Learning Semantics-Preserving Distance Metrics for Clustering Graphical Data

Aparna S. Varde[1,2] Elke A. Rundensteiner[1] Carolina Ruiz[1] Mohammed Maniruzzaman[2]
Richard D. Sisson Jr.[2,3]
[1] Department of Computer Science
[2] Center for Heat Treating Excellence
[3] Department of Mechanical Engineering
Worcester Polytechnic Institute (WPI)
Worcester, MA 01609. USA
(aparna | rundenst | ruiz | maniruzz | sisson)@wpi.edu

ABSTRACT

In mining graphical data the default Euclidean distance is often used as a notion of similarity. However this does not adequately capture semantics in our targeted domains, having graphical representations depicting results of scientific experiments. It is seldom known a-priori what other distance metric best preserves semantics. This motivates the need to learn such a metric. A technique called LearnMet is proposed here to learn a domain-specific distance metric for graphical representations. Input to LearnMet is a training set of correct clusters of such graphs. LearnMet iteratively compares these correct clusters with those obtained from an arbitrary but fixed clustering algorithm. In the first iteration a guessed metric is used for clustering. This metric is then refined using the error between the obtained and correct clusters until the error is below a given threshold. LearnMet is evaluated rigorously in the Heat Treating domain which motivated this research. Clusters obtained using the learned metric and clusters obtained using Euclidean distance are both compared against the correct clusters over a separate test set. Our results show that the learned metric provides better clusters.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning – *Parameter learning.*

General Terms

Algorithms, Experimentation.

Keywords

Distance metric, clustering, semantic graphical mining.

1. INTRODUCTION

Experimental results in scientific domains are often represented graphically. Such graphs depict the functional behavior of process parameters hence incorporating semantics. In this paper, we use the term "graph" to mean such a graphical representation. In mining graphs with techniques such as clustering [9] the measure for comparison is typically Euclidean distance [see e.g. 7] which often poses problems. For example, in the domain of Heat Treating of Materials which motivated this work, graphs are used to plot the heat transfer versus temperature during the rapid cooling of a material [see e.g. 3]. Critical regions on these graphs represent significant physical phenomena in the domain. Any algorithm (based for instance on Euclidean distance) that considers two graphs as similar (relative to other graphs) although their critical regions differ is considered semantically incorrect [3,14]. Likewise in several scientific domains, there could be significant features on graphs. Knowledge of these features and their relative importance may at best be available in a subjective form, but not as a metric. This motivates the development of a technique to learn distance metrics that capture the semantics of the graphs.

Hinneburg et al [6] propose a learning method to find the relative importance of dimensions for n-dimensional objects. However, their focus is on dimensionality reduction and not on domain semantics. In [16] they learn which type of position-based distance is applicable for the given data starting from the formula of Mahalanobis distance. However they do not deal with graphical data and semantics. Keim et al. [8] overview various distance types for similarity search over multimedia databases. However no single distance measure encompassing several types is proposed. Linear regression [1] and neural networks [2] could possibly be used for learning a domain-specific distance metric for graphs. However these techniques do not achieve accuracy acceptable in our targeted domains [14]. Genetic algorithms [5] if used for feature selection in graphs also give the problem of insufficient accuracy. This is due to lack of domain knowledge [14]. Fourier transforms [4] if used to represent the graphs do not preserve the critical regions in the domain due to the nature of the transform [14]. Hence they are not accurate enough in capturing semantics. Accuracy in this context is measured by evaluating the effectiveness of a given metric in mining unseen data [14].

We propose an approach called LearnMet to learn a distance metric for graphs incorporating domain semantics. The input to LearnMet is a training set with correct (i.e., given by domain experts) clusters of graphs over a subset of the experimental data. The steps of our approach are: (1) guess initial metric guided by domain knowledge; (2) use that metric for clustering with an arbitrary but fixed clustering algorithm; (3) evaluate accuracy of obtained clusters by comparing them with correct clusters; (4) adjust metric based on error between obtained and correct clusters, if error is below threshold or if execution times out then terminate and go to step (5), else go to step (2); and (5) once terminated, output the metric giving error below threshold, or minimum error so far as the learned metric. LearnMet is evaluated using a *distinct* test set of correct clusters of graphs provided by experts. The learned metric is used to cluster the graphs in the test set. The obtained clusters are compared with correct clusters in the test set. The closer the obtained clusters match the correct clusters, the lower the error. The clusters obtained using the default notion of Euclidean distance, are also compared with the correct clusters. The difference in the error denotes the effectiveness of the learned metric. The LearnMet metric consistently outperformed the Euclidean metric in our experiments. LearnMet is also evaluated by integrating it with the AutoDomainMine system designed for computational estimation of process parameters in scientific experiments [13]. This is explained in Section 4.

