A comparison of machine learning methods for target recognition using ISAR imagery

Karen D. Uttecht\textsuperscript{a}, Cindy X. Chen\textsuperscript{a}, Jason C. Dickinson\textsuperscript{b}, Thomas M. Goyette\textsuperscript{b}, Robert H. Giles\textsuperscript{b}, William E. Nixon\textsuperscript{c}

\textsuperscript{a}Department of Computer Science, \textsuperscript{b}Submillimeter-Wave Technology Laboratory
University of Massachusetts Lowell, 1 University Ave, Lowell MA 01854;
\textsuperscript{c}US Army National Ground Intelligence Center, 2055 Boulders Road, Charlottesville, VA 22911;

ABSTRACT

The ability to accurately classify targets is critical to the performance of automated/assisted target recognition (ATR) algorithms. Supervised machine learning methods have been shown to be able to classify data in a variety of disciplines with a high level of accuracy. The performance of machine learning techniques in classifying ground targets in two-dimensional radar imagery were compared. Three machine learning models were compared to determine which model best classifies targets with the highest accuracy: decision tree, Bayes’, and support vector machine. X-band signature data acquired in scale-model compact ranges were used. ISAR images were compared using several techniques including two-dimensional cross-correlation and pixel by pixel comparison of the image against a reference image. The highly controlled nature of the collected imagery was ideally suited for the inter-comparison of the machine learning models. The resulting data from the image comparisons were used as the feature space for testing the accuracy of the three types of classifiers. Classifier accuracy was determined using N-fold cross-validation.

Keywords: machine learning, target recognition, decision tree, bayes, support vector machine, radar

1. INTRODUCTION

Over the past twenty-five years the U.S. Army National Ground Intelligence Center (NGIC) and the University of Massachusetts Lowell (UML), under the Expert Radar Signatures Solutions (ERADS) program, have developed state-of-the-art scale model measurement systems to acquire high resolution radar signature data in support of a number of advanced radar applications. These applications include automated/assisted target recognition (ATR), foliage penetration (FOPEN), and radar absorbing material (RAM) development. Compact ranges are now operating which model VHF/UHF, X-band, Ku-band, K-band, Ka-band, and W-band radar systems.

The Submillimeter-Wave Technology Laboratory (STL) specializes in using Compact Radar Ranges to collect radar measurements which are comparable to those collected in the field. Inverse Synthetic Aperture Radar (ISAR) images are frequently used to visualize and analyze radar data. ATR systems are used in situations where mistakes can be critical; it is important that the system performs flawlessly, by classifying a target into a specific type from an ISAR image of a target. ATR analysis and development has been an important part of the ERADS program. Previous work leveraged the highly accurate compact radar range data for ATR evaluation. \cite{1, 2} To that end, machine learning methods were investigated to determine which would be most effective.
Figure 1: Example ISAR images classified into target type.

Figure 1 shows an example of ISAR images for comparison. The collected data was used to compare machine learning methods to determine which are best suited for use in classifying ISAR images. The images are first processed using 8 different image comparison calculations against a reference image for each class, and the resulting 12 measurements are then used as the feature space for the three classifiers. Decision tree, Bayesian, and support vector machines have been compared.

2. MACHINE LEARNING METHODS OVERVIEW

Three machine learning methods are studied: decision tree, Bayes', and support vector machines. All three are supervised learning methods; data for which the class label is already known are used to build the classifier model, which is later used to classify future instances.

2.1 Decision tree

A decision tree is structured like a flowchart. An instance can be classified using the decision tree by starting at the top (root) node. Each non-leaf node has an attribute associated with it; the value of the attribute for the given instance determines which branch is followed, until a leaf node is reached. The value of the leaf node is the class label for the instance.

