Knowledge Transfer with a Multiresolution Ensemble of Classifiers

Eric Eaton EEATON1@CS.UMBC.EDU
Marie desJardins MARIEDJ@CS.UMBC.EDU
University of Maryland Baltimore County, Computer Science and Electrical Engineering Department, 1000 Hilltop Circle, Baltimore, MD 21250 USA

Abstract

We demonstrate transfer via an ensemble of classifiers, where each member focuses on one resolution of data. Lower-resolution ensemble members are shared between tasks, providing a medium for knowledge transfer.[1]

1. Introduction

Most related objects are similar when viewed at a low resolution. For example, low-resolution images of most four-legged farm animals have the same general shape. Knowledge learned at a low resolution may apply to all of these animals (e.g., has four legs, eats grass). At higher resolutions, details begin to emerge that differentiate between them.

Inspired by this idea, we explore the use of *multiresolution learning* for *knowledge transfer* between tasks. We claim that by exploiting the similarities between objects at low levels of detail, learning at multiple resolutions can facilitate transfer between related tasks.

Low-resolution representations are simple and therefore easy to learn, but the value of what can be learned from them is limited. High-resolution representations have a much higher value of what can be learned from them, but learning is more difficult due to the added complexity. Learning from low-resolution data may yield limited amounts of knowledge, but that knowledge will often transfer to other related objects. This knowledge provides both a foundation for learning from the higher-resolution data, and a base of general knowledge applicable to a class of objects.

Learning at multiple resolutions has been shown to significantly improve generalization and classification time in single-task problems (Liang & Page, 1997; He et al., 2005; Blayvas & Kimmel, 2003). Multiresolution representations have also been used successfully in image retrieval systems (Li & Wang, 2003) and image classification systems (Zhang & Hebert, 1997).

We provide an ensemble framework for providing the transfer between learning tasks, with each member focusing on one resolution level. The low-resolution classifiers are shared between tasks, allowing knowledge transfer between these tasks.

[1]This work was supported by NSF ITR #0325329.

Appearing in the ICML-06 Workshop on Structural Knowledge Transfer for Machine Learning, June 29, Pittsburgh, PA.

2. Multiresolution Representations

Our experiments use feature vectors as input, so we represent the instance space at multiple resolutions. To do this, we use two methods: (1) chopping the space into hypercubes and repeatedly combining them, and (2) repeatedly merging correlated attributes.

Both methods take as input labeled instance vectors $\{x_i, y_i\}_{i=1}^N$, where each x_i belongs to the instance space $X \subset \mathbb{R}^d$, and each y_i belongs to the set of binary classes $Y = \{-1, 1\}$.

The Hypercubes Representation breaks the instance space X into hypercubes at the highest resolution, then repeatedly combines these hypercubes to generate successively lower resolutions. This representation was previously used for multiresolution learning by He et al. (2005); we use their notation.

Let Ω be a hypercube in $\mathbb{R}^d$ that contains the instance space X.[2] For each dimension of Ω, we slice that dimension into l equal-size segments. By this method, Ω is broken into l^d hypercubes. For Ω_k, the k^{th} resolution, we set $l = 2^k$. Therefore, $\Omega_k = \bigcup_{i=1}^{2^{k^d}} o_k^i$, where o_k^i denotes the i^{th} hypercube in Ω_k. The multiresolution representation of the instance space X with r resolution levels is given by $\{\Omega_k\}_{k=1}^r$.

[2]We define Ω to be six standard deviations larger in each dimension than the values in the training set.

For each level of resolution, $k = 1 \ldots r$, we can map instance $x \in X$ to the hypercube o_k^i containing x via the function $g_k : X \rightarrow \Omega_k$. Each hypercube o_k^i is represented by the coordinates of its center. Therefore, each $x \in X$ has a multiresolution representation $R(x) = \{center(g_k(x))\}_{k=1}^r$.

We estimate the label for hypercube o_k^i as the majority label of all instances mapped to it by $g_k()$; ties are broken by uniform random selection. The label for x has the multiresolution representation $S(x) = \{label(g_k(x))\}_{k=1}^r$. These labels are used solely for training; the actual labels are used for testing.

The Dimension-Merge Representation repeatedly merges correlated dimensions of the instance space X. It determines the most correlated distinct dimensions of X and then merges those dimensions by mapping them onto a linear regression fit of the correlation. This method is similar to the Hierarchical Dimensionality Reduction algorithm (Duda et al., 2001), which takes a set of data clusters and repeatedly merges the most correlated distinct clusters.