2. DISTANCE METRICS

We describe distance metric types relevant to our domains. Let *A* $(A_1,A_2...A_n)$ and *B* $(B_1,B_2...B_n)$ be two *n*-dimensional objects.

Position-based Distances: They refer to absolute position of objects [see e.g. 7]. Examples are:

Euclidean Distance: As-the-crow-flies distance between objects.

$$D_{Euclidean}(A,B)=\sqrt{\sum_{i=1}^{n}(A_i-B_i)^2}$$

Manhattan Distance: City-block distance between objects.

$$D_{Manhat\tan}(A,B)=\sum_{i=1}^{n}|A_i-B_i|$$

Statistical Distances: This refers to distances based on statistical features [see e.g. 11]. Examples are:

Mean Distance: Distance between mean values of the objects:

$$D_{Mean}(A,B)=|Mean(A)-Mean(B)|$$

Maximum Distance: Distance between maximum values:

$$D_{Max}(A,B)=|Max(A)-Max(B)|$$

In addition to these, we introduce the concept of *Critical Distance* for graphs as applicable to our targeted domains [14].

Critical Distances: *Given graphs A and B, a Critical Distance is a distance metric between critical regions of A and B where a critical region represents the occurrence of a significant physical phenomenon. They are calculated in a domain-specific manner.*

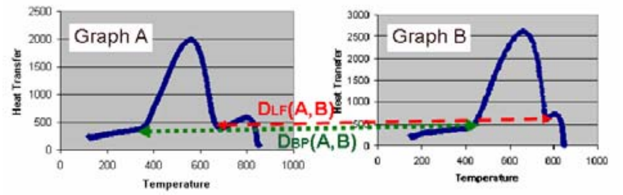


Figure 1: Critical Distance Examples

One example of critical distance as shown in Figure 1 is the Leidenfrost distance [3,14] in Heat Treating. This is for Leidenfrost Points [see e.g. 3] which denote the breaking of the vapor blanket around a part in heat treatment. It is calculated as $D_{LF}(A,B)=\sqrt{(A_{TLF}-B_{TLF})^2+(A_{hLF}-B_{hLF})^2}$ where T_{LF} is the temperature at Leidenfrost Point and h_{LF} is the heat transfer coefficient at that point [20]. Another critical distance is the Boiling Point distance $D_{BP}(A,B)$ [3,14]. This is the distance between points on the graphs corresponding to the Boiling Points [see e.g. 3] of the respective cooling media in heat treatment. $D_{BP}(A,B)=\sqrt{(A_{TBP}-B_{TBP})^2+(A_{hBP}-B_{hBP})^2}$ where T_{BP} is temperature and h_{BP} is heat transfer coefficient at Boiling Point.

3. THE LEARNMET STRATEGY

To give details of LearnMet, a distance metric is first defined.

<u>Distance Metric in LearnMet</u>*: A LearnMet distance metric D is a weighted sum of components, where each component can be a position-based, a statistical, or a critical distance metric. The weight of each component is a numerical value indicating its relative importance in the domain.*

Thus a LearnMet distance metric is of the form $D=w_1Dc_1+.....+w_mDc_m$ where each Dc_i is a component, w_i is its weight, and *m* is number of components. As required in our application domain, *D* should be a metric so that clustering algorithms requiring the notion of similarity to be a distance metric can be used; indexing structures such as B+ trees for metrics can be applied; and pruning in similarity search can be performed using triangle inequality. Conditions for *D* to be a metric are stated as a theorem below.

<u>Theorem 1:</u> If each component Dc_i *is a distance metric and each weight* $w_i\geq 0$ *then* $D=\sum_{i=1}^{m}w_iDc_i$ *is a distance metric, i.e., it satisfies the metric properties.*

The proof of this theorem is straightforward and can be found in [14]. In our targeted domains, conditions in Theorem 1 are satisfied. Since our graphs involve interval-scaled variables and the fundamental distance types applicable to these are metrics [see e.g. 7], this is sufficient to say that each Dc_i is a metric. Also, we consider only non-negative weights.

