Learning the Relative Importance of Features in Image Data

Aparna Varde[1], Elke Rundensteiner[2], Giti Javidi[1], Ehsan Sheybani[3] and Jianyu Liang[4]
1. Department of Math and Computer Science, Virginia State University, Petersburg, VA
2. Department of Computer Science, Worcester Polytechnic Institute, Worcester, MA
3. Department of Engineering and Technology, Virginia State University, Petersburg, VA
4. Department of Mechanical Engineering, Worcester Polytechnic Institute, Worcester, MA
(avarde@vsu.edu, rundenst@wpi.edu, gjavidi@vsu.edu, esheyban@vsu.edu, jianyul@wpi.edu)

Abstract

In computational analysis in scientific domains, images are often compared based on their features, e.g., size, depth and other domain-specific aspects. Certain features may be more significant than others while comparing the images and drawing corresponding inferences for specific applications. Though domain experts may have subjective notions of similarity for comparison, they seldom have a distance function that ranks the image features based on their relative importance. We propose a method called FeaturesRank for learning such a distance function in order to capture the semantics of the images. We are given training samples with pairs of images and the extent of similarity identified for each pair. Using a guessed initial distance function, FeaturesRank clusters the given images in levels. It then adjusts the distance function based on the error between the clusters and training samples using heuristics proposed in this paper. The distance function that gives the lowest error is the output. This contains the features ranked in the order most appropriate the domain. FeaturesRank is evaluated with real image data from nanotechnology and bioinformatics. The results of our evaluation are presented in the paper.

1. Introduction

Managing scientific data presents challenging research issues [6, 11, 17, 19]. The results of scientific experiments are often depicted as three-dimensional images. In several applications, the similarity between the images depicts the similarity between the phenomena that led to them [4, 14]. This similarity is often identified by individual features in the images such as color, size, depth, domain-specific observations and so forth.

Computational analysis of these images involves processes such as clustering [8], building cluster representatives [8], similarity search [9] and data visualization [18]. In such analysis, it is important to compare the images analogous to a domain expert. The output of the analysis helps to draw inferences about the corresponding phenomena [4, 14]. For example, consider the following application in the domain of nanotechnology [17]. Nanostructure images obtained as a result of phenomena such as etching [4] are clustered based on their similarity. A representative image is selected for each cluster such that it is closest to all images within the cluster. This representative image is then used to categorize the given phenomena, e.g., the nature of the etching. In order to achieve effective clustering and select a good cluster representative, it is essential to capture the semantics of the images. Certain features such as nanoparticle size and interparticle distance may be crucial while others such as color may not be. Thus it is important to incorporate such factors in defining image similarity.

Morever, certain aspects in image comparison may be related to metadata about the images. For example, images taken with a scanning electron microscope (SEM) are different from those with a transmission electron microscope (TEM). An SEM scans images at a particular cross-section, while a TEM penetrates the object in capturing the image [4, 14]. The level of zooming also needs to be taken into account. Given all these features pertaining to image data and metadata, and their relative importance, automating image comparison becomes even more challenging. Domain experts may at best have subjective notions of similarity but not a distance function that incorporates such features and their relative importance. Hence there is a need to learn such a domain-specific distance function.

In this paper, we propose an algorithm called FeaturesRank that learns the relative importance of image features. The inputs to FeaturesRank are training samples with pairs of images provided by experts. For each pair, experts identify whether they consider its images to be different or similar and indicate the extent of similarity based on levels. They also identify features that are potentially applicable to the images. Based on this, FeaturesRank defines a preliminary distance function for the images as a weighted sum of distances between its features. It then uses an iterative approach for learning using a suitable clustering algorithm [8]. In each iteration, clusters are obtained over the given images hierarchically such that the number of levels of similarity in the clusters is equal to that in the training samples. A comparison is then made between pairs of images in the clusters and training samples. Adjustments are made to the weights of the features in the distance function based on the error in clustering. Suitable heuristics for error computation and weight adjustment are proposed in the paper. The distance function corresponding to minimal error is returned as the output. The weights of the features in this distance function give their relative importance.

The FeaturesRank approach is evaluated with real data from two domains, nanotechnology and bioinformatics. We focus on similarity search using distinct test sets not used for training the technique. A target image is compared with images in the database. The database images are ranked in the order of their similarity with the target image based on the learned distance function. The effectiveness of the ranking is judged by domain experts. The details of our evaluation are described in the paper.

The rest of this paper is as follows. Section 2 gives the background. Section 3 explains the FeaturesRank method. Section 4 presents the evaluation. Section 5 outlines related work. Section 6 gives the conclusions.

