Efficient Feature Selection and Nearest Neighbour Search for Hyperspectral Image Classification

Alan Woodley¹, Timothy Chappell², Shlomo Geva³, Richi Nayak⁴
Queensland University of Technology, Brisbane, Australia
and CRC for Spatial Information
a.woodley¹t.chappell²s.geva³r.nayak⁴@qut.edu.au

Abstract — Hyperspectral images typically contain hundreds of spectral bands, which is one to two orders of magnitude larger than the number of bands in multispectral images. This greater volume of spectral information could lead to discoveries that are not possible with multispectral images; however, overcoming the complexity of the additional information is a computational challenge. Here, we present a solution that uses feature selection, logarithmic nearest neighbor classification and neighborhood spatial analysis to classify the land use of multiple hyperspectral images. Empirical analysis shows that our solution is as accurate as other much more complex approaches and it is orders-of-magnitude more efficient. This ascertain that our solution is scalable to larger datasets while maintaining high accuracy.

Keywords—hyperspectral; remote sensing; feature selection; nearest neighbour; loonne, k-tree

I. INTRODUCTION

For decades, the analysis of remotely sensed images, captured via satellites, unmanned aerial vehicles or light aircrafts, has led to socio-environmental discoveries [1]. Most of this analysis has been performed on multispectral images, which represent the magnitude of the Sun’s reflectance as 6 to 14 spectral bands. An emerging area of research is the analysis of hyperspectral images, which contain hundreds of spectral bands. Hyperspectral images’ finer spectral resolution offers the potential for discoveries that are not possible with multispectral images but their complexity introduces additional challenges, particularly in terms of computational cost.

A common application of hyperspectral image analysis is land use classification. Many hyperspectral classifiers achieve high accuracy when tested on small datasets (~50,000 pixels). However, their algorithms have high computational complexity [2] meaning they would not scale to operational use, where a single image contains millions of pixels [1]. Our motivation is to develop a scalable solution. Our solution is scalable, using algorithms tested on datasets with billions of items [3], and accurate, outperforming almost all other approaches.

Our solution consists of three steps. First, we use a feature selection method, called LOONNE [4], to identify the optimal subspace of bands to use for analysis. Second, we classify land use types using the K-Tree [5] algorithm, which is able to perform a nearest neighbor search in logarithmic time. Third, we conduct spatial analysis using a neighbor-based voting scheme.

We begin by introducing hyperspectral remote sensing and describing some of the associated challenges. Then we analyze some of the previous approaches to analyzing remote sensing images. Next, we present the steps of our solution. Finally, we present the empirical performance of our solution across four standard datasets: Indian Pines, Salinas, Pavia University and Botswana [6] and show that our solution is very accurate and efficient.

II. BACKGROUND AND RELATED WORK

A. Hyperspectral Remote Sensing

Remote sensing is the observation of objects without physical interaction. A common example of remote sensing is via images, collected from instruments such as satellites, light aircraft or unmanned aerial vehicles. These instruments are examples of passive remote sensing tools since they capture information via a naturally occurring energy source (the Sun) rather than supplying their own energy source (such as Lidar). These instruments operate as follows:

1. During the day, sunlight shines on the Earth.
2. Objects absorb and reflect different spectral wavelengths at different magnitudes.
3. The instruments capture the magnitude of reflectance across spectral bands. Each band covers multiple wavelengths.

For a given object, the plot of its spectral bands is referred to as its spectral signature. Spectral signatures can be used to distinguish between objects. Historically, remote sensing instruments, such as NASA’s Landsat satellites, have captured multispectral images, which contain a small number of spectral bands (6-14). Multispectral images have been used for decades for a variety of environmental and social purposes. For example, distinguishing different types of land use, such as cropping land versus grazing land (fallow) with 90%-95% accuracy [1].

An emerging area of research is in the analysis of hyperspectral images, which contain a much finer spectral resolution than multispectral images, for example between 100-300 bands. The finer spectral resolution of hyperspectral images provides the potential for much deeper analysis. For example, using hyperspectral images it is possible to
distinguish different crop types from one another or estimate yield [7].