The LearnMet steps are discussed in the subsections below.

3.1 Initial Metric Step

Domain experts are asked to identify components (i.e., distance metrics) applicable to the graphs that will serve as building blocks for the learning of a new metric. If the experts know the relative importance of the components, this information is used to assign initial weights. An *Initial Weight Heuristic* is proposed.

<u>Initial Weight Heuristic:</u> *Assign initial weights to the components in the LearnMet distance metric based on the relative importance of the components on the graphs in the domain.*

If the components' relative importance is unknown, random weights are assigned. Initial weights are on a scale of 1 to 10.

```
1: INITIAL METRIC STEP
• GIVEN: DOMAIN EXPERT INPUT ON DISTANCE TYPES
• FOR EACH DISTANCE TYPE ASSIGN A COMPONENT TO "D"
• IF RELATIVE IMPORTANCE OF COMPONENTS IS AVAILABLE
      THEN USE "INITIAL WEIGHT HEURISTIC"
• ELSE ASSIGN RANDOM WEIGHTS TO COMPONENTS
```

3.2 Clustering Step

Using *D* as the distance metric, *k* clusters are constructed using an arbitrary but fixed clustering algorithm (e.g., k-means [10]), where *k* is the number of clusters in the training set.

```
2: CLUSTERING STEP
• GIVEN: A TRAINING SET CONSISTING OF A COLLECTION OF
      GRAPHS AND A CORRECT k-CLUSTERING OF THEM
• SELECT AN ARBITRARY BUT FIXED CLUSTERING
      ALGORITHM
• SET NUMBER OF CLUSTERS TO k ( CONSTANT)
• CLUSTER GRAPHS USING $D = w_1 Dc_1 + ... + w_m Dc_m$
```

3.3 Cluster Evaluation Step

The clusters obtained from the algorithm, i.e., the "predicted" clusters, are evaluated by comparing them with correct clusters in the training set, i.e., the "actual" clusters. An example of predicted and actual clusters is shown in Figure 2.

Ideally, the predicted clusters should match the actual clusters. Any difference between predicted and actual clusters is considered an error. To compute this error, we consider pairs of graphs and introduce the following notation.

<u>True/False Positive/Negative Pairs of Graphs:</u> *Given a pair of graphs I and J, we say that:*

- *(I,J) is a* ***True Positive (TP) pair*** *if I and J are in the same actual cluster and in the same predicted cluster.*
- *(I,J) is a* ***True Negative (TN) pair*** *if I and J are in different actual clusters and in different predicted clusters.*
- *(I,J) is a* ***False Positive (FP) pair*** *if I and J are in different actual clusters but in the same predicted cluster.*
- *(I,J) is a* ***False Negative (FN) pair*** *if I and J are in the same actual cluster but in different predicted clusters.*

Figure 2 includes examples of each of these kinds of pairs: *(g_1,g_2)* is a true positive pair; *(g_2,g_3) is* a true negative pair; *(g_3,g_4)* is a false positive pair; and *(g_4,g_6)* is a false negative pair. The error measure of interest to us is failure rate which is defined below.

<u>Success and Failure Rates:</u> *Let TP, TN, FP and FN denote the number of true positive, true negative, false positive and false negative pairs respectively. Also let SR denote the Success Rate and FR = (1 – SR) denote the Failure Rate, as defined below:*

$$SR = \frac{TP + TN}{TP + FP + TN + FN} \qquad \therefore FR = \frac{FP + FN}{TP + FP + TN + FN}$$

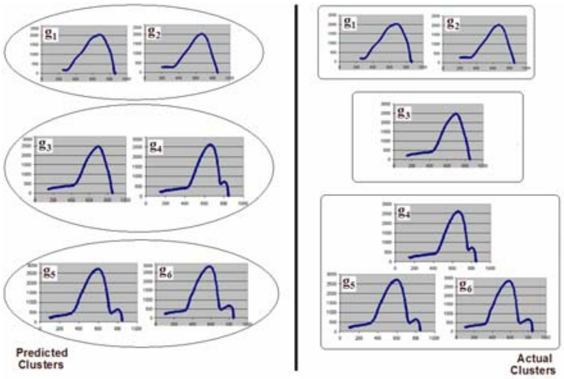


Figure 2: Predicted and Actual Clusters

In our domain, false positives and false negatives are equally undesirable. Hence, our definition of failure rate weighs them equally.

