Large-scale Classification of Fine-Art Paintings: Learning The Right Metric on The Right Feature

Babak Saleh, Ahmed Elgammal

Department of Computer Science
Rutgers University, NJ, USA
{babaks,elgammal}@cs.rutgers.edu

Abstract. In the past few years, the number of fine-art collections that are digitized and publicly available has been growing rapidly. With the availability of such large collections of digitized artworks comes the need to develop multimedia systems to archive and retrieve this pool of data. Measuring the visual similarity between artistic items is an essential step for such multimedia systems, which can benefit more high-level multimedia tasks. In order to model this similarity between paintings, we should extract the appropriate visual features for paintings and find out the best approach to learn the similarity metric based on these features. We investigate a comprehensive list of visual features and metric learning approaches to learn an optimized similarity measure between paintings. We develop a machine that is able to make aesthetic-related semantic-level judgments, such as predicting a painting's style, genre, and artist, as well as providing similarity measures optimized based on the knowledge available in the domain of art historical interpretation. Our experiments show the value of using this similarity measure for the aforementioned prediction tasks.

1 Introduction

In the past few years, the number of fine-art collections that are digitized and publicly available has been growing rapidly. Such collections span classical [1] and modern and contemporary artworks [2]. With the availability of such large collections of digitized artworks comes the need to develop multimedia systems to archive and retrieve this pool of data. Typically these collections, in particular early modern ones, come with metadata in the form of annotations by art historians and curators, including information about each painting's artist, style, date, genre, etc. For online galleries displaying contemporary artwork, there is a need to develop automated recommendation systems that can retrieve "similar" paintings that the user might like to buy. This highlights the need to investigate metrics of visual similarity among digitized paintings that are optimized for the domain of painting.

The field of computer vision has made significant leaps in getting digital systems to recognize and categorize objects and scenes in images and videos. These advances have been driven by a wide spread need for the technology, since cameras are everywhere now. However a person looking at a painting can make sophisticated inferences

[1] Examples: Wikiart; Arkyves; BBC Yourpainting

[2] Examples: Artsy; Behance; Artnet

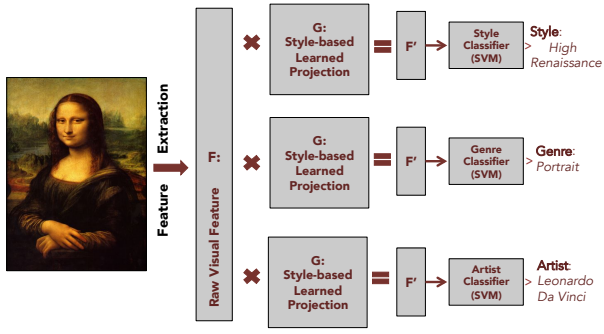

Fig. 1: Illustration of our system for classification of fine-art paintings. We investigated variety of visual features and metric learning approaches to recognize *Style, Genre* and *Artist* of a painting.

beyond just recognizing a tree, a chair, or the figure of Christ. Even individuals without specific art historical training can make assumptions about a painting's genre (portrait or landscape), its style (impressionist or abstract), what century it was created, the artists who likely created the work and so on. Obviously, the accuracy of such assumptions depends on the viewer's level of knowledge and exposure to art history. Learning and judging such complex visual concepts is an impressive ability of human perception [2].

The ultimate goal of our research is to develop a machine that is able to make aesthetic-related semantic-level judgments, such as predicting a painting's style, genre, and artist, as well as providing similarity measures optimized based on the knowledge available in the domain of art historical interpretation. Immediate questions that arise include, but are not limited to: What visual features should be used to encode information in images of paintings? How does one weigh different visual features to achieve a useful similarity measure? What type of art historical knowledge should be used to optimize such similarity measures? In this paper we address these questions and aim to provide answers that can benefit researchers in the area of computer-based analysis of art. Our work is based on a systematic methodology and a comprehensive evaluation on one of the largest available digitized art datasets.