Despite the advantages of hyperspectral image analysis, it introduces complexities compared to multispectral image analysis:

1. A higher volume of data is required per pixel, requiring more computational resources (disk space, memory, bandwidth and processing time).
2. There has been less research using hyperspectral images, therefore, fewer hyperspectral signatures are known.
3. It is not always useful to use all the bands in every situation since some bands can introduce noise (a phenomenon referred to as the Hughes effect [8]).

A particular application of interest of hyperspectral image analysis has been in land use classification. A variety of approaches have been applied to land use classification. Most of the research falls into the following three steps: feature selection, classification and spatial analysis.

B. Feature Selection

Feature selection is a machine learning technique which optimizes the subspace of features for use in further analysis [9]. Feature selection is often employed for the following reasons:

1. To improve performance, by removing irrelevant or redundant features (noise); thereby, overcoming the Hughes effect;
2. To increase execution speed;
3. To increase generalization by reducing the risk of overfitting.

The easiest form of feature selection is manual feature selection [10], whereby, a domain expert identifies certain bands to be removed based upon the domain and the composition of a particular image. For example, removing bands used to identify water from a water-free image is known to improve performance [7]. However, manual feature selection has the following limitations:

1. It is an inefficient process, which does not scale, since every image would require a domain expert.
2. It can only be applied to objects whose spectral signature is already known.
3. It can only be applied in areas where the existence or non-existence of an object can easily be manually identified.

An alternative to manual feature selection is automatic feature selection, whereby, relevant features are selected algorithmically. There are three main categories of automatic feature selection algorithms:

1. Filter feature selection methods apply a statistical measure to assign a score to each feature [11, 12]. Often features are considered independently with regard to the dependent variable. Examples of filter feature section include mutual information [13] and principal component analysis (PCA) [14].
2. Wrapper methods consider feature selection as a search problem, where different combinations are compared to each other. Wrapper methods use a combination of features within a classifier and assign a score based upon the accuracy. Some common processes include a best-first search [15], random hill-climbing algorithm [16] or forward and backward passes [4].
3. Embedded methods learn which features best contribute to the accuracy of a classifier during its execution. Two common embedded paradigm are regularization [17] and boosting [18, 19].

Both wrapper and embedded methods depend on a classifier, whereas, filter methods are classifier independent. As a result, wrapper and embedded methods tend to have better performance than filter methods, but tend to be slower and have higher risk of overfitting.

Numerous feature selection algorithms have been applied to hyperspectral datasets. Tables I - IV present a summary of approaches from previous literature (as indicated in the reference column). Accuracy is determined by the number of correctly classified pixels (true positives) divided by the total number of pixels classified. The approaches were applied on the same standard datasets [6] used in our experiments and which are described in more detail in Section IV. However, there is no guarantee that the referenced works used the same test and training subsets. Therefore, only a representative comparison can be made between approaches. More extensive overviews of methods for feature selection in hyperspectral image analysis can be found in prior research [10, 20, 21].

TABLE I. REVIEW OF EXISTING FEATURE SELECTION ALGORITHMS—INDIAN PINES DATASET

<table>
<thead>
<tr>
<th>Feature Selection Algorithm</th>
<th>Type</th>
<th>Base Classifier</th>
<th>Accuracy (%)</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCM [22]</td>
<td>Filter</td>
<td>SVM</td>
<td>92.98</td>
<td>[23]</td>
</tr>
<tr>
<td>MMRM [24]</td>
<td>Filter</td>
<td>SVM</td>
<td>79.93</td>
<td>[23]</td>
</tr>
<tr>
<td>JMI [25]</td>
<td>Filter</td>
<td>SVM</td>
<td>80.93</td>
<td>[23]</td>
</tr>
<tr>
<td>RELIEF [26]</td>
<td>Filter</td>
<td>SVM</td>
<td>87.15</td>
<td>[23]</td>
</tr>
<tr>
<td>PCA [14]</td>
<td>Filter</td>
<td>SVM</td>
<td>60.23</td>
<td>[23]</td>
</tr>
<tr>
<td>PCA [14]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>88.60</td>
<td>[27]</td>
</tr>
<tr>
<td>ICA [28]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>88.36</td>
<td>[27]</td>
</tr>
<tr>
<td>MNF [29]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>86.59</td>
<td>[27]</td>
</tr>
<tr>
<td>LFDA [30]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>84.01</td>
<td>[27]</td>
</tr>
</tbody>
</table>
C. Classification

The aim of classifiers is to assign a label to a given object based upon its features. In the context of hyperspectral image analysis, classification is often used to assign a label to a particular pixel based upon its spectral values. Numerous classification algorithms have been used for hyperspectral analysis and here, we provide an overview of some of the more well-known and successful approaches.