Given a number G of graphs in the training set, the total number of pairs of graphs is ${}^{G}C_2 = G!/(2!(G-2)!)$. Thus, for 25 graphs there are 300 pairs, for 50 graphs, 1225 pairs, etc. We define an *epoch* in LearnMet as one run of all its steps. That is, a complete training cycle.

<u>*Overfitting:*</u> To avoid overfitting in LearnMet, we use an approach analogous to incremental gradient descent [2, 14]. Instead of using all pairs of graphs for evaluation, a subset of pairs is used called *ppe* or *pairs per epoch*. In each epoch, a distinct combination of pairs is used for evaluation and weight adjustments. Thus there is enough randomization in every epoch. If ppe = 25, then we have a total of ${}^{300}C_{25} = 1.95 \times 10^{36}$ *distinct* pairs for learning [11, 14]. Thus in each epoch 25 *distinct* pairs can be used. This still gives a large number of epochs with distinct pairs for learning. This incremental approach reduces the time complexity of the algorithm and helps avoid overfitting. Determining the best *ppe* value is an optimization problem. Also in LearnMet, the random seed is altered in the clustering algorithm in different epochs as an additional method to avoid overfitting.

Ideally, the error i.e., failure rate in an epoch should be zero. However, in practice a *domain-specific error threshold "t"* is used.

<u>Error Threshold:</u> *A domain-specific error threshold "t" is the extent of error allowed per epoch in the domain, where error is measured by failure rate.*

<u>Distance between a Pair of Graphs:</u> *The distance D(g_a,g_b) between a pair of graphs g_a and g_b is the weighted sum of components in the graphs using metric D.*

Thus, $D(g_a, g_b) = w_1 Dc_1(g_a, g_b) + ... + w_m Dc_m(g_a, g_b)$

Given this, consider *FN* pairs, e.g., *(g_4,g_5)* and *(g_4,g_6)*. These pairs are in the same actual cluster. However they are predicted to be in different clusters. Since predicted clusters are obtained with the metric *D*, the (average) distance *D(g_a,g_b)* for these pairs is greater than it should be. Conversely, for *FP* pairs in different actual, same predicted clusters, e.g., *(g_3,g_4),* the (average) distance *D(g_a,g_b)* is smaller than it should be.

Average FN and FP Distances D_FN and D_FP:

$$D\_FN = (1/FN)\sum_{i=1}^{FN} D(g_a, g_b) \text{ where } (g_a,g_b) \text{ denotes FN pairs.}$$

$$D\_FP = (1/FP)\sum_{i=1}^{FP} D(g_a, g_b) \text{ where } (g_a,g_b) \text{ denotes FP pairs.}$$

```
3: CLUSTER EVALUATION STEP
  • SET "ppe" TO THE DESIRED NUMBER OF PAIRS OF GRAPHS
        FROM THE TRAINING DATASET TO BE
        CONSIDERED IN AN EPOCH
  • RANDOMLY SELECT ppe PAIRS OF GRAPHS
  • SET THE ERROR THRESHOLD "t"
  • IDENTIFY THE "TP", "TN", "FP", " FN" FROM " ppe" PAIRS
  • CALCULATE FAILURE RATE "FR"
  • IF (FR < t) THEN RETURN "CLUSTERING IS ACCURATE"
  • ELSE CALCULATE D_FN,, D_FP
```

3.4 Weight Adjustment Step

If the result of the evaluation does not indicate that the clustering is accurate, then the distances *D_FN* and *D_FP* are used to make weight adjustments to reduce the error in clustering. Consider the error in *FN* pairs. To reduce the average error *D_FN*, the weights of one or more components in the metric used to calculate the distance in the present epoch is decreased. For this we propose the *FN Heuristic.*

FN Heuristic: Decrease the weights of the components in the metric D in proportion to their contributions to the distance D_FN. That is, for each component:

$$w_i' = w_i - \frac{D\_FNc_i}{D\_FN} \text{ where } D\_FNc_i = (1/FN)\sum_{i=1}^{FN} Dc_i(g_a, g_b)$$

