Remote Sensing Image Segmentation with Probabilistic Neural Networks

LIU Gang

ABSTRACT This paper focuses on the image segmentation with probabilistic neural networks (PNNs). Back propagation neural networks (BpNNs) and multi perceptron neural networks (MLPs) are also considered in this study. Especially, this paper investigates the implementation of PNNs in image segmentation and optimal processing of image segmentation with a PNN. The comparison between image segmentations with PNNs and with other neural networks is given. The experimental results show that PNNs can be successfully applied to image segmentation for good results.

KEYWORDS image segmentation; probabilistic neural network (PNN)

Introduction

Image segmentation is becoming a widely used image processing method in remote sensing. The purpose of image segmentation is to subdivide an image into different parts that are homogeneous to certain criteria. In such a way, these regions are supposed to correspond to relevant objects in the terrain. The relevance of objects depends on users' requirements.

A PNN is an essential classifier similar to other neural networks. Its final use is to examine unknowns and to decide to which class they belong. Compared to standard statistical classification algorithms, a PNN can deal typically with very complex distribution\(^1\). Compared to other neural networks, such as, feed-forward networks, PNN has good mathematical credentials. In many applications, a PNN can provide mathematically sound confidence levels for decision-making. This has made a PNN suitable for large data processing. We can completely analyze and determine the output because PNNs give the possibility of output.

Researchers have studied about multi-source classification of complex rural areas by statistical and neural network approaches\(^2\) and neural networks for pattern recognition\(^2\). A new algorithm based on threshold estimation using the histogram\(^4\) was proposed for the segmentation of SAR images.

Although previous work has been successful in image segmentation, they have no advantage in acceptable accuracy, network training time, robustness to weights, and retraining time. This study is concentrated on the extraction of geometric and thematic information from remote sensing image data, implementation of PNN, application of modified PNN to segmentation of an image from the Baiyang Lake area and image processing fusion.

1 Methodology

The methodology in this paper is illustrated in Fig. 1.

The objective of this study is to apply a modified probabilistic neural network (PNN) to image segmentation and make a comparison of seg-
mentation with BpNNs and MLPs. In performance, Microsoft Visual C++ 6.0 program and some other image processing software such as ENVI and PCI are used to test and evaluate the experimental results.

![Diagram showing image segmentation process]

**Fig. 1 Overview of PNNs in image segmentation**

### 1.1 Description of PNN’s foundation

A probabilistic neural network is basically a classifier, whose formulation is based on the probability density estimation of the input signals. One of its uses is to examine unknowns and to decide to which class they belong. PNNs have been proven to be more efficient than back propagation neural networks in some specific applications.

A PNN is closely related to Bayes’ method of classification. It can thus be described as follows. Suppose that a collection of random samples from K populations exists, which are denoted by $k = 1, \ldots, K$, and that $x$ belongs to class $K$. Each of these samples is a vector $x = [x_1, \ldots, x_n]$. Then the values of $h_k$ are defined as prior probabilities, and thus differentiates between different populations. We define costs, $c_j$, as misclassification that actually belongs to population $k$. In most applications, the costs are treated as being equal because we often ignore these quantities so as to simplify the computation.

The complete collection of samples from known populations is called the training set. It contains $n_k$ samples from population 1 until $n_k$ samples from population $K$. We need to develop from this training set an algorithm that enables us to determine the population from which an unknown sample is taken. If we can find an algorithm of which the expected misclassification cost does not exceed the costs of any other algorithm based on the same training set, then that algorithm is called Bayes optimal, Eq. (1).

Bayes equation proved that if we happen to know the true probability density functions $f_i(x)$ for all populations, then there is an equation called Bayes optimal decision rule. What we do is classifying an unknown sample $x$ into population $i$ if

$$h_{c_i} f_i(x) > h_{c_j} f_j(x)$$

for all population $j$ not equal to $i$.

Supposing Eq. (1) is true, then we can classify an unknown sample $x$ into population $i$. In fact, Bayes optimal rule tells us to define a class if it has high density in the neighbourhood of the unknown. The Bayes optimal rule Eq. (1) tends to determine the class with high prior probability, high cost of misclassification and high density in the neighbours of the labeled classes.

Observe in Eq. (1) that an estimated density appears on both sides of the expression. If, rather than working with densities, we work with constant multiples of densities, and if the constants are the same for all classes, the results obtained would be identical to those obtained by estimating true densities. In other words, for classification purposes, we can remove the restriction on weight function mentioned later. This makes us get a simple weight function.