2. Background

2.1. Images from Nanotechnology

Figure 1 depicts an example of an image from nanotechnology. It is a top view of carbon nanofibers on glass mircofiber [4] taken with a scanning electron microscope (SEM).

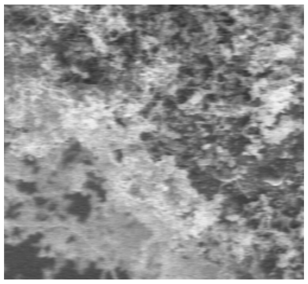

Figure 1: Carbon Nanofibers on Glass Mircofiber (SEM)

While comparing images manually, experts observe certain features of the images, such as nanoparticle size, interparticle distance and nanoparticle height [17]. Nanoparticle size refers to the dimensions of each particle in the nanostructure. Inter-particle distance is the average distance between the particles as seen in a 2-dimensional space. Nanoparticle height indicates to what extent the particles project above the surface in a cross-section [4]. These are visual features of the images. Also, domain experts take into account other factors such as the level of zooming and the nature of the cross-section (top-view, oblique view etc.). This forms image metadata.

2.2. Images from Bioinformatics

Figure 2 is an example of an image in bioinformatics. This shows the inner view of a Herb Leaf [14] taken with a transmission electron microscope (TEM).

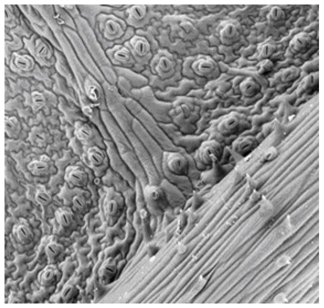

Figure 2: Herb Leaf (TEM)

In comparing such images, visual features such as pixel size and greyscale (or color) and metadata such as level of zooming and image source are applicable. Domain experts intuitively take these into account in comparing the images [14].

Thus, in order to automate image comparison for domain-specific computational analysis, it is useful to incorporate such features pertaining to visual data as well as metadata on images. This helps to simulate the reasoning of experts for effective image comparison.

3. The FeaturesRank Method

We propose a method called FeaturesRank that learns a distance function incorporating the relative importance of features in the images. Before we explain the details of the learning, the inputs to FeaturesRank are briefly described as follows.

3.1. Inputs to FeaturesRank

Experts provide correct training samples with n pairs of images made from $2n$ distinct images. Each image is identified by a unique ID. For each pair they indicate whether it is similar or different and if similar, they specify the level of similarity in training samples, LT(P) from among L total levels. This is based on the notion of correctness as per the domain. We explain this with an example. Consider that $L = 3$, i.e., we have 3 levels of similarity where 1 indicates the least similar and 3 indicates the most similar. By default, 0 indicates dissimilar. Example 1 below shows 20 distinct images organized into 10 disjoint pairs of images $P = (I_a, I_b)$ with the level of similarity identified for each pair P. As stated above, we denote this as $LT(P)$, i.e., level of similarity of pair P in training samples.

Example 1: Training Samples

P_1: (I_1,I_{16}), LT(P_1) = 2
P_2: (I_5,I_{14}), LT(P_2) = 1
P_3: (I_2,I_3), LT(P_3) = 0
P_4: (I_6,I_{18}), LT(P_4) = 1
P_5: (I_7,I_9), LT(P_5) = 0
P_6: (I_{12},I_{19}), LT(P_6) = 2
P_7: (I_{17},I_{20}), LT(P_7) = 1
P_8: (I_4,I_{11}), LT(P_8) = 3
P_9: (I_8,I_{10}), LT(P_9) = 2
P_{10}: (I_{13},I_{15}), LT(P_{10}) = 3

In addition, other inputs given by experts are features applicable to images and their individual distance functions. For a given feature *f,* its distance function is defined as Δ_f. For example, in nanotechnology, consider the feature "nanoparticle size". The distance between 2 images based on this feature is the absolute difference between the sizes of their particles. For images I_a and I_b this is given as: $\Delta_f (I_a,I_b) = |f_a - f_b|$. Likewise, individual distance functions are specified for all features.

3.2. Definition of Distance Function

Based on the given inputs, we now define the concept of a distance function for the images in order to proceed with the learning in FeaturesRank.