Conversely, consider *FP* pairs. To reduce their error, we increase *D_FP*. This is done by increasing the weights of one or more components in the metric using the *FP Heuristic.*

FP Heuristic: Increase the weights of the components in the metric D in proportion to their contributions to the distance D_FP. That is, for each component:

$$w_i'' = w_i + \frac{D\_FPc_i}{D\_FP} \text{ where } D\_FPc_i = (1/FP)\sum_{i-1}^{FP} Dc_i(g_a, g_b)$$

Combining these two adjustments:

Weight Adjustment Heuristic:

$$w_i''' = \min(0, w_i - FN(\frac{D\_FNc_i}{D\_FN}) + FP(\frac{D\_FPc_i}{D\_FP})).$$

Thus $D''' = w_1'''Dc_1 + ... + w_m'''Dc_m$

The new metric *D'''* obtained after weight adjustments is likely to minimize the error due to the *FN* and *FP* type pairs. If the weight of a component becomes negative, it is converted to zero as we consider only non-negative weights. Clustering is done with this new metric. If the resulting error is below the threshold, then a confirmatory test using the same metric to cluster for 2 more epochs is performed, and then this step is complete.

```
4: WEIGHT ADJUSTMENT STEP
  • IF CLUSTERING IS ACCURATE OR MAX EPOCHS REACHED
        THEN GO TO "5: FINAL METRIC STEP"
  • ELSE APPLY WEIGHT ADJUSTMENT HEURISTIC TO GET D'''
  • GO TO "2: CLUSTERING STEP"
```

3.5 Final Metric Step

If the weight adjustment terminates because the error is below the threshold then the metric in the last epoch is considered accurate and it is returned as output. However if termination occurs because the maximum number of epochs is reached, then the most reasonable metric to be output is the one corresponding to the epoch with the minimum error among all epochs.

```
5: FINAL METRIC STEP
  • IF (FR < t) THEN RETURN METRIC D
  • ELSE FIND EPOCH WITH MINIMUM Failure Rate
        RETURN CORRESPONDING METRIC D
```

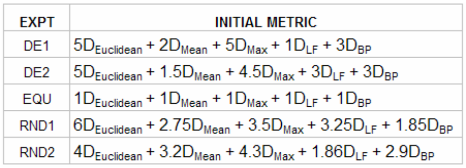
Convergence: LearnMet is not guaranteed to converge or to yield an optimal distance metric. However, thorough experimental evaluation in our application domain has shown consistent convergence to errors below the required threshold.

4. EXPERIMENTAL EVALUATION

4.1 Evaluation of LearnMet .

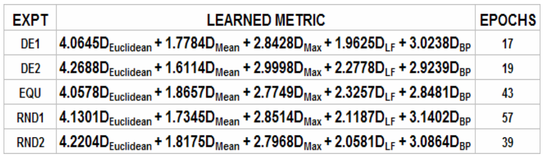
Experimental Parameters: A training set of 300 pairs of graphs in Heat Treating is obtained from correct clusters of 25 graphs given by experts. A distinct test set of 300 pairs of graphs is derived from 25 graphs given by experts. We select *"ppe = 25"* which yields ${}^{300}C_{25} = 1.95 \times 10^{36}$ total distinct combinations of pairs. Thus 25 distinct pairs are used in each epoch. Experts give an error threshold of 10%, i.e., 0.1 for estimation. We use the same threshold for clustering. Initial components in the metric are given by experts. Two distinct assignments of initial weights are given by two different experts [14]. The corresponding two metrics are denoted by *DE1* and *DE2* respectively. A third initial metric *EQU* is obtained by assigning equal weights to all components. Several experiments are run by assigning random weights to components in the initial metric [14]. We present two experiments with randomly generated metrics called *RND1* and *RND2*. See Table 1.