Artists use different concepts to describe paintings. In particular, stylistic elements, such as space, texture, form, shape, color, tone and line are used. Other principles include movement, unity, harmony, variety, balance, contrast, proportion, and pattern. To this might be added physical attributes, like brush strokes as well as subject matter and other descriptive concepts [13].

For the task of computer analyses of art, researchers have engineered and investigated various visual features[3] that encode some of these artistic concepts, in particular brush strokes and color, which are encoded as low-level features such as texture statistics and color histograms (e.g. [19, 20]). Color and texture are highly prone to variations

[3] In contrast to art disciplines, in the fields of computer vision and machine learning, researchers use the term"visual features" to denote statistical measurements that are extracted from images for the task of classification. In this paper we stick to this typical terminology.

Task Name	List of Members
Style	Abstract Expressionism(1); Action Painting(2); Analytical Cubism(3); Art Nouveau-Modern Art(4); Baroque(5); Color Field Painting(6); Contemporary Realism(7); Cubism(8); Early Renaissance(9); Expressionism(10); Fauvism(11); High Renaissance(12); Impressionism(13); Mannerism-Late-Renaissance(14); Minimalism(15); Primitivism-Naive Art(16); New Realism(17); Northern Renaissance(18); Pointillism(19); Pop Art(20); Post Impressionism(21); Realism(22); Rococo(23); Romanticism(24); Symbolism(25); Synthetic Cubism(26); Ukiyo-e(27)
Genre	Abstract painting(1); Cityscape(2); Genre painting(3); Illustration(4); Landscape(5); Nude painting(6); Portrait(7); Religious painting(8); Sketch and Study(9); Still Life(10)
Artist	Albrecht Durer(1); Boris Kustodiev(2); Camille Pissarro(3); Childe Hassam(4); Claude Monet(5); Edgar Degas(6); Eugene Boudin(7); Gustave Dore(8); Ilya Repin(9); Ivan Aivazovsky(10); Ivan Shishkin(11); John Singer Sargent(12); Marc Chagall(13); Martiros Saryan(14); Nicholas Roerich(15); Pablo Picasso(16); Paul Cezanne(17); Pierre-Auguste Renoir(18); Pyotr Konchalovsky(19); Raphael Kirchner(20); Rembrandt(21); Salvador Dali(22); Vincent van Gogh(23)

Table 1: List of Styles, Genres and Artists in our collection of fine-art paintings. Numbers in the parenthesis are index of the row/column in confusion matrices 5, 6& 7 accordingly.

during the digitization of paintings; color is also affected by a painting's age. The effect of digitization on the computational analysis of paintings is investigated in great depth by Polatkan et al. [24]. This highlights the need to carefully design visual features that are suitable for the analysis of paintings.

Clearly, it would be a cumbersome process to engineer visual features that encode all the aforementioned artistic concepts. Recent advances in computer vision, using deep neural networks, showed the advantage of "learning" the features from data instead of engineering such features. However, It would also be impractical to learn visual features that encode such artistic concepts, since that would require extensive annotation of these concepts in each image within a large training and testing dataset. Obtaining such annotations require expertise in the field of art history that can not be achieved with typical crowed-sourcing annotators.

Given the aforementioned challenges to engineering or learning suitable visual features for painting, in this paper we follow an alternative strategy. We mainly investigate different state-of-the-art visual elements, ranging from low-level elements to semantic-level elements. We then use metric learning to achieve optimal similarity metrics between paintings that are optimized for specific prediction tasks, namely style, genre, and artist classification. We chose these tasks to optimize and evaluate the metrics since, ultimately, the goal of any art recommendation system would be to retrieve artworks that are similar along the directions of these high-level semantic concepts. Moreover, annotations for these tasks are widely available and more often agreed-upon by art historians and critics, which facilitates training and testing the metrics.

In this paper we investigate a large space of visual features and learning methodologies for the aforementioned prediction tasks. We propose and compare three learning methodologies to optimize such tasks. We present results of a comprehensive comparative study that spans four state-of-the-art visual features, five metric learning approaches and the proposed three learning methodologies, evaluated on the aforementioned three artistic prediction tasks.