Support vector machines (SVM) [31] are a widely used algorithm in image analysis, and are often the baseline approach for hyperspectral image analysis. SVMs map data in a training set to a hyperplane (Hilbert Space) that maximizes the margin between classes. SVMs tend to be very flexible classification algorithms that can model a wide range of decision boundaries. SVMs are a binary classifier, and so non-binary classification is usually achieved by combining several binary classifiers, often via one versus the rest and the one versus one paradigms [31].

Multimodal linear regression (MLR) [32] is a technique that models the posterior class distributions in a Bayesian framework. This allows these methods to select the maximum likelihood that a given object belongs to a particular class. In doing so, it adds greater theoretical justification to the degree of plausibility for such classes – other than just the boundary and distances between classes. MLRs are based upon the concept of binary linear regression adapted to multiple classes.

K-Nearest neighbor (kNN) is a search based classification method [33, 34]. For every unlabeled object, kNN methods calculate the distance between that object and set of labeled objects, with the closest k labeled objects selected.

The Iterative Self-Organizing Data Analysis Technique (ISODATA) uses an iterative classification algorithm. ISODATA [35] is similar to the k-means clustering algorithm but removes the need for the number of clusters to be specified a priori. Instead, clusters can be merged or split iteratively until a set of rules are met.

Decision trees [36] classify objects via a series of generalized rules. The rules are represented within a hierarchy (that is the tree structure). Objects start at the root of the tree and based upon their properties and rules traverse down a relevant branch until they reach the leaf, where they are assigned a class. Common examples of decision trees used to classify hyperspectral images include random forest [37, 38], boosted regression trees [18] and rotation forest [27].

Finally, ensemble classifiers combine together the results of multiple classifiers [39]. The idea behind ensemble classifiers is that the combination of many classifiers, even weak ones, will outperform any single classifier. Common examples of ensemble classifiers include AdaBoost [40] and bragging [41].

Tables V and VI provide a performance summary for some of above classifier types using standard hyperspectral datasets, again taken from previous studies (as indicated in the reference column). Again, not every study is directly comparable, due to using different test and training sets. More detailed reviews of classifiers used in hyperspectral image analysis can be found in prior research [27, 39, 42].

A key challenge for hyperspectral image analysis is scalability [42, 43]. While it is common for authors to report on the accuracy of their approaches, very few report on the efficiency of the approaches – even in terms of execution time. Therefore, it is difficult to determine the approaches scalability. However, estimations can be established via a theoretical analysis– for example we know that approaches such as SVMs, MLRs, boosted regression trees and random forests tend to have higher complexity than a simpler method such as a kNN search [4] – particularly if those other approaches require complex, and relatively inefficient, learning algorithms.

D. Spatial Analysis

The central idea of spatial analysis is to include the spatial values as well as other types of values (here spectral values) in classification. Broadly, it stems from Tobler’s first law of geography: “everything is related to everything else, but near things are more related than distant things”.