Table 1: Initial Metrics in LearnMet Experiments



EXPT	INITIAL METRIC
DE1	$5D_{Euclidean} + 2D_{Mean} + 5D_{Max} + 1D_{LF} + 3D_{BP}$
DE2	$5D_{Euclidean} + 1.5D_{Mean} + 4.5D_{Max} + 3D_{LF} + 3D_{BP}$
EQU	$1D_{Euclidean} + 1D_{Mean} + 1D_{Max} + 1D_{LF} + 1D_{BP}$
RND1	$6D_{Euclidean} + 2.75D_{Mean} + 3.5D_{Max} + 3.25D_{LF} + 1.85D_{BP}$
RND2	$4D_{Euclidean} + 3.2D_{Mean} + 4.3D_{Max} + 1.86D_{LF} + 2.9D_{BP}$

Table 2: Learned Metrics and Number Epochs to Learn



EXPT	LEARNED METRIC	EPOCHS
DE1	$4.0645D_{Euclidean} + 1.7784D_{Mean} + 2.8428D_{Max} + 1.9625D_{LF} + 3.0238D_{BP}$	17
DE2	$4.2688D_{Euclidean} + 1.6114D_{Mean} + 2.9998D_{Max} + 2.2778D_{LF} + 2.9239D_{BP}$	19
EQU	$4.0578D_{Euclidean} + 1.8657D_{Mean} + 2.7749D_{Max} + 2.3257D_{LF} + 2.8481D_{BP}$	43
RND1	$4.1301D_{Euclidean} + 1.7345D_{Mean} + 2.8514D_{Max} + 2.1187D_{LF} + 3.1402D_{BP}$	57
RND2	$4.2204D_{Euclidean} + 1.8175D_{Mean} + 2.7968D_{Max} + 2.0581D_{LF} + 3.0864D_{BP}$	39

Observations during Training: Table 2 shows the metric learned in each experiment with number of epochs. Figures 3 to 7 depict the behavior of LearnMet during training. Experiments *EQU, RND1* and *RND2* take longer to converge than *DE1* and *DE2*. However they all converge to approximately the same *D*.

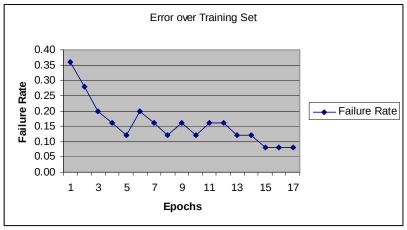


Figure 3: Experiment DE1

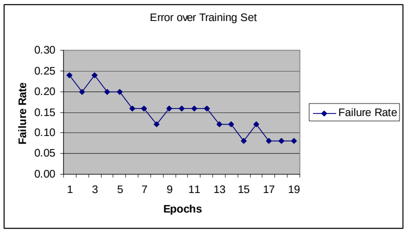


Figure 4: Experiment DE2

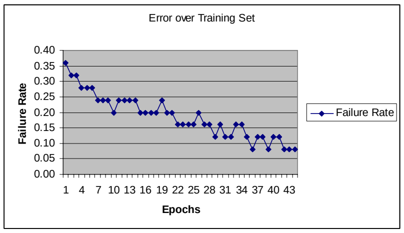


Figure 5: Experiment EQU

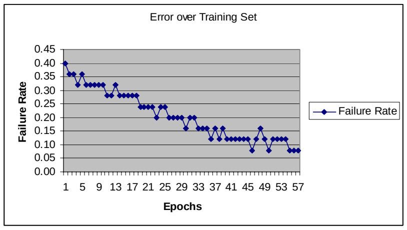


Figure 6: Experiment RND1

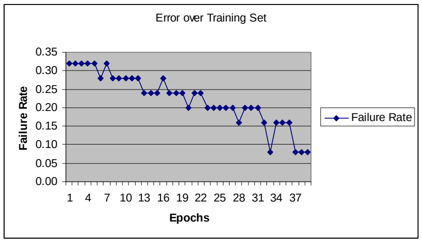


Figure 7: Experiment RND2

Observations during Testing: The learned metric in each experiment is used to cluster graphs in the test set. These clusters are compared with correct clusters over the test set. Euclidean distance (ED) is also used to cluster the graphs and the clusters are compared with the correct clusters. The observations for all the experiments are shown in Figure 8. This figure depicts the accuracy (success rate) for each metric over the test set. Clustering accuracies of the learned metrics are higher.