The J48 decision tree classifier was used. J48 is an implementation of the C4.5 algorithm developed by J. Ross Quinlan [3, 4]. The C4.5 algorithm builds a tree structure which can later be used to classify data by traversing the tree from the root to a leaf node. The decision tree is traversed based on the value of the attributes of the instance being classified, with the final leaf node reached determining the class.
As with most algorithms for decision tree induction, C4.5 follows a greedy recursive divide-and-conquer approach. At each level, the algorithm determines the best attribute to use as the root of the current sub-tree. The algorithm determines the best attribute by evaluating a criterion based on the formula for entropy on each possible split, choosing the attribute which has the highest gain in entropy if that attribute was used for the node. The criterion used by C4.5 is gain ratio, calculating the ratio of the gain in entropy for each possible attribute split. The algorithm then divides the data based on the chosen split and recursively calls itself on the subset of the data for each branch of tree, passing only those instances which meet the criteria. Once the algorithm reaches a point where the dataset being evaluated contains only instances with a single class or all attributes have already been used, the algorithm adds a leaf node with the majority class and return, resulting in a tree similar to the example in Figure 2. One major benefit of the decision tree method is the model is easy for humans to interpret and apply.

2.2 Naïve Bayes’

Bayesian classification is a statistical system based on Bayes’ theorem [5]. The probability that a given data instance belongs to each class is calculated based on statistical data gathered from the test data set. The data instance will be classified into the class with the highest probability.

From the training data set, the prior probability that an instance has a value of $X$ for attribute $A$ and is in class $Y$ can be calculated by counting the number of instances which have $A = X$ and $class = Y$. The probability that an instance would have $class = Y$ can be determined the same way. Bayes’ theorem can then be used to determine the probability that an instance is in $class Y$, given the instance has a value of $X$ for attribute $A$. The same calculation is performed for the other classes and the class with the highest probability is predicted for the given instance.

\[
P(class = Y \mid A = X) = \frac{P(A = X \mid class = Y)P(class = Y)}{P(A = X)}
\]

Using Bayes’ theorem, shown in Equation 1, posterior probability can be calculated from the prior probability. Building the model involves simply pre-computing and storing probabilities for all attribute and class combinations. When testing an instance the appropriate pre-computed probabilities are inserted into Equation 1, based on the attribute values of the instance, and the instance is placed in the class with the highest probability.
The Bayesian classification method is called *Naïve*, because the Bayes’ Theorem assumes the attributes are independent of each other. In reality the attributes are not independent. Additionally, the Bayes’ method assumes that the training set has the same statistical properties as the data universe, which is also often not the case.

### 2.3 Support vector machines

Support vector machines (SVM) were originally developed by Vladimir Vapnik, based on his earlier work on statistical learning theory [6, 7]. SVMs are based on linear models designed to separate a dataset using a dividing hyperplane which maximizes the margin between two classes. Points that lie along the margin hyperplanes are called support vectors. Once the model has been trained, classifying data only requires the linear equation for the hyperplane to determine on which side the new point lies. Figure 3 illustrates a simplified two-dimensional SVM example.

![SVM Example](image)

**Figure 3:** Example of a margin separating two dimensional data. The green points in the upper left quadrant represent one class and the blue points in the lower right another class. The dashed lines represent the margin between classes with the maximum width. The red solid line in the middle of the margin is the decision function, used to determine if a future point should be classified as blue or green.

Not all datasets can be separated linearly. In the non-linearly separable case, the original set of attributes, or input space, can be mapped into a higher dimension space via a non-linear mapping. The linear hyperplane with the maximum margin can be found in the higher dimensional feature space, which will translate to a non-linear decision function in the original input space. Mapping all data into a higher dimensional space suffers from the curse of dimensionality. The calculations in SVM solve a set of linear equations by ultimately calculating the inner product of each training instance against each other training instance. The mapping functions only ever appear in the SVM formulation together as an inner product; allowing the replacement of the inner product of two mapping functions with a kernel function. ‘The kernel trick’ is that if the kernel function is chosen very carefully and meets certain criteria, the calculation can mimic performing the inner product in the higher dimensional space without having to actually transform the data. [6, 7, 8] Calculations can be performed solely in the input space, and the true mapping function is not actually known. Figure 4 illustrates the transformation from input space to feature space and how the decision function translates back into input space.
The Gaussian kernel can model an infinite feature space where the data may be linearly separable; allowing the training data to be classified with 100% accuracy. However, increasing the feature space to higher dimensions can cause the model to fit too specifically to all the data, despite possible noise or outliers, called overfitting. To counter overfitting, the SVM formulation includes slack variables which allow some of the data points to stray into the margin; a user-defined slack variable determines how much. The use of slack variables is illustrated in two-dimensions in Figure 5.