<table>
<thead>
<tr>
<th>Feature Selection Algorithm</th>
<th>Type</th>
<th>Base Classifier</th>
<th>Accuracy (%)</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCM [22]</td>
<td>Filter</td>
<td>SVM</td>
<td>70.12</td>
<td>[23]</td>
</tr>
<tr>
<td>MRMR [24]</td>
<td>Filter</td>
<td>SVM</td>
<td>42.92</td>
<td>[23]</td>
</tr>
<tr>
<td>JMI [25]</td>
<td>Filter</td>
<td>SVM</td>
<td>42.92</td>
<td>[23]</td>
</tr>
<tr>
<td>RELIEF [26]</td>
<td>Filter</td>
<td>SVM</td>
<td>54.81</td>
<td>[23]</td>
</tr>
<tr>
<td>PCA [14]</td>
<td>Filter</td>
<td>SVM</td>
<td>33.69</td>
<td>[23]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature Selection Algorithm</th>
<th>Type</th>
<th>Base Classifier</th>
<th>Accuracy (%)</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA [14]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>83.14</td>
<td>[27]</td>
</tr>
<tr>
<td>ICA [28]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>78.04</td>
<td>[27]</td>
</tr>
<tr>
<td>MNF [29]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>73.28</td>
<td>[27]</td>
</tr>
<tr>
<td>LFDA [30]</td>
<td>Filter</td>
<td>Rotation Forest</td>
<td>75.57</td>
<td>[27]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature Selection Algorithm</th>
<th>Type</th>
<th>Base Classifier</th>
<th>Accuracy (%)</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MCM [22]</td>
<td>Filter</td>
<td>SVM</td>
<td>97.27</td>
<td>[23]</td>
</tr>
<tr>
<td>MRMR [24]</td>
<td>Filter</td>
<td>SVM</td>
<td>93.97</td>
<td>[23]</td>
</tr>
<tr>
<td>JMI [25]</td>
<td>Filter</td>
<td>SVM</td>
<td>90.80</td>
<td>[23]</td>
</tr>
<tr>
<td>RELIEF [26]</td>
<td>Filter</td>
<td>SVM</td>
<td>83.18</td>
<td>[23]</td>
</tr>
<tr>
<td>PCA [14]</td>
<td>Filter</td>
<td>SVM</td>
<td>82.59</td>
<td>[23]</td>
</tr>
</tbody>
</table>
TABLE V. REVIEW OF EXISTING CLASSIFIERS–INDIAN PINES DATASET

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Type</th>
<th>Accuracy</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CART [44]</td>
<td>Regression</td>
<td>68.57</td>
<td>[27]</td>
</tr>
<tr>
<td>Bragging [41]</td>
<td>Ensemble</td>
<td>80.76</td>
<td>[27]</td>
</tr>
<tr>
<td>AdaBoost [40]</td>
<td>Ensemble</td>
<td>80.35</td>
<td>[27]</td>
</tr>
<tr>
<td>Random Forest [37]</td>
<td>Decision Tree</td>
<td>83.96</td>
<td>[27]</td>
</tr>
<tr>
<td>SVM [42]</td>
<td>SVM</td>
<td>87.06</td>
<td>[27]</td>
</tr>
<tr>
<td>LORSAL [45]</td>
<td>Regression</td>
<td>90.01</td>
<td>[27]</td>
</tr>
<tr>
<td>Rotation Forest [27]</td>
<td>Decision Tree</td>
<td>88.60</td>
<td>[27]</td>
</tr>
<tr>
<td>SMLR [46]</td>
<td>MLR</td>
<td>75.54</td>
<td>[46]</td>
</tr>
<tr>
<td>SVM [31]</td>
<td>SVM</td>
<td>78.80</td>
<td>[39]</td>
</tr>
<tr>
<td>Majority Voting [39]</td>
<td>Ensemble</td>
<td>90.80</td>
<td>[39]</td>
</tr>
<tr>
<td>k-NN [33]</td>
<td>kNN</td>
<td>87.90</td>
<td>[33]</td>
</tr>
<tr>
<td>Local SRNN (LSRNN) [33]</td>
<td>kNN</td>
<td>89.49</td>
<td>[33]</td>
</tr>
<tr>
<td>Spatially joint LSRNN (JLSRNN) [33]</td>
<td>kNN</td>
<td>89.49</td>
<td>[33]</td>
</tr>
<tr>
<td>Local mean-based NN (LMNN) [33]</td>
<td>kNN</td>
<td>92.80</td>
<td>[33]</td>
</tr>
</tbody>
</table>

