APPLYING AN EFFICIENT *K*-NEAREST NEIGHBOR SEARCH TO FOREST ATTRIBUTE IMPUTATION

ANDREW O. FINLEY [1], RONALD E. MCROBERTS [2], ALAN R. EK [1]

[1] Department of Forest Resources
University of Minnesota
115 Green Hall
1530 Cleveland Ave. N.
St. Paul, Minnesota 55108, USA

[2] Forest Inventory and Analysis
North Central Research Station
USDA Forest Service
1992 Folwell Ave.
St. Paul, Minnesota 55108, USA

ABSTRACT. This paper explores the utility of an efficient nearest neighbor (NN) search algorithm for applications in multi-source *k*NN forest attribute imputation. The search algorithm reduces the number of distance calculations between a given target vector and each reference vector, thereby, decreasing the time needed to discover the NN subset. Results of five trials show gains in NN search efficiency ranging from 75 to 98 percent for $k = 1$. The search algorithm can be easily incorporated into routines that optimize feature subsets or weights, values of k, distance decomposition coefficients, and mapping.

1 INTRODUCTION

Qualities of the nonparametric *k*-nearest neighbor (*k*NN) technique make it an attractive tool for forest attribute estimation and mapping. Some of the technique's notable qualities include: simultaneous imputation of multiple dependent variables; preservation of the covariance structure among dependent and independent variables (Moeur, 1988; Tomppo, 1991); relaxed assumptions such as normality or homoscedasticity that are typically required by parametric estimators, and; the technique is straightforward and easy to implement.

A significant disadvantage of the *k*NN technique is the search through the reference set to find the nearest neighbor (NN). This search can make algorithm parameterization and subsequent mapping very time consuming. This paper describes one approach for improving the NN search efficiency. The modified search algorithm reduces the number of distance calculations between a given target vector and each reference vector, thereby, decreasing the time needed to discover the NN subset. The general approach used here was separately proposed by Ra and Kim (1993) and Cheng and Lo (1996). This paper reviews their proposed efficient NN search within the context of multi-source *k*NN forest attribute mapping, and compares the efficient and conventional NN search algorithms using several data sets.

2 METHODS

2.1 kNN technique. A thorough description of the *k*NN technique is presented by McRoberts et al. (2002) and Franco-Lopez et al. (2001). In short, the *k*NN technique is a non-parametric method used to impute a categorical or continuous variable from a point, or k points, for which both dependent and

independent variables are known, to a point for which only independent variables are known. The set of points for which both the independent and dependent variables are known is referred to as the reference set and the vector of independent variables associated with each point is labeled **c**. The set of points for which only independent variables are known is referred to as the target set, with each vector labeled **x**.

A distance metric, *d*, is used to associate a given **x** with a set of **c**. This set of **c**, of predefined size *k*, is chosen from the reference set and consists of those vectors having the minimum distance to the given **x**.

2.2 Mean distance inequality. Squared Euclidean distance is a common metric used to measure the "nearness" between a given target vector, **x**, and each vector, **c**, in the reference set (Tomppo, 1991; Franco-Lopez et al., 2001). In application, the independent variables are typically normalized ($\mu = 0$, $\sigma = 1$) or a weight vector is included in the distance calculation. Unless otherwise specified, we compute distance with normalized variables. Squared Euclidean distance is defined as:

$$d_{E,i} = \sum_{j=1}^{J} (x_j - c_{i,j})^2 \tag{1}$$

where J is the vectors' dimension (i.e., the number of independent variables considered) and i represents the i^{th} **c** vector in the reference set. Squared mean distance is defined as:

$$d_{M,i} = \left(\sum_{j=1}^{J} x_j - \sum_{j=1}^{J} c_{i,j} \right)^2 \tag{2}$$