2 Related Work

On the subject of painting, computers have been used for a diverse set of tasks. Traditionally, image processing techniques have been used to provide art historians with quantification tools, such as pigmentation analysis, statistical quantification of brush strokes, etc. We refer the reader to [28, 5] for comprehensive surveys on this subject.

Several studies have addressed the question of which features should be used to encode information in paintings. Most of the research concerning the classification of paintings utilizes low-level features encoding color, shadow, texture, and edges. For example Lombardi [20] has presented a study of the performance of these types of features for the task of artist classification among a small set of artists using several supervised and unsupervised learning methodologies. In that paper the style of the painting was identified as a result of recognizing the artist.

Since brushstrokes provide a signature that can help identify the artist, designing visual features that encode brushstrokes has been widely adapted.(e.g. [25, 18, 22, 15, 6, 19]). Typically, texture statistics are used for that purpose. However, as mentioned earlier, texture features are highly affected by the digitization resolution. Researchers also investigated the use of features based on local edge orientation histograms, such as SIFT [21] and HOG [10]. For example, [12] used SIFT features within a Bag-of-words pipeline to discriminate among a set of eight artists.

Arora et al. [3] presented a comparative study for the task of style classification, which evaluated low-level features, such as SIFT and Color SIFT [1], versus semantic-level features, namely Classemes [29], which encodes object presence in the image. It was found that semantic-level features significantly outperform low-level features for this task. However the evaluation was conducted on a small dataset of 7 styles, with 70 paintings in each style. Carneiro et al [9] also concluded that low-level texture and color features are not effective because of inconsistent color and texture patterns that describe the visual classes in paintings.

More recently, Saleh et al [26] used metric learning approaches for finding influence paths between painters based on their paintings. They evaluated three metric learning approaches to optimize a metric over low-level HOG features. In contrast to that work, the evaluation presented in this paper is much wider in scope since we address three tasks (style, genre and artist prediction), we cover features spanning from low-level to semantic-level and we evaluate five metric learning approaches. Moreover, The dataset of [26] has only 1710 images from 66 artists, while we conducted our experiments on 81,449 images painted by 1119 artists. Bar et al [4] proposed an approach for style classification based on features obtained from a convolution neural network pre-trained on an image categorization task. In contrast we show that we can achieve better results with much lower dimensional features that are directly optimized for style and genre classification. Lower dimensionality of the features is preferred for indexing large image collections.

3 Methodology

In this section we explain the methodology that we follow to find the most appropriate combination of visual features and metrics that produce accurate similarity measure-

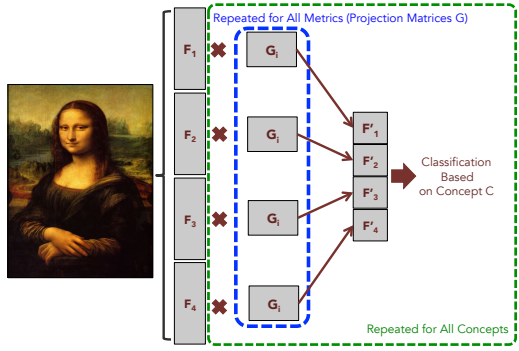

Fig. 2: Illustration of our second methodology - Feature Fusion.

ments. We acquire these measurements to mimic the art historian's ability to categorize paintings based on their style, genre and the artist who made it. In the first step, we extract visual features from the image. These visual features range from low-level (e.g. edges) to high-level (e.g. objects in the painting). More importantly, in the next step we learn how to adjust these features for different classification tasks by learning the appropriate metrics. Given the learned metric we are able to project paintings from a high dimensional space of raw visual information to a meaningful space with much lower dimensionality. Additionally, learning a classifier in this low-dimensional space can be easily scaled up for large collections.

In the rest of this section: First, we introduce our collection of fine-art paintings and explain what are the tasks that we target in this work. Later, we explore methodologies that we consider in this work to find the most accurate system for aforementioned tasks. Finally, we explain different types of visual features that we use to represent images of paintings and discuss metric learning approaches that we applied to find the proper notion of similarity between paintings.