Three kernel functions were considered: the linear kernel, the polynomial kernel and the Gaussian radial basis function (RBF) kernel. The linear kernel does not perform any mapping into higher dimensions and is in fact a special case of the polynomial kernel, where the parameter $E$ is set to 1. The linear kernel is naturally the fastest to calculate, but will only result in a linear hyperplane. The polynomial kernel raises the features to an exponential power, specified in the parameter $E$, which will model more complex hyper-surfaces in the input space. As a result, the polynomial kernel has increased computation time over the linear kernel, which increases with the parameter $E$. Lastly, the Gaussian kernel, which uses a radial basis function, models an infinite number of dimensions and can be faster than the polynomial kernel in some applications, but requires tuning to get the proper accuracy. The algorithm used in this experiment to compute the SVM with each of the described kernels is the sequential minimal optimization (SMO) method, developed by Platt [9].
It can be difficult for humans to interpret the model produced by SVMs, making classification with SVMs a poor choice for attempting to learn information for human consumption. However, SVMs are able to model much more complex relationships within the data than decision tree and Bayes’ classifiers.

3. FEATURE SET

Measurements were collected on 5 targets in an X-band compact radar range. The targets consist of 2 main battle tanks and 3 armored personnel carriers (APC). 720 ISAR images were used for each target, from 0 to 359 degrees azimuth at two different elevation angles. Based on the previous work described in Goyette et al [10], each of the 5 targets were compared to measurements of two reference targets, with the two reference targets set on different ground planes, which mimic different ground terrain dielectrics and clutter, from the original comparison measurements. Comparisons were computed for ISAR images at the same azimuth and elevation for both targets. The comparison between two ISAR images results in 12 attributes. The specifics of these attributes can be found in reference 10. The elevation angle was included as a 13th attribute, but the azimuth was not included. Azimuth was not included as an attribute because in the field generally the elevation angle of the instrumentation to the target is known, but the azimuth of the target is not. Each target was assigned a class, either “tank” or “apc”. The class combined with the 13 attributes makes up an instance in the dataset. Five targets were used: Target 1, 2 and 3 are APCs; Target 4 and 5 are tanks. Reference target 1 is identical to target 1, except the measurement was taken on a different ground plane: Reference target 2 is identical to target 2, except for the ground plane. Three datasets were created: one comparing all 5 targets with reference target 1, another comparing all 5 targets with reference target 2, and a third dataset which contained all instances from both of the first two datasets. Table 1 shows the data contained in each dataset.

Table 1: Datasets used in the experiment

<table>
<thead>
<tr>
<th>File</th>
<th>Data</th>
<th>Total instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dataset 1</td>
<td>Reference target 1 compared with targets 1, 2, 3, 4 &amp; 5</td>
<td>3600</td>
</tr>
<tr>
<td>Dataset 2</td>
<td>Reference target 2 compared with targets 1, 2, 3, 4, &amp; 5</td>
<td>3600</td>
</tr>
<tr>
<td>Dataset 3</td>
<td>File 1 &amp; 2 Combined (appended)</td>
<td>7200</td>
</tr>
</tbody>
</table>

Each comparison of targets resulted in 720 instances – 2 at each azimuth from 0 to 359 for 2 different elevation angles. Each of these 3 files were used to compare the 3 machine learning methods.