A PNN is actually based on the probability density estimation of the inputs. The biggest problem with Eq. neuron(1) for Bayes’ classification is that we usually do not know the probability density functions $f_i(x)$. To solve this problem, Parzen proposed a method of density estimation. It is a good method for estimating a univariate probability density function from a random sample. It is a set of bell-shaped kernel functions centered at each sample point. This estimator converges asymptotically to the true
Parzen's probabilistic density function estimator uses a weight function, \( W(x) \), called a potential function, and frequently called a kernel, which has its largest value at \( d = 0 \) and which decreases rapidly as the absolute value of \( d \) increases. The density distribution is reconstructed by aggregating the information contributed by each window. If we collect a sample of size \( n \) from a single population, the estimated density function for that population is

\[
g(x) = \frac{1}{n\sigma} \sum_{i=1}^{n} W\left( \frac{x - x_i}{\sigma} \right)
\]

(2)

where \( g(x) \) is the estimated density function; \( n \) is a sample size from a single population; \( \sigma \) is a scaling parameter. Usually, PNN uses identical scale factors \( \sigma \) for all classes. This makes \( \sigma \) be a constant for all classes in all densities. For classification purposes, it can be ignored along with the density normalizing factor.

And \( W \) is a weight function, related to each class and scaling parameter \( \sigma \). The weight function must be properly normalized if the estimate is going to be a density function, rather than a constant multiple of a density function.

In Eq. (2), \( n \) indicates the size of a sample; \( \sigma \) indicates the scaling parameter, defining the width of the area of influence and should decrease while the sample size increases. The choice of a value for \( \sigma \) strongly influences the performance of the PNN.

In general, the weighting function \( W \) adapts the Gaussian function

\[
g(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}}
\]

(3)

The Gaussian function well behaves and is easily computed. Compared to Eq. (2), it is easy to understand and usually adopted in a lot of practical works.

Eqs. (1), (2) and (3) give a clear route to a PNN's foundation although they are not adopted by PNN. The foundation of the original PNN is shown in Eq. (4), and it is also about density estimator. In this work, Specht's density estimator function is used to image segmentation.

\[
g(x) = \frac{1}{(2\pi)^{d/2} \sigma^d} \sum_{i=1}^{n} e^{-\frac{(x - x_i)^2}{2\sigma^2}}
\]

(4)

Eq. (4) is proved to be a long history of good performance, and it is also proved that its training speed is fast.

Simply, PNNs may be considered as a kernel discriminate analysis. Learning is actually searching the optimal parameters for the kernel functions.

As a classifier, a PNN has come into widespread uses. From Eq. (4), it is obvious that the foundation of a PNN is based and synthesized by the Gaussian function and Specht function. Having understood this, we can have a look at the architecture of PNNs.

### 1.2 Segmentation with a PNN

With reference to part of PNN computation algorithms provided by in Reference [1], it is necessary to analyze components of remote sensing image data, the acceptable input data, the meanings and the structure of its output data as a result in Masters’ implementation. A developed program for PNN should be built to use the basic program and can be applied to image segmentation specifically. This would be successful after the transfer of original data.

The constructed network consists of four layers: an input layer, a pattern layer, a summation layer, and an output layer. The neurons in the input layer distribute the inputs to the pattern units. A distance measure between the input and the training case is calculated for each pattern neuron, which are represented by that neuron. The pattern layer for the network contains a hidden unit at every training case. Each unit is called a kernel and is often associated with a probability density function. The summation layer has one neuron for each class. Each summation neural that is delicate to a single class sums the pattern layer neurons corresponding to numbers of that summation neuron’s class. The output neuron is a threshold discriminator that decides which of its input from the summation units is the maximum. Usually the segmentation based on PNN may obtain the results with a high output probability.

Before training has commenced, training sets
should be in place. The training set is chosen by sampling the region of interest in an image. In procedure of training and learning, the probabilities of each pixel to which category they belong will be computed. The biggest probability determines the category of the pixel. This step is to be done in summation layer of a PNN, the biggest probability is assigned to 1, and other probabilities are assigned to 0.

The greatest biggest advantage of PNNs is the fact that the output is probabilistic, which makes the interpretation of output easy. Also the training speed of a PNN is obvious fast in performance.

The Microsoft Visual 6.0 C++ code for the PNN algorithm and for the implementation of segmentation were developed in this study, and in the developed program an image can be directly input and then a segmented image will be output.

2 Case study

2.1 Description and selection of study data

The success of this work depends on the availability of high quality images, so that relevant objects are represented by a significant number of pixels. In this work, a multi-spectral Landsat 7 ETM+ image data sets of Baiyang Lake, Hebei province are selected to test and evaluate the processing and obtained results (Fig. 2). This data contains some ground truth points, which can be used to evaluate the accuracy of segmentation. Band 4 is usually used because the different spectral characteristics of Landsat 7 ETM+ bands lead to the representative uses of them, and the same image data has also been used to segmentation with a BpNN, and an MLP. The ground truth information was obtained by fieldwork, and it can be adopted to evaluate the overall accuracy of segmentation (Fig. 3).