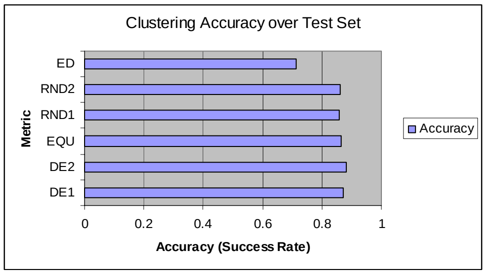


Figure 8: Test Set Observations

4.2 Evaluation with System Integration

*Purpose of Integration***:** LearnMet has been developed mainly for the AutoDomainMine system, which performs computational estimation of process parameters [13]. AutoDomainMine estimates the graphical result of a scientific experiment given its input conditions, using existing data to predict future trends. This technique clusters graphs from existing experiments and sends clustering output to a decision tree classifier e.g., ID3/J4.8 [12,15] to learn the clustering criteria. For each graph, the input conditions of its corresponding experiment and the cluster in which it was placed are used to construct the decision tree. The tree identifies the combination of input conditions that characterize each cluster. A representative graph is also selected per cluster. When input conditions of a new experiment are submitted, the tree is traversed to predict the cluster. The representative graph of that cluster is the estimated graph for that experiment.

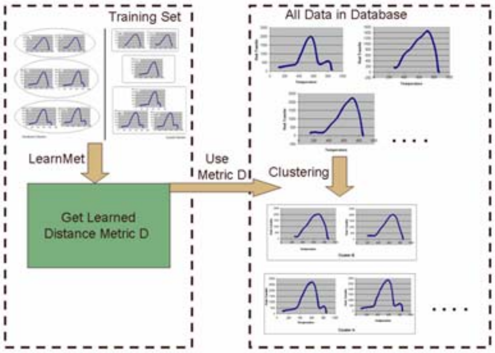
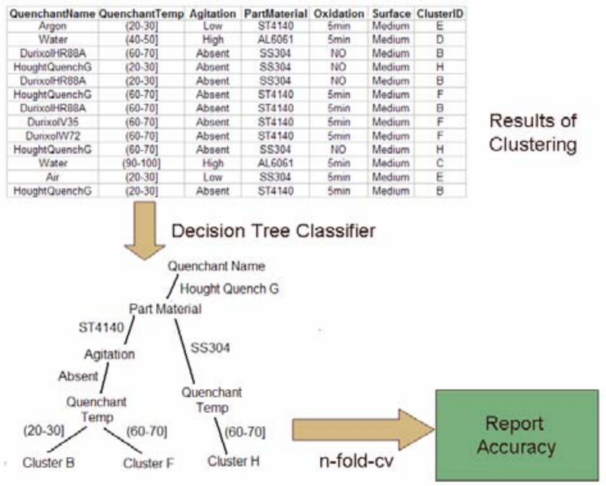


Figure 9a: Evaluation with AutoDomainMine Step 1



QuenchantName	QuenchantTemp	Agitation	PartMaterial	Oxidation	Surface	ClusterID
Argon	(20-30]	Low	ST4140	5min	Medium	E
Water	(40-50]	High	AL6061	5min	Medium	D
DurixolHR88A	(60-70]	Absent	SS304	NO	Medium	B
HoughtQuenchG	(20-30]	Absent	SS304	NO	Medium	H
DurixolHR88A	(20-30]	Absent	SS304	NO	Medium	B
HoughtQuenchG	(60-70]	Absent	ST4140	5min	Medium	F
DurixolHR88A	(60-70]	Absent	ST4140	5min	Medium	B
DurixolV35	(60-70]	Absent	ST4140	5min	Medium	F
DurixolW72	(60-70]	Absent	ST4140	5min	Medium	F
HoughtQuenchG	(60-70]	Absent	SS304	NO	Medium	H
Water	(90-100]	High	AL6061	5min	Medium	C
Air	(20-30]	Low	SS304	5min	Medium	E
HoughtQuenchG	(20-30]	Absent	ST4140	5min	Medium	B

Figure 9b: Evaluation with AutoDomainMine Step 2

Evaluating LearnMet with AutoDomainMine: LearnMet is evaluated by measuring the accuracy of the AutoDomainMine estimation with and without the learned metrics. This process is illustrated in Figures 9a and 9b. The estimation obtained from clustering using the learned metrics is compared with that from clustering using Euclidean distance. Another criterion for comparison is $D = D_{Euclidean} + D_{LF} + D_{BP}$ which is called the AutoDomainMine metric denoted as ADM [13,14].