The components of the squared mean distance (i.e., the sum of the elements in the **x** and **c** vectors) can be found at start-up time, making this a very simple computation. Ra and Kim (1993) note that this metric is a near sufficient statistic for squared Euclidean distance. This relationship and the following inequality form the base for the efficient NN search. Following Cheng and Lo (1996), we define M_x and M_{c_i} to be the means of the **x** and $\mathbf{c}_i$ elements, respectively, and M_{M_i} to be the difference between M_x and M_{c_i}. Thus,

$$\sum_{j=1}^{J} (x_j - c_{i,j}) = J(M_x - M_{c_i}) = JM_{M_i} \tag{3}$$

Deriving the useful inequality:

$$\sum_{j=1}^{J} [(x_j - c_{i,j}) - M_{M_i}]^2 \geq 0$$

$$\sum_{j=1}^{J} [(x_j - c_{i,j})^2 - 2M_{M_i}(x_j - c_{i,j}) + M_{M_i}^2] \geq 0$$

$$\sum_{j=1}^{J} (x_j - c_{i,j})^2 - 2M_{M_i} \sum_{j=1}^{J} (x_j - c_{i,j}) + JM_{M_i}^2 \geq 0$$

$$\sum_{j=1}^{J} (x_j - c_{i,j})^2 - 2M_{M_i}(JM_{M_i}) + JM_{M_i}^2 \geq 0$$

$$\sum_{j=1}^{J} (x_j - c_{i,j})^2 - JM_{M_i}^2 \geq 0$$

$$\sum_{j=1}^{J} (x_j - c_{i,j})^2 \geq JM_{M_i}^2 \tag{4}$$

The relationship derived in (4) provides the inequality (5) which can be used to reject vectors in the reference set without computing their full squared Euclidean distance. It is also important to note that the inequality is not affected by the addition of a weight vector, and therefore is also useful for weighted squared Euclidean distance.

$$d_{M,i} \leq Jd_{E,i} \tag{5}$$

2.3 Search implementation. This section outlines the steps in the efficient NN search. For the applications described here, the algorithm begins by calculating the components of $d_{M,i}$ (i.e., the sum of the x and c elements). In a mapping application, x might represent a vector of feature values associated with each pixel. If this is the case, then the sum of the vectors' elements can be computed and added as a new feature to the feature stack (i.e., stack of independent variable data layers) for later retrieval. For the c vectors in the reference set, the sum of each vectors' elements is calculated and stored in a tuple (i.e., record) along with the original vector and associated independent variables. These tuples are then added to a hash. Once the hash of all reference vectors is formed it is sorted based on M_{c_i}. Note that this sort occurs only once. Also, the direction of the sort does not matter; our illustration sorts from small to large values. Once the hash is sorted the NN search process for a given target vector (x) follows these steps for $k = 1$ (refer to Figure 1 for an illustration of the search pattern):

1. The search begins with the tuple in the reference set that has the minimum $d_{M,i}$. For example, in Figure 1, the start tuple is labeled $\mathbf{c}_i$. Because this is a sorted list, the start tuple can be found with a fast binary search.

2. The $d_{E,i}$ is calculated and stored along with the tuple's elements (or pointer to the tuple).

3. The $d_{M,i+1}$ is then computed for the $\mathbf{c}_{i+1}$ in the search pattern. This value is compared against the $d_{E,i}$ found in Step 2 using inequality (5). If $d_{M,i+1} \leq Jd_{E,i}$ then $d_{E,i+1}$ is calculated between the x and $\mathbf{c}_{i+1}$. Then if $d_{E,i+1} < d_{E,i}$ the $\mathbf{c}_{i+1}$ tuple is set as the NN. If $d_{M,i+1} > d_{E,i}$, the search terminates in that direction in the hash.

4. The search iterates through Steps 1-3 until the inequality fails in both directions or the end of the hash is reached. In Figure 1, the inequality failed at $\mathbf{c}_{i+2}$ and $\mathbf{c}_{i-4}$

If $k > 1$, then the comparison described in Step 3 is done between the current $\mathbf{c}$ and the k^{th} NN found thus far. Following Step 4, the dependent variables from the selected *k*NN are imputed to the target using methods described in previous papers.

3 Analysis

3.1 Trial data sets. The efficient NN search was tested on five data sets. Each data set associates Landsat imagery with ground based inventory data. High classification accuracy is not the objective of this analysis; rather, we wish evaluate the utility of the efficient NN search. For this reason, the data sets are only briefly described.