Distance Function for Images: For images I_a and I_b distance $\Delta (I_a,I_b)$ between them is calculated as a weighted sum of distances between their features, where the weights indicate the relative importance of the features. Thus the distance function Δ is defined as:

$\Delta = \sum_{f=1}^{F} \alpha_f \Delta_f$ where
f = index of each feature
F = total number of features
α_f = weight of feature f
Δ_f = distance between the images based on feature f

Having defined this distance function for images, we now proceed to learn the weights of the features. Note that, at the start we consider all applicable features. If a feature happens to be insignificant, its weight should finally be zero. This would be learned by FeaturesRank.

3.3. Process of Learning

The learning in FeaturesRank occurs as follows. Consider any suitable clustering algorithm [8]. The notion of distance in clustering is an initial distance function $\Delta = \sum_{f=1}^{F} \alpha_f \Delta_f$ with features *f = 1 to F* given as inputs and their weights assigned randomly. Using this distance, clustering is performed over the *2n* images in *L* levels, where *L* is the number of levels of similarity in the training samples (except the base level of zero). In order to draw an analogy with the training samples, the clustering is performed with two clusters at each level. We consider levels with two clusters each because the training samples indicate whether the pairs of images are similar or different (analogous to being in the same or different clusters) up to a certain level. We explain this with reference to Example 1.

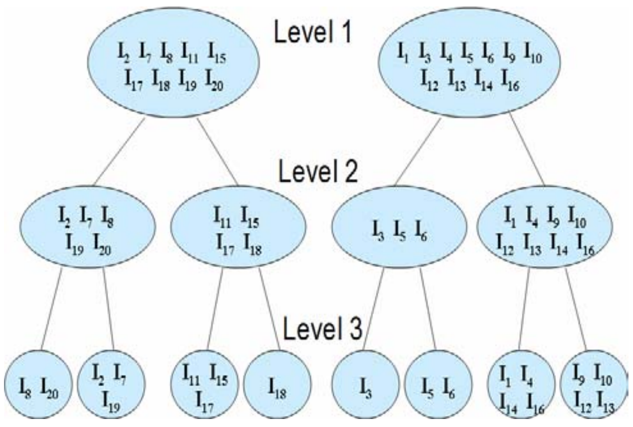

Figure 3: Clusters for Images in Example 1

Clustering would be performed in three levels here since there are three levels of similarity in the training samples. In the first level, the 20 images would be placed in two clusters. In the second level, each cluster would again be partitioned into two clusters. Thus we get four clusters at this level. In the third level, each of these four clusters would in turn be partitioned into two more clusters each, thus giving a total of eight clusters. An example of such obtained clusters is shown in Figure 3 with reference to the images in Example 1.

Note that the clustering is always performed such that the clusters at the next level are sub-clusters of those at the previous level. In other words, if a pair of images is in different clusters at a given level, it would remain in different clusters at all levels thereafter.

Ideally the obtained clusters should match the given training samples in terms of similarity between pairs of images. Thus, if for example, a pair of images is in different clusters at the very first level, ideally its level of similarity should be zero in the samples. If not, then some adjustment must be made to the distance function. Before introducing the heuristics for adjustment, we need to explain the calculation of error between the obtained clusters and the training samples. In order to calculate the error we first define the level of similarity between a pair of images in the clusters.

Level of Similarity for a Pair of Images in Clusters: For a pair of images $P = (I_a, I_b)$, its level of similarity in the clusters denoted as *LC(P)* is equal to:
0, if I_a, I_b are in different clusters at level 1
x, if I_a, I_b are in the same cluster up to level x where $x \geq 1$ and in different clusters at all levels y, $\forall y > x$

For example, with reference to Figure 3, the level of similarity of the pair of images I_1 and I_{16} in the clusters $LC(I_1,I_{16}) = 3$, since they are in the same cluster up to the third level. Likewise, $LC(I_5, I_{14}) = 1$ since they are in the same cluster only up to the first level. For the pair I_2 and I_3, $LC(I_2,I_3) = 0$, since they are in different clusters at the very first level. Given this notion of similarity, we now define the concept of error in clustering as below.

Error in Clustering: For a pair of images $P = (I_a, I_b)$, if the level of similarity in clusters is not equal to the level of similarity in the training samples, i.e., $LC(I_a,I_b) \neq LT(I_a,I_b)$ then it is considered to be an error pair. The error in clustering is defined as the ratio of the number of error pairs over the total number of pairs. If n is the total number of pairs and E is the number of error pairs, then error denoted as Φ is calculated as $\Phi = E/n$.

Thus, with reference to Example 1, (I_1,I_{16}) is an error pair, since $LC(I_1,I_{16}) = 3$, while $LT(I_1,I_{16}) = 2$. Likewise, out of the10 pairs, 6 are error pairs. Thus, error $\Phi = 6/10 = 0.6$ or 60 %.