4. APPROACH

The WEKA tool, version 3.6.4, open source software created by the University of Waikato and licensed under the GNU license, was used to run the experiment [11]. Experiments were run on an iMac with an Intel Core 2 Duo 3.06GHz processor, with 8GB of 1067 MHz DDR3 RAM. For the naïve Bayes’ and decision tree experiment, the classifier was run on each of the 3 datasets using 10 fold cross-validation to determine accuracy [5]. Cross-validation randomly divides the data into 10 folds, then runs the experiment 10 times, each time leaving just one of the folds out for testing. The results are then averaged to determine accuracy. Cross-validation allows more of the data to be used for training, but avoids overfitting to the portion of the data which is used for training. In addition to accuracy, training time, kappa statistic, mean absolute error, root mean squared error, relative absolute error, root relative squared error, true positive rate, false positive rate, precision, recall, f-measure, and receiver operating characteristic (ROC) area were measured.

Support vector machines have additional parameters which can affect the accuracy. For example, the choice of a kernel function; two kernel functions were evaluated: polynomial and Gaussian radial-basis function (RBF). The SVM formulation used by the SMO algorithm, described in [8] takes a penalty parameter C. Each kernel function also has parameters. The polynomial kernel has a parameter E, which specifies what power the kernel will use. The linear kernel
is a special formulation of the polynomial kernel with $E = 1$. The RBF kernel takes a kernel parameter $\gamma$ which specifies the width of the radial basis function.

The choice of parameters can make a considerable difference in the final accuracy. The optimal parameters were determined using the grid search method outlined in [12]. The grid search included the values of $C = \{2^{-5}, 2^{-3}, 2^{-1}, 2^1, 2^3, 2^5, 2^7, 2^9, 2^{11}, 2^{13}, 2^{15}\}$ for all kernels and $E = \{1, 2, 3\}$ for the polynomial kernel and $Y = \{2^{-15}, 2^{-11}, 2^{-7}, 2^{-3}, 2^{-1}, 2^1, 2^3\}$ for the RBF (Gaussian) kernel. Tests were run using 10-fold cross-validation with the selected parameters which had the highest accuracy during tuning. With SVM, there is often a marked tradeoff between accuracy and training time.

5. RESULTS

5.1 Support vector machine parameter tuning

An ideal method for determining the optimal parameters to use with SVM has not been found. Currently the best method of tuning is to perform a log scale grid search [12]. A direct tradeoff between classifier accuracy and training time was found when determining the optimal parameters for SVM, particularly with the polynomial kernel. The Gaussian (RBF) kernel had the best overall results, however higher accuracy could be achieved using the Polynomial kernel with higher values of $E$, but the training time quickly became prohibitive. The overall results reported in the next section are those for the RBF kernel because the polynomial kernel was slow for higher accuracy configurations. The tradeoff of accuracy and training time is a difficult issue, because in the automated target recognition problem, high accuracy is vital, but quick response is also important.

![SVM Accuracy and Training Time (Polynomial Kernel)](image)

Figure 6: Using the polynomial kernel, the SMO Algorithms parameter $C$ is increased logarithmically, a linear increase in training time results, but only an extremely sublinear increase in accuracy. The training time for $E = 3$ became so large, the experiment was not continued past $C = 2048$.

Figure 6 shows the tradeoff between training time and accuracy. For each value of the kernel parameter $E$, the graph shows a pair of lines: one showing accuracy on the axis on the left and the other showing training time on the axis on the right. The training time increases linearly in the parameter $C$, where the accuracy increases sub-linearly. Increasing the value of $E$ also increases the accuracy, but comes with an increase in training time.
The tradeoff between accuracy and training time for the RBF kernel is illustrated in Figure 7. The RBF kernel will also see exponential increases in training time, but maximum accuracy is achieved with reasonable training time.

5.2 Comparison results

Each of the three machine learning methods was applied to each of the three datasets; the results are shown in Table 2. Accuracy and training time are shown graphically in Figure 8, the true and false positive rate are shown graphically in Figure 9, and the relative absolute error is shown graphically in Figure 10. The Support vector machines outperformed the other two methods in all measures. The highest accuracy was achieved for dataset 2, because the reference target compared against in dataset 2 is better at differentiating between the classes than the reference target used in dataset 1. The reference target in dataset 1 has tracks as opposed to wheels, so the target has similarities with both classes. However, the training time of the tuned SVM is significantly higher than the other two models.