Trial 1 - Comprised of 6,435 reference points and 4 bands of Landsat MSS imagery. The reference points represent 6 classes of exposed soil types. (See Srinivasan 1996 for a complete description of the data set)

Trial 2 - Comprised of 2,484 reference points and 13 bands of Landsat TM and ETM+ imagery. The reference points represent 10 classes of landcover. (See RSLa for a complete description of the data set)

Trial 3 - Comprised of 989 reference points and 21 bands of Landsat TM and ETM+ imagery. The reference points represent non-forest, red pine plantations, and other forest types. (See RSLb for a complete description of the data set)

Trial 4 - Comprised of 197 reference points and 6 bands of Landsat TM imagery. The reference points represent 4 classes of forest type. (See Crookston et al. 2002 for a complete description of the data set)

Trial 5 - Comprised of 324 reference points and 12 Tassel Cap and NDVI variables derived from Landsat TM imagery. The reference points represent timber volume. (See McRoberts et al. 2002 for a complete description of the data set)

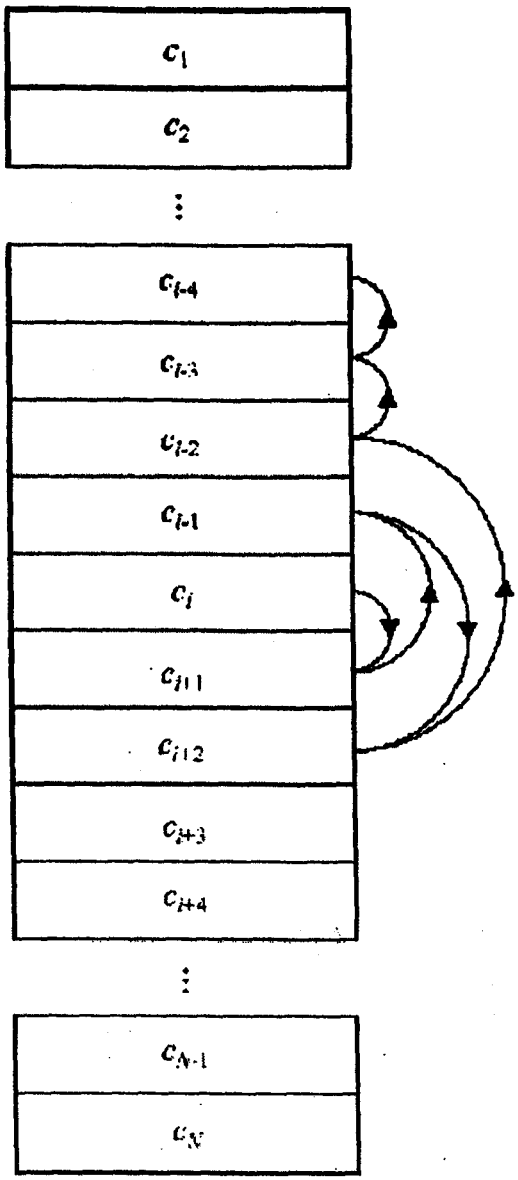

Figure 1. Mean distance ordered search. The search starts at $\mathbf{c}_i$ which has the minimum $d_{M,i}$ to the target vector. In this example the search terminates at $\mathbf{c}_{i+2}$ and $\mathbf{c}_{i-4}$.

3.2 Measurement of efficiency gain. The utility of the efficient NN algorithm was judged on its ability to reduce the number of reference vectors considered in a search for each given target vector. For each trial data set we preformed a leave-one-out (LOO) cross-validation using the efficient NN algorithm for each k = 1 - 35. LOO is an accepted method for generating unbiased estimates of expected classification or estimation error for the *k*NN estimator (Gong, 1986). Given