![Fig. 2 A tested image of Band4 (ETM +), which contains 33 ground points](image)

2.2 Experimental results

Quantifying errors in map is based on various statistics derived from the error matrix (also called a contingency table or confusion matrix) concept, first expounded for remote sensing data in the 1970s. The aim of the error matrix is to estimate the mapping accuracy (i.e. the number of correctly mapped points) within an image or map. An error matrix is constructed from points sampled from the map. The reference (also called ground truth or verification) data is normally represented along the columns of the error matrix, and is compared with the classified (or
thematic) data represented along the rows of the error matrix.

The numbers of samples strongly influence the accuracy statements. A limitation of this study is that only 33 ground truth points are provided. Therefore, if we calculate overall accuracy of 5 classes, an evaluation error will occur, which causes the decrease of the evaluation accuracy.

The overall classification accuracy is calculated by the ratio of the sum of correctly classified pixels in all classes to the sum of the total pixels tested. In this case, the ground truth points are used as reference (Table 1, Table 2 and Table 3).

Table 1 Overall accuracy performed by a BpNN: 54.5%

| Ground Truth Class | Class | I | II | III | IV | V | Total |
|--------------------|-------|---|----|-----|----|---|-------|
|                    | I     | 12| 5  | 2   | 1  | 18|       |
| Classified class   | II    | 1 |    |     |    |   |       |
|                    | III   | 2 | 1  | 1   | 1  | 4 |       |
|                    | IV    |   |    |     | 1  |   |       |
|                    | V     | 5 | 1  |     | 3  | 9 |       |
| Total              |       | 19| 7  | 2   | 4  | 33|       |

Table 2 Overall accuracy performed by an MLP: 51.5%

| Ground Truth Class | Class | I | II | III | IV | V | Total |
|--------------------|-------|---|----|-----|----|---|-------|
|                    | I     | 12| 5  | 2   | 1  | 18|       |
| Classified class   | II    | 1 |    |     |    |   |       |
|                    | III   | 1 | 1  | 1   | 2  | 4 |       |
|                    | IV    |   |    |     | 1  | 2 |       |
|                    | V     | 5 | 1  |     | 3  | 10|       |
| Total              |       | 19| 7  | 2   | 4  | 33|       |

Table 3 Overall accuracy performed by a PNN: 63.6%

| Ground Truth Class | Class | I | II | III | IV | V | Total |
|--------------------|-------|---|----|-----|----|---|-------|
|                    | I     | 13| 5  | 2   | 1  | 19|       |
| Classified class   | II    | 1 |    |     | 1  | 2 |       |
|                    | III   | 1 | 2  | 3   | 3  | 6 |       |
|                    | IV    |   |    |     | 2  |   |       |
|                    | V     | 3 |    |     | 3  |   |       |
| Total              |       | 19| 7  | 2   | 4  | 33|       |

3 Discussion and conclusions

The segmentation of remote sensing image is an important step in image analysis, which directly affects the performance of the subsequent processing steps in image analysis. In this paper, image segmentation is applied at the pixel level by processing the remote sensing data in order to arrive at a meaningful partitioning of the image. The procedure of pixel-based segmentation and classification starts with the original data and end up with a segmented image.

This study demonstrates that neural networks can be considered as a good tool in image processing. Neural networks indeed expand our thoughts on image processing even though other classifier such as maximum likelihood classifier is popular.

The following are the outcomes of this study.

1. When BpNNs, MLPs, and PNNs are applied to image segmentation, they gain different results. Through the comparison between segmented results, by a PNN we can attain high segmentation accuracy; therefore, PNNs are superior in image segmentation as compared to BpNNs and MLPs methods if one band image is used.

2. If the image is segmented to two segments, the overall accuracy will increase.

3. Using Microsoft Visual C++ 6.0 to implement image segmentation by PNN algorithms is a good choice. The developed software can realize segmentation by PNN algorithms, and it is easy for users to make decision because the first output is a matrix, and this matrix indicates to which class each pixel can be assigned with high probability.

4. For decreasing incorrectly segmented pixels on the boundary, the adjustment of training set is a good solution.

5. Seldom would the probabilities belonging to different classes be equal because of uncertainty. Although the adjustment of training data may solve this problem, other uncertainty will still remain. For example, the differences of probabilities are little and then it is difficult to determine the results even though we can find the absolutely biggest probability.

6. Multi-spectral image can also be segmented using PNNs. The software developed in Microsoft Visual C++ 6.0 in this study is useful for the segmentation of a multi-spectral image.