3.1 Dataset and Proposed Tasks

In order to gather our collection of fine-art paintings, we used the publicly available dataset of *"Wikiart paintings"*[4]; which, to the best of our knowledge, is the largest online public collection of digitized artworks. This collection has images of 81,449 fine-art paintings from 1,119 artists ranging from fifteen centuries to contemporary artists. These paintings are from 27 different styles (Abstract, Byzantine, Baroque, etc.) and 45 different genres (Interior, Landscape, etc.) Previous work [26, 9] used different resources and made smaller collections with limited variability in terms of style, genre and artists. The work of [4] is the closest to our work in terms of data collection procedure, but the number of images in their collection is half of ours.

We target automatic classification of paintings based on their style, genre and artist using visual features that are automatically extracted using computer vision algorithms. Each of these tasks has its own challenges and limitations. For example, there are large

[4] http://www.wikiart.org/

variations in terms of visual appearances in paintings from one specific style. However, this variation is much more limited for paintings by one artist. These larger intra-class variations suggests that style classification based on visual features is more challenging than artist classification. For each of the tasks we selected a subset of the data that ensure enough samples for training and testing. In particular for style classification we use a subset of the date with 27 styles where each style has at least 1500 paintings with no restriction on genre or artists, with a total of 78,449 images. For genre classification we use a subset with 10 genre classes, where each genre has at least 1500 paintings with no restriction of style or genre, with a total of 63,691 images. Similarly for artist classification we use a subset of 23 artists, where each of them has at least 500 paintings, with a total of 18,599 images. Table 1 lists the set of style, genre, and artist labels.

3.2 Classification Methodology

In order to classify paintings based on their style, genre or artist we followed three methodologies.

Metric Learning: First, as depicted in figure 1, we extract visual features from images of paintings. For each of these prediction tasks, we learn a similarity metric optimized for it, i.e. style-optimized metric, genre-optimized metric and artist-optimized metric. Each metric induces a projector to a corresponding feature space optimized for the corresponding task. Having the metric learned, we project the raw visual features into the new optimized feature space and learn classifiers for the corresponding prediction task. For that purpose we learn a set of one-vs-all SVM classifiers for each of the labels in table 1 for each of the tasks.

While our first strategy focuses on classification based on combinations of a metric and a visual feature, the next two methodologies that we followed fuse different features or different metrics.

Feature fusion: The second methodology that we used for classification is depicted in figure 2. In this case, we extract different types of visual features (four types of features as will explained next). Based on the prediction task (e.g. style) we learn the metric for each type of feature as before. After projecting these features separately, we concatenate them to make the final feature vector. The classification will be based on training classifiers using these final features. This feature fusion is important as we want to capture different types of visual information by using different types of features. Also concatenating all features together and learn a metric on top of this huge feature vector is computationally intractable. Because of this issue, we learn metrics on feature separately and after projecting features by these metrics, we can concatenate them for classification purposes.

Metric-fusion: The third methodology (figure 3) projects each visual features using multiple metrics (in our experiment we used five metrics as will be explained next) and then fuses the resulting optimized feature spaces to obtain a final feature vector for classification. This is an important strategy, because each one of the metric learning approaches use a different criteria to learn the similarity measurement. By learning all metrics individually (on the same type of feature), we make sure that we took into account all criteria (e.g. information theory along with neighbor hood analysis).

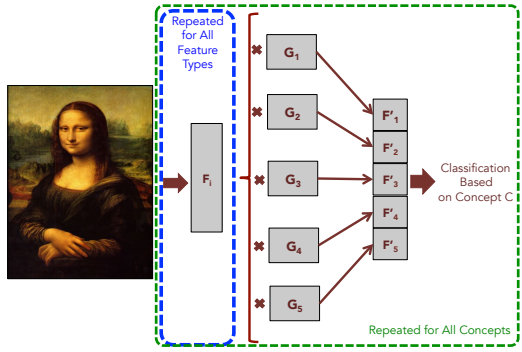

Fig. 3: Illustration of our third methodology– Metric Fusion.