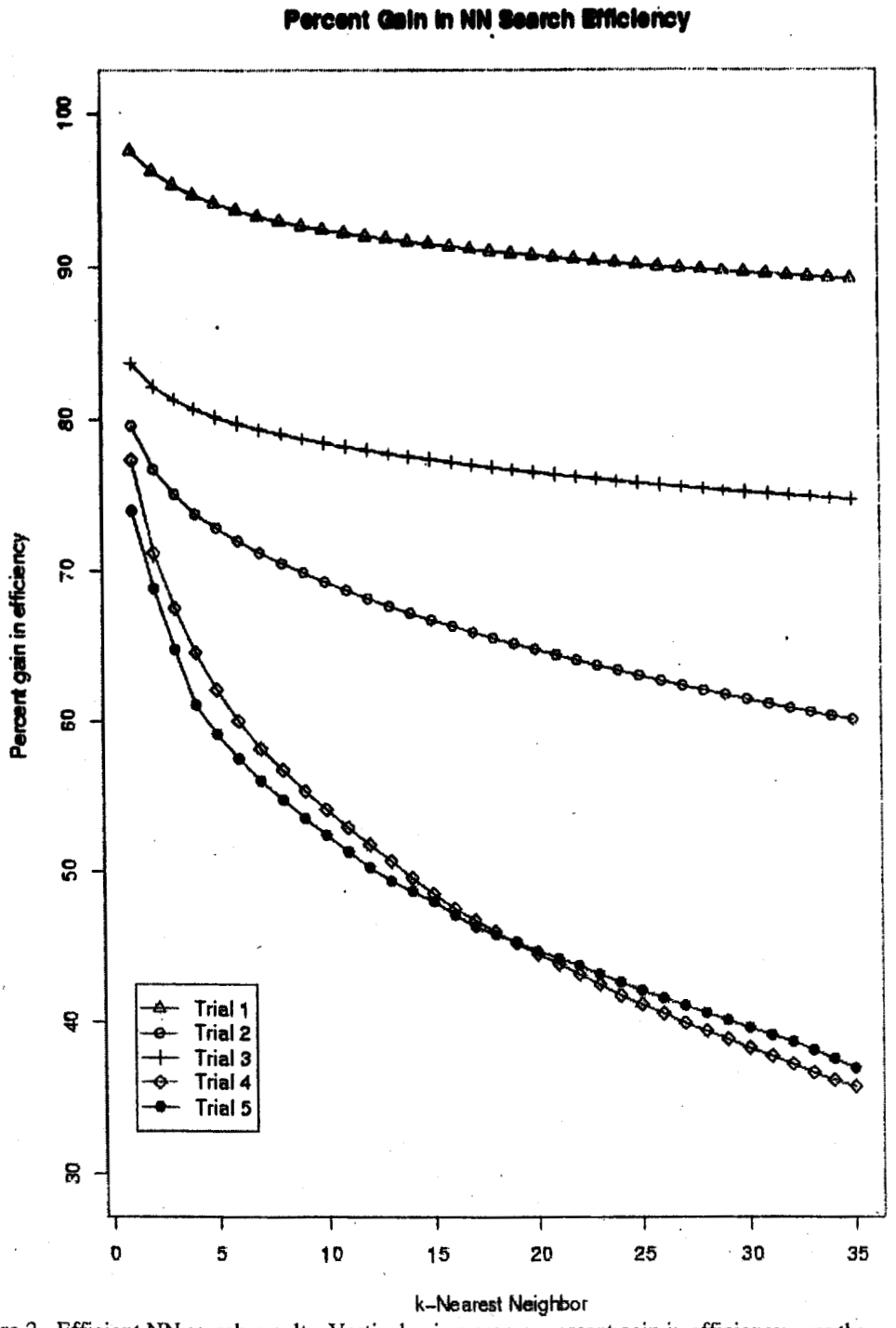

Figure 2. Efficient NN search results. Vertical axis measures percent gain in efficiency over the conventional NN search for values of k = 1-35.

n vectors in the reference set, LOO removes the i^{th} vector then uses the n-1 remaining vectors to estimate its value. This process is then repeated for all n vectors, and the appropriate error statistics are generated based on the n observed and estimated values.

For the purpose of these trials LOO was used only to count the number of reference vectors considered in a given NN search. For each value of k, the total number of vectors considered in the

efficient NN search algorithm LOO were counted (referred to as p). Then $1-[p/n(n-1)]$ was expressed as a percent gain in efficiency.

4 Results and Discussion

Results for the five trials are presented in Figure 2. Trial 1 data set had the largest gain in efficiency (greater than 90 percent), requiring a search of less than 10 percent of the reference set across all values of k. Trials 4 and 5, showed less substantial gains in efficiency ranging between approximately 75 and 35 percent.