TABLE VI. REVIEW OF EXISTING CLASSIFIERS–PAVIA UNIVERSITY DATASET

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Type</th>
<th>Accuracy</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>CART [44]</td>
<td>Regression</td>
<td>91.57</td>
<td>[27]</td>
</tr>
<tr>
<td>Bragging [41]</td>
<td>Ensemble</td>
<td>92.17</td>
<td>[27]</td>
</tr>
<tr>
<td>AdaBoost [40]</td>
<td>Ensemble</td>
<td>92.61</td>
<td>[27]</td>
</tr>
<tr>
<td>Random Forest [37]</td>
<td>Decision Tree</td>
<td>94.80</td>
<td>[27]</td>
</tr>
<tr>
<td>SVM [47]</td>
<td>SVM</td>
<td>95.10</td>
<td>[27]</td>
</tr>
<tr>
<td>LORSAL [39]</td>
<td>Regression</td>
<td>94.80</td>
<td>[27]</td>
</tr>
<tr>
<td>Rotation Forest [27]</td>
<td>Decision Tree</td>
<td>95.81</td>
<td>[27]</td>
</tr>
<tr>
<td>k-NN [33]</td>
<td>kNN</td>
<td>76.45</td>
<td>[33]</td>
</tr>
<tr>
<td>SRNN [33]</td>
<td>kNN</td>
<td>76.45</td>
<td>[33]</td>
</tr>
<tr>
<td>LSRNN [33]</td>
<td>kNN</td>
<td>75.75</td>
<td>[33]</td>
</tr>
<tr>
<td>JLSRNN [33]</td>
<td>kNN</td>
<td>75.75</td>
<td>[33]</td>
</tr>
<tr>
<td>LMNN [33]</td>
<td>kNN</td>
<td>86.33</td>
<td>[33]</td>
</tr>
</tbody>
</table>

In hyperspectral classification, it has been found that incorporating both spatial and spectral analysis is often superior to just using one type of analysis [48]. The simplest approach of incorporating spatial and spectral is to change values of a classified pixel based upon the values of a window of neighboring pixels [49]. For example, if a pixel has the class $x$ but the majority of surrounding pixels have class $y$, then there is a high probability that the pixel was misclassified, and it should be reclassified as class $y$.

Another common approach is to use segmentation, whereby, adjacent pixels with very similar values are grouped together and treated as a single object [50]. Segmentation is an established technique in image analysis and often displays better performance than analyzing individual pixels.

III. THE PROPOSED SOLUTION

A. LOONNE Feature Section

We used the Leave One Out Nearest Neighbor Error (LOONNE) [4] algorithm to find the optimal subspace of bands for classification. LOONNE is wrapper feature selection algorithm that has been successfully applied to a wide variety of datasets such as: insurance, breast cancer, credit screening, iris classification and image segmentation. LOONNE has either outperformed or performed comparably to other approaches such as linear machine learning and C5 boost. We utilized the LOONNE algorithm in the following way.

We start with all the bands in our feature space. For each item in the test set we search the training set to find its nearest neighbor using all values in the feature set and a distance metric (Euclidian distance). We compare the classes of the item and nearest neighbor and record if they are different. When every item in the test set has found its nearest neighbor we save the number of differences as the global subspace error.

We now create a series of candidate subspaces, each of which contain the previous feature space less one band. We repeat our near neighbor search experiment for each of the candidate subspaces. The candidate subspace that produces the smallest global subspace error becomes our new feature space.

We repeat the process of producing candidate subspaces and determine the one that produces the lowest subspace error, which becomes the updated feature space. We continue until all bands have been removed or until a success criterion is met (for example, no band can be removed that decreases the subspace error). The subspace with the lowest error becomes the optimal subspace.

The process of removing one band at a time to find the optimal subspace is backward sequential selection (BSS). An alternative approach is forward sequential selection (FSS), which starts with no bands and adds the band that results in the lowest error rate each time. While the two approaches have similar execution times if executed until all bands have been added/removed, forward sequential selection generally converges on the optimal result much faster and can therefore be useful when there is a limited computational budget. However, the process of adding bands rather than taking them away can mean that features that exhibit co-dependence with other bands may be handled incorrectly.

Efficiency is another advantage of LOONNE. Remember that classification needs to be performed for every candidate subspace. If a relatively inefficient classifier
such an SVM was used, then the classification would become computationally expensive. However, LOONNE uses a nearest neighbor search which has shown to be very efficient [4]. Furthermore, the cost of evaluating each subspace error is calculated in the initial classification pass, reducing the cost of performing an entire round of subspace evaluations from $O(n^d)$ to $O(nd)$, which can be significant when the dimensionality ($d$) is high.