Ideally, the error should be zero since the clusters should match the training samples perfectly. However, a perfect match may not always occur. Hence in practice an error threshold is defined as follows.

Error Threshold: The error threshold τ is defined as the fraction of the total number of pairs allowed to be error pairs for the clustering to be considered correct. Thus, if $\Phi \leq \tau$ then the clustering is correct.

With reference to Example 1, consider that the error threshold is given as $\tau = 0.1$, i.e., 10% error is allowed. Since the error in this example is calculated as $\Phi = 0.6$, we get $\Phi > \tau$, i.e., the error is above the given threshold. Thus the clustering is not correct in this example.

In order to rectify the error, adjustments need to be made to the distance function used for clustering. The heuristic for adjustment is explained below.

Consider that the level of similarity of a particular error pair (I_a,I_b) in the clusters is greater than that in the training samples, i.e., $LC(I_a,I_b) > LT(I_a,I_b)$. This implies that the images in the pair (I_a,I_b) have been considered closer to each other in the clusters obtained with our distance function than they should be in reality (as indicated by the training samples). An example of such a pair is (I_1,I_{16}). In order to rectify this, the images need to be pushed further apart from each other. We propose to do this by increasing the weights of one or more features in the distance function used for clustering.

We define two concepts *Step* and *Blame*, analogous to step size and blame assignment in machine learning.

Definition of Step: The *step* $S(I_a, I_b)$ due to a pair $P = (I_a, I_b)$ is the difference between the correct level of similarity of that pair in the training samples and its obtained level of similarity in the clusters. This is calculated as:

$$S(I_a,I_b) = |LT(I_a,I_b) - LC(I_a,I_b)|.$$

Definition of Blame: The *blame* $B_f(I_a, I_b)$ due to a feature f and pair $P = (I_a, I_b)$ is the ratio of the distance between the images of pair P due to feature f over the total distance between the images. This is calculated as:

$$B_f(I_a,I_b) = \Delta_f(I_a,I_b)/\Delta(I_a,I_b).$$

Given these two definitions, we propose that if the weight of the given feature is α_f, its new weight should be $\alpha_f + S(I_a,I_b) \times B_f(I_a,I_b)$ in order to rectify the error caused by the given feature in the given pair.

Conversely consider an error pair where we have $LC(I_a,I_b) < LT(I_a,I_b)$. Applying the reverse argument, the new weight of each feature should be $\alpha_f - S(I_a,I_b) \times B_f(I_a,I_b)$ due to such a pair.

Thus, in order to take into account the effect of all error pairs, we propose a heuristic for adjustment called the *Feature Weight Heuristic*. This is defined below.

Feature Weight Heuristic: For each error pair, the new weight of each feature is given as:

$$\alpha_f + S(I_a,I_b) \times B_f(I_a,I_b) \text{ if } LC(I_a,I_b) > LT(I_a,I_b) \text{ and}$$
$$\alpha_f - S(I_a,I_b) \times B_f(I_a,I_b) \text{ if } LC(I_a,I_b) < LT(I_a,I_b) \text{ where}$$
$$B_f(I_a,I_b) = \Delta_f(I_a,I_b)/\Delta(I_a,I_b) \text{ and}$$
$$S(I_a,I_b) = |LT(I_a,I_b) - LC(I_a,I_b)|$$

The justification of the feature weight heuristic is as follows. This heuristic embodies in spirit the reasoning of the experts [4, 14]. They reason in terms of levels of similarity in making image comparisons instead of having precise numbers for feature weights at the top of their mind. Therefore we enable them to provide the training data to our method accordingly. Hence the adjustment in the heuristic is made by scaling up or down the weights based on the difference between the actual level of similarity in the training samples and the obtained level of similarity in the clusters. The *step* denotes the difference between the levels of similarity. The *blame* of each feature helps to calculate the extent to which the given feature is responsible for the difference. The subjective reasoning of the experts is thus mapped into an objective measure for feature weight adjustment.

Likewise, after considering the effect of each error pair using this heuristic, the adjusted distance function thus obtained is likely to rectify the error in clustering. This adjusted distance function is then used for another iteration of clustering.

The process of clustering, calculating error and making adjustments is repeated until the error is below a given threshold or the maximum number of allowable iterations is reached. The distance function that corresponds to the lowest error is then returned as the output of FeaturesRank. This contains the features ranked in the order appropriate in the respective domain.