Observations with AutoDomainMine: The average estimation accuracy over 10 experiments, each using 10-fold cross validation, is shown in Figure 10. Accuracy with each metric output from LearnMet is higher than that with Euclidean distance. Accuracies with the learned metrics are also higher than the accuracy with the AutoDomainMine metric.

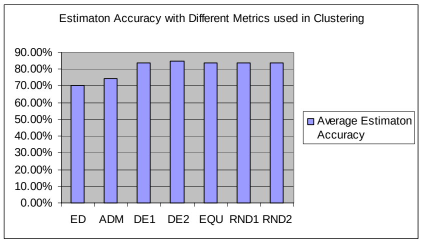


Figure 10: Evaluation Results with AutoDomainMine

5. CONCLUSIONS

A technique called LearnMet is proposed to learn a domain-specific distance metric for mining graphs. LearnMet compares clusters of graphs obtained from a state-of-the-art clustering algorithm with correct clusters given by experts. An initial metric is guessed and refined with every round of clustering to give a final metric with error below a threshold. LearnMet is evaluated rigorously in the Heat Treating domain that inspired its development. Additional evaluations will be conducted in related domains. Ongoing research includes determining a good number of pairs per epoch; using normalized weights; considering scaling factors in weight adjustments; refining thresholds; and assigning suitable components to the initial metric without expert input.

ACKNOWLEDGMENTS

This work is supported by the Center for Heat Treating Excellence and by DOE – ITP Award DE-FC-07-01ID14197.

REFERENCES

[1] Anscombe, F.J. Graphs in Statistical Analysis, in American Statistician, 27, 17-21, 1973.

[2] Bishop, C.M. Neural Networks for Pattern Recognition, Oxford University Press, England, 1996.

[3] Boyer, H., and Cary, P. Quenching and Control of Distortion, ASM International, OH, USA, 1989.

[4] Fourier, J. The Analytical Theory of Heat (translated by A. Freeman) Dover Publications Inc., NY, USA, 1955.

[5] Friedberg, R.M. A Learning Machine: Part I, in IBM Journal, 2, 2-13, 1958.

[6] Hinneburg, A., Aggarwal, C., and Keim, D. What is the Nearest Neighbor in High Dimensional Spaces, in Proceedings of VLDB-00. 506 – 515. Cairo, Egypt. August 2000.

[7] Han, J., and Kamber, M. Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, CA, USA. 2001.

[8] Keim, D., and Bustos, B. Similarity Search in Multimedia Databases, in Proceedings of ICDE-04. 873 – 874. Boston, MA, USA, March 2004.

[9] Kaufman, L., and Rousseeuw, P. Finding Groups in Data: An Introduction to Cluster Analysis, John Wiley, NY, USA. 1988.

[10] MacQueen, J. B. Some Methods for Classification and Analysis of Multivariate Observations, in Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability. 1:281-297. Berkeley, CA, USA, 1967.

[11] Petrucelli, J., Nandram, B., and Chen, M., Applied Statistics for Engineers and Scientists, Prentice Hall, NJ, USA, 1999.

[12] Quinlan, J. R. Induction of Decision Trees, in Machine Learning, 1:81-106. 1986.

[13] Varde, A., Rundensteiner, E., Ruiz, C., Maniruzzaman, M., and Sisson Jr., R. Data Mining over Graphical Results of Experiments with Domain Semantics, in Proceedings of ACM SIGART's 2nd International Conference on Intelligent Computing and Information Systems ICICIS-05. 603 – 611. Cairo, Egypt, March 2005.

[14] Varde, A., Rundensteiner, E., Ruiz, C., Maniruzzaman, M., and Sisson, Jr., R. The LearnMet Approach for Learning a Domain-Specific Distance Metric to preserve the Semantic Content in Mining Graphical Data, TR#: CS-CHTE-06-2005, WPI, Worcester, MA, USA, June 2005.

[15] Witten, I., and Frank, E., Data Mining: Practical Machine Learning Algorithms with Java Implementations, Morgan Kaufmann Publishers, CA, USA, 2000.

[16] Xing, E., Ng., A., Jordan, M., and Russell, S. Distance Metric Learning with Application to Clustering with Side Information, in Proceedings of NIPS-03, 503-512. Vancouver, Canada, December 2003.