B. K-Tree Nearest Neighbour Classification

Classification was performed by nearest neighbor search using the K-Tree data structure and clustering algorithm [3, 51-54]. Clustering is a common data science approach that assists with analysis of very large datasets. Here, we apply it to nearest neighbor search. In an exhaustive nearest neighbor search every unclassified item needs to be compared with every classified item. If there are $n$ unclassified items and $c$ classified items, then the complexity is $O(nc)$. However, if the classified items are clustered then the search can be split into two stages. First, the relevant cluster is identified, by comparing each unclassified item with a set of cluster representations, such as their means. Second, an exhaustive search is performed just on the items in the relevant cluster. If there are $c$ clusters, then the complexity becomes $O(nc + nm/c)$. This highlights the advantage of creating a large number of clusters – however, traditional approaches do not scale beyond a small number of clusters (at most 10,000) [52].

K-Tree is a hierarchical data structure and algorithm that stores clusters in leafs and representatives of those clusters in non-leafs (up to and including the root). The hierarchical structure of the K-Tree means that it can find a relevant cluster in logarithmic time. Therefore, the complexity of a K-Tree nearest neighbor search is $O(ln(nc + nm/c))$. As the size of $n$ becomes larger, the speed advantage of the K-Tree over an exhaustive nearest neighbor search increases. Fortunately, unlike traditional clustering algorithms such as $k$-means, the K-Tree was designed to handle very large datasets and has been able to cluster close to one billion objects into millions of clusters [3] – orders of magnitude larger sizes than traditional approaches.

We need to build our K-Tree before we can search it. We have built our K-Tree using the pixels in the training dataset using the following steps.

First, the user specifies the tree order – which is the maximum number of pixels in a cluster, or children in a non-leaf node. Initially, a single node is created. Pixels are streamed from input and added to the node. When the node becomes full (the number of pixels/objects exceeds the tree order), the node is split into two and the pixels are grouped into the two clusters using an existing algorithm ($k$-means) and distance metric (Euclidian). A third node is created to act as a parent to the other two and contains the means of each child as its value. This is called a promotion.

Pixels continue to be streamed into the K-Tree. The values of the root node’s children are compared with the values of the pixel. The child whose values have the smallest distance to the pixel’s values is chosen as the nearest neighbor. This process is repeated at every level of the tree until the appropriate cluster is identified. The pixel is then added to the cluster. If a node becomes full then a split and promotion occur. If the root becomes full then a split and promotion occur and the tree grows a level and becomes deeper. This process continues until all the pixels have been added to the K-Tree.

Once the K-Tree is built a nearest neighbor search can be performed. For each unclassified pixel the relevant branch of the tree is followed and the relevant cluster identified. An exhaustive nearest neighbor search is then performed on all the pixels in the cluster to identify the nearest neighbor and assign a label.

C. Neighbourhood Spatial Analysis

After classification, spatial analysis was performed to improve performance. Each pixel is compared to its eight immediate neighboring pixels. Eight pixels were chosen as our limit, since it provides an efficient solution while maintaining high accuracy. If the majority of neighborhood pixels have a different class to the main pixel, then its class is changed to match the neighborhood.

IV. RESULTS

A. Datasets and Computational Setting

Four hyperspectral datasets [6] have been used in our experiments. These datasets have been chosen because they: 1) represent a variety of land use types; 2) contain ground truth data; 3) are open access and 4) have been used in a large number of previous experiments. The datasets are:

- **Indian Pines**: An image captured by the AVIRIS sensor over the Indian Pines site in Indiana, United States. It contains 145 * 145 pixels, 224 bands and 16 classes consisting of mostly (two-thirds) crops, as well as forest, low density and transport infrastructure.

- **Salinas**: An image captured by the AVIRIS sensor over Salinas Valley in California, United States. It contains 512 * 217 pixels, 224 bands and contains 16 classes including crops, bare soils, and vineyard fields.

- **Pavia University**: This was an image taken by the ROSIS sensor over Pavia in northern Italy. It contains 610 * 610 pixels, 103 bands and 9 classes including bricks, roads and metal.

- **Botswana**: This is an image captured by the Hyperion sensor on the NASA EO-1 satellite, it contains 256 * 1480 pixels, 145 bands and 14 classes including swamps and woodlands.