The Dimension-Merge algorithm is given in Figure 1. Each successive lower resolution contains one less dimension than the previous resolution, and the precision of the values along the merged dimension is reduced naturally by the merging process. The Dimension-Merge algorithm determines the sequence of attribute merges from the set of training instances $\{x_i\}_{i=1}^N$. During the testing phase, the resolutions of the test instances are computed using the sequence of attribute merges determined during training.

Standard Feature Selection Methods (e.g., principal components analysis, information gain) can be used to repeatedly reduce the dimension of the instance space; however, they typically produce successive resolutions with significant overlap. Consequently, using them in our ensemble architecture (Section 3) produces an ensemble of members with highly correlated errors. The ensemble members are not diverse; therefore, the ensemble will not be more accurate than any of its members (Dietterich, 2000), yielding poor results in our experiments (omitted for space reasons).

Given: $X = \{x_i\}_{i=1}^N$, $x_i \in \mathbb{R}^d$
  Set the array of resolutions $R = \{\}$.
  Set $R[d] = X$.
  for k from $d-1$ downto 1 **do**
    Set $R[k] = R[k+1]$.
    Compute the correlation matrix for all pairs of distinct dimensions of $R[k]$.
    Determine the most correlated distinct attributes of $R[k]$, say d_1 and d_2.
    Determine the linear regression line l for dimensions d_1 and d_2 of $R[k]$.
    for $i = 1 \ldots N$ **do**
      Let r_i be the i^{th} element of $R[k]$.
      Project the point $(r_i[d_1], r_i[d_2])$ onto l.
      Let v be the Euclidean distance between $(0,0)$ and the projection of $(r_i[d_1], r_i[d_2])$ onto l.
      Set $r_i[d_1] = v$.
    end for
    Delete the dimension d_2 of $R[k]$.
  end for
  return R

Figure 1. The Dimension-Merge algorithm.

3. The Multiresolution Ensemble

Given a set of multiresolution data R with r resolutions and associated class labels S, we create an ensemble of classifiers $\{c_k\}_{k=1}^r$ where each member focuses on one resolution of the data. Let $R_k(X)$ represent the instance space X viewed at resolution k, for $k = 1 \ldots r$. The k^{th} ensemble member, c_k, is trained on and outputs class predictions for instances from $R_k(X)$.

The ensemble's prediction is a weighted majority vote of the member classifiers' predictions. We use the Adaboost weighting scheme (Schapire, 1999) to determine the weight of each member classifier. The weight α_k of classifier c_k is inversely proportional to its error ϵ_k on the training data at resolution k:

$$\alpha_k = \frac{1}{2} \ln \left(\frac{1 - \epsilon_k}{\epsilon_k} \right) \quad . \tag{1}$$

3.1. Knowledge Transfer with the Ensemble

In this paper, we assume that transfer occurs between two tasks, A and B. To allow knowledge transfer between the tasks, we combine the A and B ensembles, E_A and E_B, into a tree. Each task has a unique multiresolution ensemble, but the lower-resolution ensemble members are shared between the tasks.

Suppose that the ensemble tree splits after resolution i. (In this paper, the split points are manually specified.) The root of the tree contains the set of shared lower-resolution members $n_{AB} = \{c_1, \ldots, c_i\}$. Below the root, the tree branches into two sets of non-shared higher-resolution members, $n_A = \{c_{i+1}^A, \ldots, c_r^A\}$ and $n_B = \{c_{i+1}^B, \ldots, c_r^B\}$. Then the two ensembles are given by $E_A = n_{AB} \bigcup n_A$ and $E_B = n_{AB} \bigcup n_B$. Members in n_{AB} are shared between the A and B ensembles, and trained on both A and B. Members in n_A and n_B are trained on only A or only B, respectively. In future work, we plan to generalize this single-level binary tree structure to more than two tasks, where "more similar" tasks share more of their ensemble structure than "less similar" tasks.

The weights $\{\alpha_k\}_{k=1}^{i}$ of the shared ensemble members are determined based on both tasks and are shared between ensembles. We are currently exploring methods for determining the split point computationally, and using unshared weights for the shared members.

4. Experiments

We conducted experiments using the letter dataset from the UCI Machine Learning Repository (Blake & Merz, 1998). The letter dataset consists of various fonts of the twenty-six capital letters in the English alphabet characterized by 16 features. We use a subset of the letter dataset consisting of 1,000 instances. We examine several tasks involving transfer in the recognition of pairs of similar letters: "C" to "G," "O" to "Q," and "M" to "W." We also tested several pairs of dissimilar letters, and show results for one pair of letters that are similar in terms of construction, but differ when viewed at a low resolution: "F" to "E."