Two factors limit the ability of the efficient NN search algorithm. First, at the beginning of the NN search, it is necessary to select the $\mathbf{c}_i$ that has the minimum $d_{E,i}$ to the given target vector. Because the algorithm relies on $d_{M,i}$ to select the location in the hash to begin the search, it is critical to have strong correlation between $d_{E,i}$ and $d_{M,i}$. If $d_{M,i}$ is a poor proxy for $d_{E,i}$ then the search will fail to quickly find the NN subset.

Second, the level of dispersion within the reference set point cloud in feature space will affect the algorithm's efficiency. If reference points are tightly clustered (i.e., under dispersed) then more points will pass the early stop inequality (5) and, as a result, they will need to be considered with a full squared Euclidean distance comparison. Conversely, as the point cloud becomes more dispersed the inequality will fail earlier in the NN search, thereby reducing the number of reference points considered.

All trials show that gain in search efficiency decreases with increasing values of k. This trend occurs because there is a greater likelihood that the $d_{E,i}$ of the intermediate k^{th} reference vector found and the $d_{M,i\pm1}$ reference vectors will pass the early stop inequality. This idea is similar to the idea of point dispersion described above.

5 Conclusion

This paper explored the utility of an efficient NN search algorithm for applications in multi-source *k*NN forest attribute imputation. Our trials suggest that the algorithm can greatly reduce the number of reference vectors considered in the NN search, this translates into increased efficiency for routines that optimize feature subsets or weights, values of k, distance decomposition coefficients, and mapping.

Literature Cited

RSLa. Forest Service red pine classification project final report. http://www.rsl.gis.umn.edu/metadata/FSRedPine.html

RSLb. NASA REASON final report. http://rsl.gis.umn.edu/metadata/REASONFinalReport.html

Cheng S.M., and Lo K.T. 1996. Fast clustering process for vector quantisation codebook design. Electronic Letters. 32:4,311-312.

Crookston, N.L., M. Moeur, D. Renner. 2002. Users guide to the Most Similar Neighbor Imputation Program Version 2. Gen. Tech. Rep. RMRS-GTR-96. Ogden, UT: U.S. Department of Agriculture, Forest Service, Rocky Mountain Research Station. 35 p.

Franco-Lopez H., A.R. Ek, and M.E. Bauer. 2001. Estimation and mapping of forest stand density, volume, and cover type using the k-nearest neighbor method. Remote Sensing of Environment. 77:251-274.

Gong, G. 1986. Cross-validation, the jackknife, and the bootstrap: excess error estimation in forward logistic regression. Journal of the American Statistical Association. 81:393,108-113.

Hawang W.J., S.S. Jeng, and B.Y. Chen. 1997. Fast codeword search algorithm using wavelet transform and partial distance search techniques. Electronic Letters. 33:5,356-366.

Holmgren J., S. Joyce, N. Nilsson, and H. Olsson. 2000. Estimating stem volume and basal area in forest compartments by combining satellite image data with field data. Scandinavian Journal of Forest Research. 15:103-111.

McRoberts R.E., M.D. Nelson, and D.G. Wendt. 2002. Stratified estimation of forest area using satellite imagery, inventory data, and the k-Nearest Neighbor technique. Remote Sensing of Environment. 82:457-468.

Moeur M. 1988. Nearest neighbor inference for correlated multivariate attributes. In A.R. Ek, S.R. Shifley, and T.E. Burk (Eds.), Forest growth modeling and prediction. Proceeding of the IUFRO conference (Gen. Tec. Rep. 120, pp. 716-723). St. Paul, MN: US Department of Agriculture, Forest Service, Norther Central Research Station.

Ra S.W., and Kim J.K. 1993. A fast mean-distance-ordered partial codebook search algorithm for image vector quantization. IEEE Transactions on Circuits and Systems. 40:9,576-579.

Tomppo E. 1991. Satellite imagery-based national forest inventory of Finland. International Archives of Photogrammetry and Remote Sensing. 28:419-424.

Srinivasan A. 1996. UCI Repository of machine learning databases, statlog http://www.ics.uci.edu/ mlearn/MLRepository.html. Irvine, CA: University of California, Department of Information and Computer Science.