3.3 Visual Features

Visual features in computer vision literature are either engineered and extracted in an unsupervised way (e.g. HOG, GIST) or learned based on optimizing a specific task, typically categorization of objects or scenes (e.g. CNN-based features). This results in high-dimensional feature vectors that might not necessary correspond to nameable (semantic-level) characteristics of an image. Based on the ability to find a meaning, visual features can be categorized into low-level and high-level. Low-level features are visual descriptors that there is no explicit meaning for each dimension of them, while high-level visual features are designed to capture some notions (usually objects). For this work, we investigated some state-of-the-art representatives of these two categories:

Low-level Features: On one hand, in order to capture low-level visual information we extracted GIST features [23], which are holistic features that are designed for scene categorization. GIST features provide a 512 real-valued representation that implicitly captures the dominant spatial structure of the image.

Learned Semantic-level Features: On the other hand, for the purpose of semantic representation of the images, we extracted three object-based representation of the images: Classeme [29], Picodes [8], and CNN-based features [16]. In all these three features, each element of the feature vector represents the confidence of the presence of an object-category in the image, therefore they provide a semantic encoding of the images. However, for learning these features, the object-categories are generic and are not art-specific. First two features are designed to capture the presence of a set of basic-level object categories as following: a list of entry-level categories (e.g. horse and cross) is used for downloading a large collection of images from the web. For each image a comprehensive set of low-level visual features are extracted and one classifier is learned for each category. For a given test image, these classifiers are applied on the image and the responses (confidences) make the final feature vector. We followed the implementation of [7] and for each image extracted a 2659 dimensional real-valued Classeme feature vector and a 2048 dimensional binary-value Picodes feature.

Convolutional Neural Networks(CNN) [17] showed a remarkable performance for the task of large-scale image categorization [16]. CNNs have four convolutional layers followed by three fully connected layers. Bar et al [4] showed that a combination of the

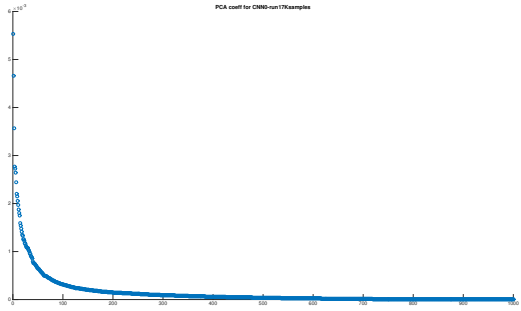

Fig. 4: PCA coefficients for CNN features

output of these fully connected layers achieve a superior performance for the task of style classification of paintings. Following this observation we used the last layer of a pre-trained CNN [16] (1000 dimensional real-valued vectors) as another feature vector.

3.4 Metric Learning

The purpose of Metric Learning is to find some pair-wise real-valued function $d_M(x, x')$ which is non-negative, symmetric, obeys the triangle inequality and returns zero if and only if x and x' are the same point. Training such a function in a general form can be seen as the following optimization problem:

$$\min_M l(M, D) + \lambda R(M) \tag{1}$$

This optimization has two sides, first it tries to minimize the amount of loss $l(M, D)$ by using metric M over data samples D while trying to adjust the model by the regularization term $R(M)$. The first term shows the accuracy of the trained metric and second one estimates its capability over new data and avoids overfitting. Based on the enforced constraints, the resulted metric can be linear or non-linear and depending on the amount of labels used for training, it can be supervised or unsupervised.

For consistency over the metric learning algorithms, we need to fix the notation first. We learn the matrix M that will be used in Generalized Mahalanobis Distance: $d_M(x, x') = \sqrt{(x - x')' M (x - x')}$, where M by definition is a positive semi-definite matrix and can be decomposed as $M = G^T G$. We use this matrix G to project raw visual features. Measuring similarity in this projection space is simply computing the euclidean distance between two item.