3.4. The FeaturesRank Algorithm

Based on the above discussion, the FeaturesRank algorithm we propose is outlined below.

```
Algorithm: FeaturesRank
• Given:
    o Training samples with n pairs of 2n distinct objects
    o Total number of similarity levels L
    o Level of similarity in training samples LT(P) identified for each pair P
    o F features $f_1$ through $f_F$ applicable to objects
    o Error threshold $\tau$
• Learn: Distance function $\Delta$ with relative importance of features
• Process:
    1. Define $\Delta = \sum_{f=1}^{F} \alpha_f \Delta_f$ where $\alpha_f$ is weight of feature f
    2. Consider any clustering algorithm
    3. Let number of clusters = 2 throughout,, initialize number of error pairs E = 0
        a. For i = 1 to L
            • Cluster 2n objects in L levels using distance $\Delta = \sum_{f=1}^{F} \alpha_f \Delta_f$
        b. For P= 1 to n pairs in training samples
        c. Calculate LC(P) in clusters
        d. If $LC(P) \neq LT(P)$ then E = E + 1
        e. Calculate error $\Phi$ = E/n
        f. If $\Phi \leq \tau$ or max iterations reached go to step 4
        g. Apply Feature Weight Heuristic to get new $\Delta_f$ and go to 3(a)
    4. Return current $\Delta = \sum_{f=1}^{F} \alpha_f \Delta_f$ as learned distance function
```

The Feature Weight Heuristic proposed in the FeaturesRank algorithm is not guaranteed to provide convergence in theory. However, in practice, we have obtained consistent convergence to error below threshold. This is elaborated in the section on evaluation.

4. Experimental Evaluation

The FeaturesRank approach has been evaluated with real data from nanotechnology and bioinformatics. A summary of our evaluation is presented below.

4.1. Performance Evaluation over Training Data

Training samples in the form of pairs of images are provided to us by domain experts from nanotechnology and bioinformatics respectively. We have 30 pairs of images from nanotechnology and 20 pairs from bioinformatics. Levels of similarity in training samples are identified for all the pairs of images. For the given nanotechnology data, similarity levels are identified from among three total levels (i.e., L=3). In bioinformatics, there are two total levels (i.e., L=2). This is in addition to the base level of zero implying dissimilar in both the domains. Features of these images are also provided as applicable. Initial weights of these features are either randomly assigned or equal weights are used. We conduct experiments with both equal and random weights. We consider error threshold between $\tau = 0.1$ and $\tau = 0.05$. The maximum number of iterations is set to 1000. Clustering seeds are altered for randomization.

For each data set, we show two combinations of random initial weights and one combination of equal initial weights. We depict the results for the two extremes of $\tau = 0.1$ and $\tau = 0.05$. We record the training behavior in terms of error versus iterations for each experiment.

4.1.1. Nanotechnology Data. Figures 4 through 6 show the training behavior for the data set from nanotechnology. In these charts, the thick line shows the results with 10% error threshold and the thin line corresponds to the experiments with 5% error threshold.

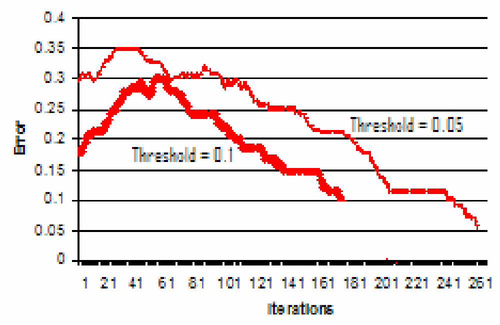

Figure 4: Nanotechnology Data, Random Weights

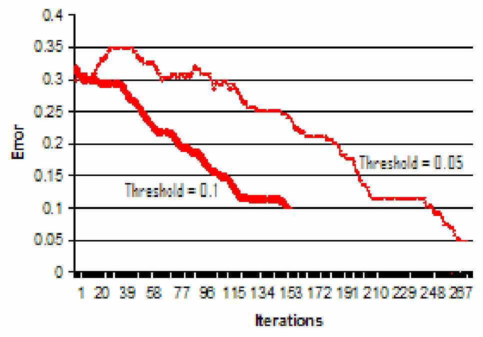

Figure 5: Nanotechnology Data, Equal Weights

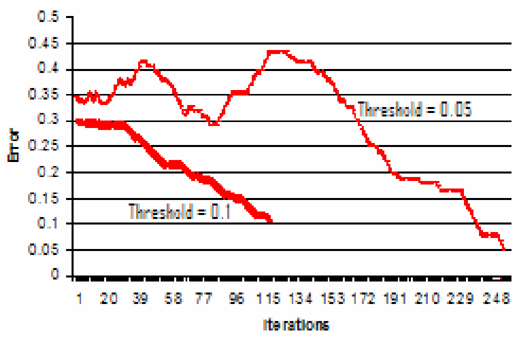

Figure 6: Nanotechnology Data, Another Combination of Random Weights

It is observed that convergence to error below threshold occurs for all the experiments in less than 300 iterations. The experiments with error threshold of 5% take longer to converge than those with 10% threshold. We do not observe a huge difference in the behavior of experiments with random and equal initial weights in terms of number of iterations for convergence.

