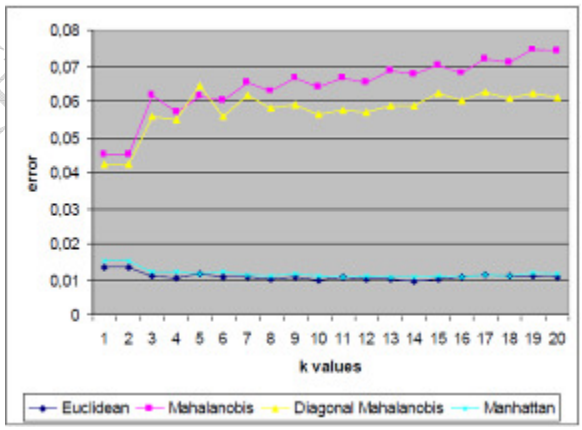

Figure 6.1. Effect of Different Distance Function.

Weight Functions

The weight function in knn algorithm can implement one of the following weight functions:

- None (Equal Weight),
- Fraction,
- Stairs,
- Inverse distance,
- Inverse square distance.

After the selection of the Euclidean distance metric as the best one, the different weight functions were taken into consideration. Again, Leave One Out crossvalidation was performed for each weight function and different values of k, and the error values were calculated. The best four weight functions were "Inverse Distance", "Inverse Square Distance", "Stairs" and "Fraction". The results can be seen in Figure 6.2.

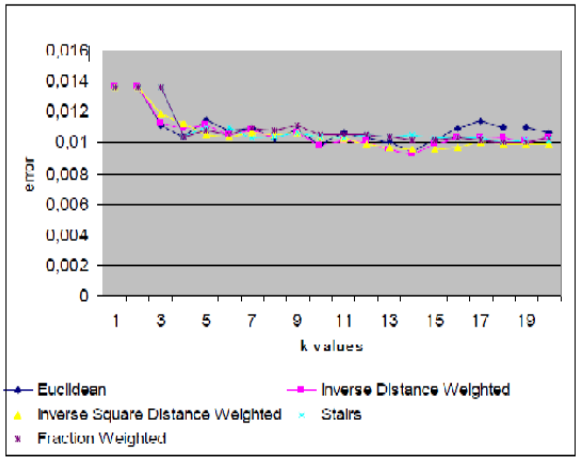

Figure 6.2. Effect of Different Weight Function.

VII: SUMMARY

The application of image classification algorithms to remote sensing data makes it possible to map land cover types of huge areas automatically. In the particular field of forest area detection, appropriate classifications methods help improve forest planning. A lot of research on the area of image classification focuses on improving classification accuracy and thus increasing its applicability for practical use. This study detailed the KNN algorithm in particular, which is commonly used in the detection of forest areas. KNN is fast, objective and transparent & also produces good results over larger areas. The main advantage of *k*-NN methods is their simplicity and lack of parametric assumptions Accuracies improve as areas are aggregated. Data intensive and training data must cover distribution of population. This method is cost and time saving as compared to other classification method.

REFRENCES

[1] "Digital Image Processing in Remote Sensing"- Leila M. G. Fonseca, Laércio M. Namikawa and Emiliano F. Castejon, IEEE 2009.

[2]"RECENT ADVANCES IN REMOTE SENSING IMAGE PROCESSING" -*Devis Tuia* University of Lausanne, Switzerland, -*Gustavo Camps-Valls* Universitat de Val`encia, Spain.

[3] "Introduction to the Issue on Advances in Remote Sensing Image Processing" -Gustavo Camps-valls, Jón Atli Benediktsson, Lorenzo Bruzzone, Jocelyn Chanussot, Ieee Journal Of Selected Topics In Signal Processing, Vol. 5, No. 3, June 2011.

[4] "Digital Image Classification – Remote Sensing" - Dr. James Campbell December 10, 2001.

[5] "A review of the status of satellite remote sensing and image processing techniques for mapping natural hazards and disasters" Karen E. Joyce, Stella E. Belliss, Sergey V. Samsonov, Stephen J. McNeill and Phil J. Glassey, 2009.

[6] "K-Nearest Neighbors Algorithm: Prediction and Classification" - Prof. Thomas B. Fomby, Department of Economics, Southern Methodist University Dallas, TX 75275, February 2008.

[7] "Nearest Neighbor Classification" - Charles Elkan , elkan@cs.ucsd.edu January 11, 2011.

[8] "k-Nearest Neighbour Classifiers" - P´adraig Cunningham and Sarah Jane Delany, March 27, 2007.

[9] "Extensions of the k Nearest Neighbour Methods for Classification Problems" Zacharias Voulgaris and George D. Magoulas University of London,2007.

[10] "Supervised Classification of Remotely Sensed Imagery Using a Modified *k*-NN Technique" - Luis Samaniego, András Bárdossy, and Karsten Schulz, IEEE Transactions On Geoscience And Remote Sensing, VOL. 46, NO. 7, JULY 2008.