Table 2: Comparison of the results for the three machine learning methods for each dataset

<table>
<thead>
<tr>
<th>Dataset 1</th>
<th>Dataset 2</th>
<th>Dataset 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bayes' DT SVM</td>
<td>Bayes' DT SVM</td>
<td>Bayes' DT SVM</td>
</tr>
<tr>
<td><strong>Accuracy (percent correct)</strong></td>
<td>60.94% 83.03% 86.42%</td>
<td>94.92% 98.50% 99.61%</td>
</tr>
<tr>
<td><strong>Training Time (seconds)</strong></td>
<td>0.02 0.16 148.90</td>
<td>0.03 0.07 1.86</td>
</tr>
<tr>
<td><strong>Kappa Statistic</strong></td>
<td>0.288 0.646 0.713</td>
<td>0.896 0.969 0.992</td>
</tr>
<tr>
<td><strong>Mean Absolute Error</strong></td>
<td>0.391 0.190 0.136</td>
<td>0.051 0.017 0.004</td>
</tr>
<tr>
<td><strong>Root mean squared error</strong></td>
<td>0.612 0.382 0.369</td>
<td>0.212 0.119 0.062</td>
</tr>
<tr>
<td><strong>Relative Absolute Error</strong></td>
<td>81.36% 35.60% 28.30%</td>
<td>10.63% 3.57% 0.81%</td>
</tr>
<tr>
<td><strong>Root Relative Sq. Error</strong></td>
<td>124.96% 77.92% 75.23%</td>
<td>43.26% 24.31% 12.73%</td>
</tr>
<tr>
<td><strong>True Positive Rate</strong></td>
<td>0.609 0.830 0.864</td>
<td>0.949 0.985 0.995</td>
</tr>
<tr>
<td><strong>False Positive Rate</strong></td>
<td>0.280 0.184 0.162</td>
<td>0.042 0.015 0.004</td>
</tr>
<tr>
<td><strong>Precision</strong></td>
<td>0.747 0.830 0.864</td>
<td>0.952 0.985 0.996</td>
</tr>
<tr>
<td><strong>Recall</strong></td>
<td>0.609 0.830 0.864</td>
<td>0.949 0.985 0.996</td>
</tr>
<tr>
<td><strong>F-Measure</strong></td>
<td>0.590 0.830 0.863</td>
<td>0.949 0.985 0.996</td>
</tr>
<tr>
<td><strong>ROC Area</strong></td>
<td>0.732 0.866 0.851</td>
<td>0.992 0.986 0.996</td>
</tr>
</tbody>
</table>
Figure 8: SVM shows a higher accuracy (left) for all three datasets, but also higher training time (right).

Figure 9: The true positive rate (left) measures the number of instances which were true positive out of the total number of positives. The false positive rate (right) measures the number of instances which were false positive out of the total number of positives. SVM has a higher false positive rate and a lower false negative rate.

Figure 10: SVM has a lower relative absolute error
For all the measurements except training time, the SVM was the best. For the datasets which were harder to classify, as evidenced by the lower accuracy of all the classifiers, the SVM had a significantly longer training time.

6. CONCLUSIONS

For all the datasets analyzed, the Support vector machine method was clearly superior, however significantly more tuning was required to reach a high level of accuracy. Decision tree was not far behind SVM, and in many applications the ability for the decision tree to be human interpretable might be worth the tradeoff in accuracy. However, in ATR, creating a model for interpretation by humans is not a requirement, and higher accuracy is likely to be more important. Bayes’ did not fare well against the other two, which is not surprising, since Bayes’ is based on probabilities, making the assumption that the dataset is going to be statistically similar to the data universe. In business applications where the information gathered is generally a subset of the data universe, basing a classifier on probabilities works well. In an ATR application, the training set will not necessarily include a statistically similar number of measurements of the same targets found in the real world, rather the training datasets may contain one or two examples of each type of target. Support vector machines have been shown to be viable for classifying processed ISAR data.

7. REFERENCES


8. ACKNOWLEDGEMENTS

This work was supported by the U.S. Army National Ground Intelligence Center.