4.1.2. Bioinformatics Data. The training behavior for the data set from bioinformatics is depicted in Figures 7 through 9. In these charts also, the thick and thin lines depict the experiments with error thresholds of 10% and 5% respectively.

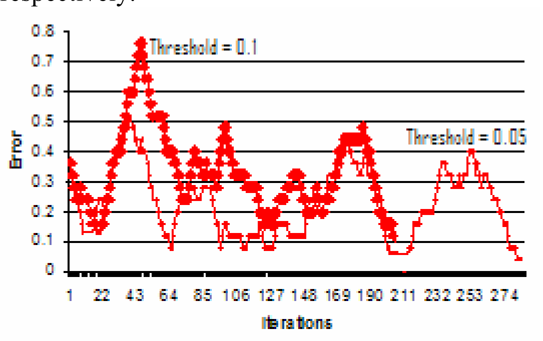

Figure 7: Bioinformatics Data, Random Weights

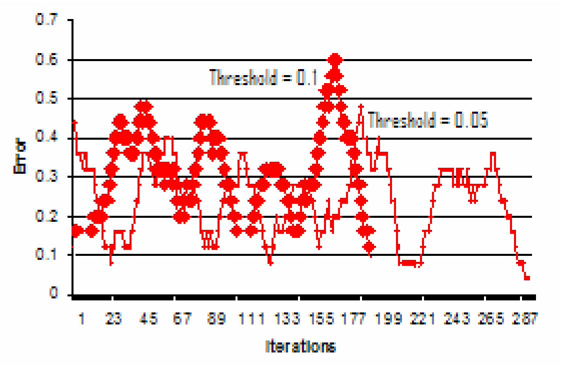

Figure 8: Bioinformatics Data, Equal Weights

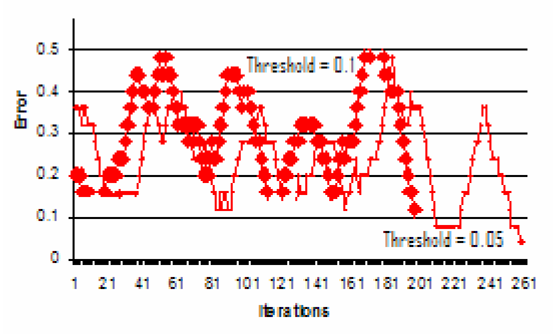

Figure 9: Bioinformatics Data, Another Combination of Random Weights

It is seen from these charts that the error in the bioinformatics data set fluctuates more than in the nanotechnology data set. Possible reasons for this could be that fewer images were used as training samples for learning and fewer levels of similarity were used. Other observations were similar to those in the nanotechnology data sets.

In both data sets, convergence to error below threshold occurs for all the experiments we have conducted so far. Many are not shown here due to space limitations.

4.2. User Evaluation over Test Data

We conduct user evaluation in the context of similarity search for both the nanotechnology and bioinformatics data sets. The purpose of this evaluation is to assess the effectiveness of the distance function output by FeaturesRank that incorporates the relative importance of features in the image data. We use test data distinct from that used for training the technique. The goal is to rank images in the test data set in the order of their similarity with a target image. The results of the ranking are verified by domain experts. Based on the outcome of our experiments with different target images, experts provide a subjective assessment of the ranked results. We present a few excerpts from our evaluation.

4.2.1. Similarity Search with Nanotechnology Data. Figure 8 shows an example of a target image from the field of nanotechnology. Using our learned distance function from FeaturesRank, this is compared with all the images in the test data set. The top 4 images that are most similar to the target image are returned, ranked in their order of similarity as shown in Figure 9

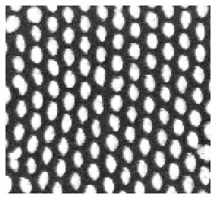

Figure 8: Target Image in Nanotechnology Data

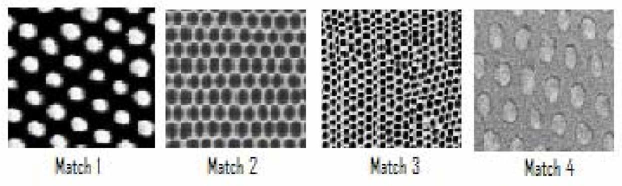

Figure 9: Top 4 matches in Nanotechnology Data

The experts are asked to assess the effectiveness of the ranking. They conclude that the top 4 ranked answers in this case are accurate. Likewise, on conducting several experiments with test data in nanotechnology, the experts indicate that the overall performance is good.