For example, consider transfer from the task of recognizing "C" to recognizing "G." We select out all instances of the target concept ("C" and "G") from the data set D. We create the following sets:

- C: all "C" instances in D, labeled as *positive*
- G: all "G" instances in D, labeled as *positive*
- G_{update}: subsets of G, of various sizes
- *Neg*: $D - (C \bigcup G)$, labeled as *negative*.

The sets C, G, and *Neg* are divided into equal-sized training and testing portions (C_{train}, C_{test}, etc.). The sizes of Neg_{train} and Neg_{test} are trimmed to $\max(|C|, |G|)$, so the ratio of positive to negative instances in the training and test sets is roughly two-thirds, with the exception of the training set for the shared ensemble members.

Suppose that the ensemble tree splits after member i. The shared classifiers $n_{CG} = \{c_1, \ldots, c_i\}$ are trained on $C_{train} \bigcup G_{update} \bigcup Neg_{train}$; the other "C" classifiers $n_C = \{c^C_{i+1}, \ldots, c^C_r\}$ are trained on $C_{train} \bigcup Neg_{train}$; and the other "G" classifiers $n_G = \{c^G_{i+1}, \ldots, c^G_r\}$ are trained on $G_{update} \bigcup Neg_{train}$. We evaluate the "C" ensemble using $C_{test} \bigcup Neg_{test}$ and the "G" ensemble using $G_{test} \bigcup Neg_{test}$.

As the baseline for transfer, we train a single multiresolution "G" ensemble using $G_{update} \bigcup Neg_{train}$. We compare the learning curves for the "G" ensemble to the ensemble tree across varying sizes of G_{update}.

We experimented using both the Hypercubes representation (with $r = 7$ resolutions, specified manually as used by He et al. (2005)) and the Dimension-Merge representation ($r = 16$, since the letter data set has 16 dimensions). Our experiments used the J48 implementation of C4.5 provided in the Weka toolkit (Witten & Frank, 2005) as the base classifier.

4.1. Results and Discussion

Figure 2 shows the results of our experiments. Consider the learning task "C" to "G," depicted in Figures 2(a) and 2(b). The figures show the "C to G Tree" for learning with transfer against the "G Ensemble" for learning without transfer.

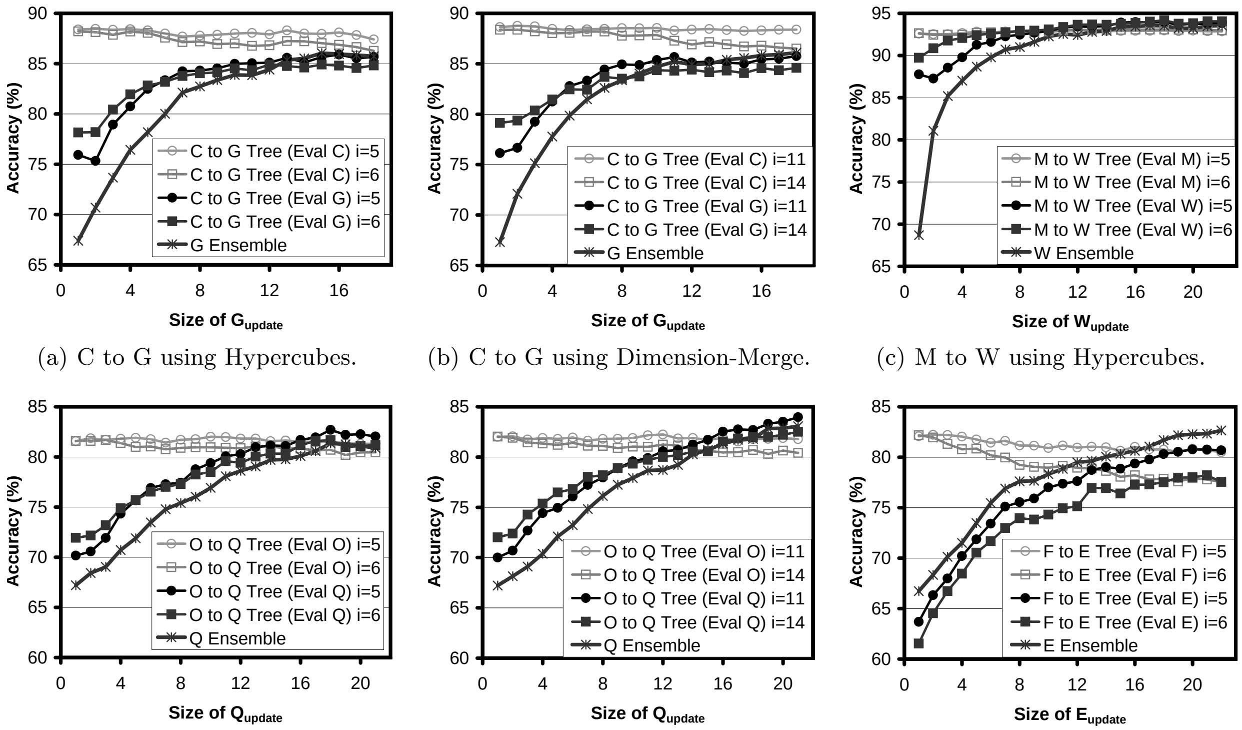
The black lines with round and square markers show the multiresolution ensemble tree's performance on the transfer task (recognizing "G"). The light gray lines show the multiresolution ensemble tree evaluated on the background task (recognizing "C"), demonstrating how the performance on the background task varies as the shared ensemble members learn the transfer task.