It is interesting that we can reduce the dimension of features during learning the metric when M is a low rank matrix. More importantly, there are significantly important information in the ground truth annotation associated with paintings that we use to learn a more reliable metric in a supervised fashion for both the linear and non-linear cases. We consider following approaches that differ based on the form of M or the amount of regularization.

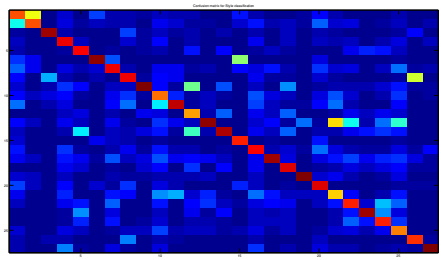

Fig. 5: Confusion matrix for Style classification. Confusions are meaningful only when seen in color.

Neighborhood Component Analysis (NCA) The objective function of NCA [14] is related to analyzing the nearest neighbors. The idea starts with projecting the data by matrix M and training a leave-one-out classifier. Then the probability of correctly classifying x_i is $P_i = \sum_{j:y_i=y_j} P_{ij}$, where P_{ij} is the mean expected loss of classifying x_i as a member of class j.

Then this metric is learned by optimizing the following term: $\max_M \sum_i P_i$. We can decompose M as $L' * L$ and choosing a rectangular L will result in a low-rank matrix M. Although this method is easy to understand and implement, it is subject to local minimums. This happens due to the non-convexity of the proposed optimization problem. The next approach has the advantage of solving a convex optimization.

Large Margin Nearest Neighbors (LMNN) LMNN [32] is an approach for learning a Mahalanobis distance, which is widely used because of its global optimum solution and superior performance in practice. The learning of this metric involves a set of constrains, all of which are defined locally. This means that LMNN enforces the k nearest neighbor of any training instance belonging to the same class (these instances are called "target neighbors"). This should be done while all the instances of other classes, referred as "impostors", should be far from this point. For finding the target neighbors, Euclidean distance has been applied to each pair of samples, resulting in the following formulation:

$$\min_M (1-\mu) \sum_{(x_i,x_j)\in T} d^2_M(x_i,x_j) + \mu \sum_{i,j,k} \eta_{i,j,k}$$
$$s.t. : d^2_M(x_i,x_k) - d^2_M(x_i,x_j) \geq 1 - \eta_{i,j,k} \forall (x_i,x_j,x_k) \in I.$$

Where T stands for the set of *Target* neighbors and I represents *Impostors*. Since these constrains are locally defined, this optimization leads to a convex formulation and a global solution. This metric learning approach is related to Support Vector Machines (SVM) in principle, which theoretically engages its usage along with SVM for the task of classification.

Due to the popularity of LMNN, different variations of it have been introduced, including a non-linear version called gb-LMNN [32] which we used in our experiments

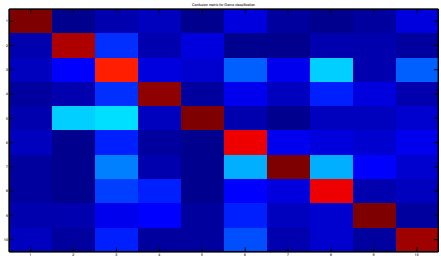

Fig. 6: Confusion matrix for Genre classification. Confusions are meaningful only when seen in color.

as well. However its performance for classification tasks was worse that linear LMNN. We assume this poor performance is rooted in the nature of visual features that we extract for paintings.

Boost Metric This approach is based on the fact that a positive semi-definite matrix can be decomposed into a linear combination of trace-one rank-one matrices. Shen et al [27] use this fact and instead of learning M, finds a set of weaker metrics that can be combined and give the final metric. They treat each of these matrices as a *Weak Learner*, which is used in the literature of Boosting methods. The resulting algorithm applies the idea of AdaBoost to Mahalanobis distance, which has been shown to be quiet efficient in practice.

This method is particularly of our interest, because we can learn an individual metric for each style of paintings and finally merge these metrics to get a unique final metric. Theoretically the final metric can perform well to find similarities inside each style/genre of paintings as well.