4.2.2. Similarity Search with Bioinformatics Data. Figure 10 illustrates a sample target image from bioinformatics. Using the output of FeaturesRank, this is compared with the images in the given test data set to find the top 4 closest matches ranked in their order of similarity. The results of the ranking appear in Figure 11.

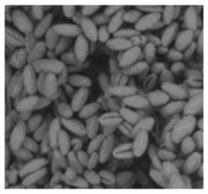

Figure 10: Target Image in Bioinformatics Data

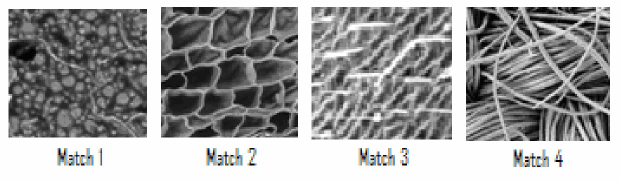

Figure 11: Top 4 matches in Bioinformatics Data

This ranking is considered to be effective as verified by the domain experts. Based on the outcomes of all the experiments conducted with bioinformatics data, we find that the experts are satisfied with the ranked results for similarity search in image retrieval.

5. Related Work

The FastMap algorithm [5] maps objects in multimedia databases to points in a user-defined k-dimensional space preserving similarities in the original objects. This aids efficient retrieval with spatial access methods, visualization and data mining. In [12], feature-based and correlation based methods are combined to derive a trainable similarity measure to classify images. In [1], probabilistic measures are proposed for similarity search over images. These are robust to normalization and outperform geometric measures. One way in which our work differs from these is that we take into account domain-specific metadata on the images in addition to visual features. Moreover, our work is orthogonal to the literature, in being able to combine various individual distance functions corresponding to different aspects of images in order to learn their relative importance.

Keim et al. [10] overview various distance types applicable to similarity search in multimedia databases. However no single distance function combining such distance types is proposed. Hinneburg et al. [7] propose a learning method to find the relative importance of dimensions for n-dimensional objects. Their goal is to learn the minimal number of dimensions that give the same result of a query in the original and reduced space. They do not however semantically rank the dimensions based on the respective features captured.

Our earlier work [16] involves learning distance metrics for graphs plotting scientific functions. However, the assumption is that true clusters of such graphs are provided as a training set. It is however not always feasible for experts to provide such large volumes of training data in the form of clusters. Moreover the issues is comparing images are even more challenging than those with graphs. Also in dealing with the images here, ranking them based on levels of similarity is meaningful in the context of the given domains.

In [3], Das et al. perform automated selection and ranking of attributes in query results. They propose a hybrid Split-Plane approach that combines top attributes from variants of attribute selection approaches. In [2], Chaudhuri et al. rank the results of database queries based on probabilistic information retrieval models. They propose a ranking function that depends on the global importance of unspecified attribute values as well as the strengths of correlations between specified and unspecified attribute values. The work in ranking problems such as these is more of a post-processing nature in terms of presenting results to users. Our work deals primarily with data preprocessing issues that involve learning similarity measures in order to proceed with the analysis of the data. Moreover we focus on image comparison and its targeted applications.

There has been work in capturing the subjective reasoning of users to learn objective measures for query processing, image retrieval and graphical user interface design. The MindReader system [9] guesses the distance function in the mind of the users based on combining several examples given by users and their relative importance. In [13], they propose a human computer interaction approach to perform content based retrieval of images. The users' queries and subjective perceptions are used to learn feature weights based on relevance feedback. In [15], suitable cluster representatives are designed for targeted applications based on the respective users' subjective interests learned through an objective encoding. The proposed encoding is analogous through the Minimum Description Length principle. Our FeaturesRank approach falls in the same general category of such techniques. Heuristics in FeaturesRank are defined as appropriate for the given problem.

Wavelets [8, 19] are often used for image processing, in order to compare and rank images. However, the wavelet coefficients need to be computed each time the image comparison is made which is computationally expensive. FeaturesRank on the other incurs a one-time cost of learning the relative importance of the features. The learned distance function can then be used for image comparison without further expense.