We explored using every possible split point (i) for the tree, and plot the two that show the greatest transfer in most cases. In every case, as i decreases to 1, the learning curve approaches learning in isolation using the single ensemble. When all ensemble members are shared ($i = 7$ on Hypercubes and $i = 16$ on Dimension-Merge), the ensemble trees show excellent transfer with small transfer task (update) set sizes (in some cases, more than shown on the plots). However, with larger numbers of transfer task instances, the performance may drop below that of the single ensemble, due to interference between the tasks.

Figures 2(a)–2(e) show that learning using the multiresolution ensemble tree can outperform learning the transfer task in isolation. These figures also show that the optimal number of shared ensemble members may vary depending on the size of the transfer task set, and that the ensemble tree's performance on the background task may decrease as i increases. These observations support our future work of making the ensemble tree dynamic in response to the training data. We have observed cases where the performance on the background task *increased* slightly with additional transfer task instances – an ideal case for transfer.

From the results, it appears that both the Hypercubes and Dimension-Merge representations are sufficient for allowing task transfer.

Figure 2(f) depicts a situation where the background task and transfer task are similar from a standpoint of letter construction ("F" and "E" differ by one stroke); however, the letters differ from a low-resolution viewpoint in the feature space. Using transfer inhibits



(a) C to G using Hypercubes. (b) C to G using Dimension-Merge. (c) M to W using Hypercubes.

(d) O to Q using Hypercubes. (e) O to Q using Dimension-Merge. (f) F to E using Hypercubes.

Figure 2. Learning curves for the letter recognition transfer over 200 trials using multiresolution ensembles.

learning in this task: the best results are obtained by sharing only one ensemble member and are identical to that of learning the transfer task in isolation. We have tested the transfer between several other pairs of dissimilar letters and obtained similar results.

5. Conclusion and Future Work

Our results show that the multiresolution ensemble tree can successfully transfer knowledge between learning tasks. Currently, we are exploring methods for computationally selecting the ensemble tree split point, adapting the split point dynamically in response to the training data, and creating ensemble trees with multiple ensembles. Our future work also includes adapting the multiresolution transfer framework to use image data and alternate resolution methods.

References

Blake, C., & Merz, C. (1998). UCI repository of machine learning databases. Available on-line: http://www.ics.uci.edu/~mlearn/MLRepository.html.

Blayvas, I., & Kimmel, R. (2003). Machine learning via multiresolution approximation. *IEICE Transactions on Information and Systems*, *E86-D*, 1172–1180.

Dietterich, T. G. (2000). Ensemble methods in machine learning. *Proceedings of the First International Workshop on Multiple Classifier Systems* (pp. 1–15).

Duda, R. O., Hart, P. E., & Stork, D. G. (2001). *Pattern Classification*. Wiley-Interscience. 2nd edition.

He, Y., Zhang, B., & Li, J. (2005). A new multiresolution classification model based on partitioning of feature space. *Proceedings of the IEEE Conference on Granular Computing* (pp. 462 – 467).

Li, J., & Wang, J. Z. (2003). Automatic linguistic indexing of pictures by a statistical modeling approach. *IEEE Trans. Pattern Anal. Mach. Intell.*, *25*, 1075–1088.

Liang, Y., & Page, E. W. (1997). Multiresolution learning paradigm and signal prediction. *IEEE Trans. Signal Process.*, *45*, 2858–2864.

Schapire, R. E. (1999). A brief introduction to boosting. *IJCAI* (pp. 1401–1406).

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann. 2nd edition.

Zhang, D., & Hebert, M. (1997). Multi-scale classification of 3-D objects. *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition* (pp. 864–869).