The size of the training and test sets is detailed in Table VII. Note that ground truth was not available for every pixel. Pixels without ground truth were not included in either the training or test sets. This is consistent with previous studies and experiments that we compare with. All classification experiments were performed on a laptop computer with four 2.4 GHz i7-5500U processors and 16 GB of memory. All code was written and executed in Matlab version 2015b.
### B. Feature Selection

The LOONNE feature selection algorithm was executed on our training set using the prescribed method and backward sequential selection approach. Figure 1 shows the global subspace error of each LOONNE iteration for the Indian Pines dataset (from 220 bands to 1 band). The results show a high error at the extremities of the graph (when all or few bands are used), but a smaller error in between. While Indian Pines is the only dataset with full results presented, the other datasets followed the same trend. Table VIII displays the full and optimal number of bands for each dataset.

![Classification error in different feature subspaces](image)

**Fig. 1.** Classification error in different feature subspaces

### C. Classification and Feature Selection

The following nearest neighbor classification experiments were performed on each of the datasets:

- Exhaustive search with full set of bands;
- K-Tree search with full set of bands;
- Exhaustive search with LOONNE subspace of bands;
- K-Tree search with LOONNE subspace of bands;
- K-Tree search with full set of bands and spatial analysis;
- K-Tree search with LOONNE subspace of bands and spatial analysis;

In order to reduce the risk of overfitting we perform five cross-fold validation. We present the average and standard deviations of these validations in Tables IX and X. As metrics, we have used both accuracy, calculated as the number of correctly classified pixels in the test set divided by the number of the pixels in the test set, and the execution time of each experiment. From analyzing the results, we can make the following observations.

In terms of accuracy:

- Our results were comparable to – and mostly outperformed – the approaches presented in Tables V and VI. The complete solution (LOONNE + K-Tree + spatial analysis) outperformed all but one alternative [23]. Again, these are representative comparisons since we used different training and test sets to others (the specific splits were not previously published). However, by evaluating with five cross fold validation and achieving a small standard deviation it lowers the risk of observing significantly worse results if a different test and training set were used.
- LOONNE was effective as a feature selection algorithm, boosting the accuracy of all the datasets for both the exhaustive and K-Tree search by between 5% (exhaustive/Salinas) and 70% (K-Tree/Pavia University).
- The K-Tree preformed worse than a nearest neighbor search between 1% (exhaustive/Indian Pines) and 21% (exhaustive/Pavina University).
- The spatial analysis increased the accuracies in all scenarios by between 4% (KTree + Spatial/Salinas) to 40% (exhaustive/Pavia University).

In terms of efficiency:

- The K-Tree was one or two orders of magnitude faster than the exhaustive search. The K-Tree would outperform most of the other approaches discussed in Section II to a similar degree – since most of the other approaches would likely have at least similar if not worse complexity than an exhaustive nearest neighbor search.
- LOONNE also tended to increase efficiency, up to 300% (Indian Pines).
- The spatial analysis decreased efficiency by a small amount from less than 1% (KTree + Spatial/Salinas) to 5% (LOONNE + KTree + Spatial/Botswana).
- Overall, our solution is both very efficient and very scalable. Further experiments could explore how our solution and others deal with larger datasets.
V. CONCLUSION

Here, we explored methods to classify hyperspectral images. Our solution combined a feature selection algorithm (LOONNE), a logarithmic nearest neighbor classifier (K-Tree) and neighborhood spatial analysis. Our solution was orders of magnitude faster than the tested alternative approach yet maintained a very high accuracy – as good as or better than alternative and more complex approaches. Still, caution must be taken when analyzing the results, since the overall dataset sizes were small and different specific splits into training and test sets were used in experiments reported (and sometimes not reported) in different papers.

Further research should involve a deeper analysis on the methods described in this work, and in particular how well they compare with other methods using the same training and test datasets. In addition, experiments should be performed on larger datasets in order to evaluate the scalability of the different approaches. Previous work has shown that our approaches are scalable, so they should efficiently classify large hyperspectral data and maintain high accuracy.

ACKNOWLEDGEMENT

This work has been supported by the Cooperative Research Centre for Spatial Information, whose activities are funded by the Business Cooperative Research Centres Programme.

REFERENCES