6. Conclusions

This paper describes our FeaturesRank approach to learn a distance function incorporating the relative importance of features in images. The learned distance function is assessed in the context of ranking images in similarity search. Real data from the domains of nanotechnology and bioinformatics has been used for learning and evaluation. User studies conducted with the help of domain experts confirm that the output of FeaturesRank is effective as per their needs. Ongoing work involves conducting formal user surveys over larger volumes of data; evaluating FeaturesRank in other applications; and performing comparative studies with state-of-the-art.

Acknowledgments

The authors acknowledge the support of the the Nanofabrication and Nanomanufacturing group in the Department of Mechanical Engineering at Worcester Polytechnic Institute. We also thank the Bioinformatics group in the Department of Engineering and Technology at Virginia State University.

References

[1] S. Aksoy and R. M. Haralick, "Probabilistic vs. Geometric Similarity Measures for Image Retrieval", *IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 2, pp. 357- 362.

[2] S. Chaudhuri., G. Das, V. Hristidis and G. Weikum, "Probabilistic Ranking of Database Query Results", *VLDB*, Aug 2004, pp. 888- 899.

[3] G. Das, V. Hristidis, N. Kapoor and S. Sudarshan, "Ordering the Attributes of Query Results" *SIGMOD*, Jun 2006, pp. 395- 406.

[4] S, Dougherty, J. Liang, J. and G. Pins, *Precision Nanostructure Fabrication for the Investigation of Cell Substrate Interactions*. Technical Report, Worcester Polytechnic Institute, Worcester, MA, Jun 2006.

[5] C. Faloutsos and K. I. Lin, "FastMap: A Fast Algorithm for Indexing, Data Mining and Visualization of Traditional and Multimedia Datasets", *SIGMOD* Record, 1995, Vol. 24, No. 2, pp. 163 - 174.

[6] U. Fayyad, D. Haussler and P. Storoltz "Mining Scientific Data", *Communications of the ACM*, 1996, Vol. 39, No. 11, pp. 51 - 57.

[7] A. Hinneburg, C. Aggarwal, and D. Keim, "What is the Nearest Neighbor in High Dimensional Spaces", *VLDB,* Aug 2000, pp. 506 - 515.

[8] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, CA, 2001.

[9] Y. Ishikawa, R. Subramanya and C. Faloutsos, "MindReader: Querying Databases through Multiple Examples", *VLDB,* Aug 1998, pp. 218 – 227.

[10] D. Keim and B. Bustos, "Similarity Search in Multimedia Databases", *ICDE Tutorial*, Mar 2004, pp. 873 - 874.

[11] Y. Reich and S. Barai, "Evaluating Machine Learning Models for Engineering Problems", *Artificial Intelligence in Engineering,* 1999, Vol. 13, pp. 257 - 272.

[12] P. Paclik, J. Novovicova, and R.P.W Duin, "A Trainable Similarity Measure for Image Classificarion", *ICPR*, Aug 2005, pp. 391 - 394.

[13] Y. Rui, T. Huang and S. Mehrotra, "Relevance Feedback Techniques in Interactive Content Based Image Retrieval", *SPIE,* Jan 1998, pp. 25-36.

[14] E. Sheybani and A. Varde, *Issues in Bioinformatics Image Processing,* Technical Report, Virginia State University, Petersburg, VA, Oct 2 006.

[15] A. Varde, E. Rundensteiner, C. Ruiz, D. Brown, M. Maniruzzaman and R. Sisson Jr., "Effectiveness of Domain-Specific Cluster Representatives for Graphical Plots", *SIGMOD IQIS,* June 2006, pp. 24 – 29.

[16] A. Varde, E. Rundensteiner, C. Ruiz, M. Maniruzzaman and R. Sisson Jr., "LearnMet: Learning Domain-Specific Distance Metrics for Plots of Scientific Functions", *MTAP Journal Special Issue on Multimedia Data Mining*, 2006.

[17] A. Varde, J. Liang, E. Rundensteiner and R. Sisson Jr., "Mining Images of Material Nanostructure Data", *ICDCIT,* Dec 2006, pp. 403 – 413.

[18] M. Ward, "XMDV Tool: Integrating Multiple Methods for Visualizing Multivariate Data". *Visualization*, Oct 1994, pp. 326 – 333.

[19] J. Z. Wang, G. Wiederhold, O. Firschein and S. X. Wei, "Content-Based Image Indexing and Searching Using Daubechies' Wavelets", *International Journal of Digital Libraries,* 1997, Vol. 1, pp. 311-328.