Information Theory Metric Learning (ITML) This metric learning algorithm is based on Information theory rather than Mahalanobis distances. In other words the optimization problem of learning a metric involves an information measure.

Davis et al [11] introduce the measure of *LogDet divergence regularization* between two matrices M, M'(can be interpreted as metrics). By using this measure, learning the metric can be represented by:

$$\min_{M' \in PSD} D_{ld}(M, M') + \gamma \sum_{i,j} \epsilon_{i,j}$$
$$s.t. : d^2_{M'}(x_i, x_j) \leq u + \epsilon_{i,j} \forall (x_i, x_j) \in S.$$
$$d^2_{M'}(x_i, x_j) \geq v - \epsilon_{i,j} \forall (x_i, x_j) \in D.$$

Learning ITML via this formulation aims to satisfy a set of *Similarity(S)* and *Dissimilarity(D)* constrains while keeping the new metric M' close to the initial metric

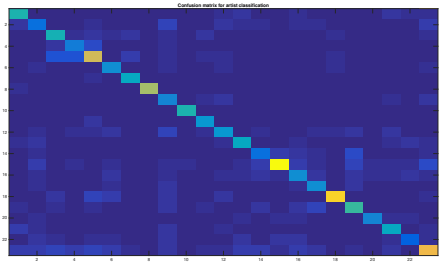

Fig. 7: Confusion matrix for Artist classification. Confusions are meaningful only when seen in color.

M. There are two key features of the *LogDet divergence*: 1) It is finite if and only if matrices are positive semi-definite(PSD), 2) This function is rank-preserving.

These properties indicate that if we start learning the metric M' by setting the initial matrix M as identity matrix(I), ITML returns a metric that is from the same rank and is very similar to the Euclidean distance.

Although this iterative process converges to a global minimum which performs well in practice, it is very sensitive to the choice of initialization of metric(M).

Metric Learning for Kernel Regression (MLKR) Similar to NCA objective function, which minimizes the classification error; Weinberger and Tesauro [31] learn a metric by optimizing the leave-one-out error for the task of kernel regression. In kernel regression, there is an essential need for proper distances between points that will be used for weighting sample data. MLKR learn this distance by minimizing the leave-one-out error for regression on training data. Although this metric learning method is designed for kernel regression, the resulted distance function can be used in variety of tasks.

4 Experiments

4.1 Experimental Setting

Visual Features As we explained in section 3, we extract GIST features as low-level visual features and Classeme, Picodes and CNN-based features as the high-level semantic features. We followed the original implementation of Oliva and Torralba [23] to get a 512 dimensional feature vector. For Classeme and Picodes we used the implementation of Bergamo et al [29], resulting in 2659 dimensional Classeme features and 2048 dimensional Picodes features. We used the implementation of Vedaldi and Lenc [30] to extract 1000 dimensional feature vectors of the last layer of CNN.

Object-based representations of the images produce feature vectors that are much higher in dimensionality than GIST descriptors. In the sake of a fair comparison of all types of features for the task of metric learning, we transformed all feature vectors to have the same size as GIST (512 dimensional). We did this by applying Principle Component Analysis (PCA) for each type and projecting the original features onto the first

Metric / Features	GIST	Classemes	Picodes	CNN	Dim.
Baseline	10.83	22.62	20.76	12.32	512
Boost	**16.07**	**31.77**	**28.58**	15.18	512
ITML	13.02	30.67	28.42	15.34	512
LMNN	12.54	27	24.14	**16.83**	100
MLKR	12.65	24.12	14.86	12.63	512
NCA	13.29	28.19	24.84	16.37	27

Table 2: Accuracy for the task of style classification

512 eigenvectors (with biggest eigenvalues). In order to verify the quality of projection, we looked at the corresponding coefficients of eigenvalues for PCA projections. Independent of feature type, the value of these coefficients drops significantly after the first 500 eigenvectors. For example, figure 4 plots these coefficients of PCA projection for CNN features. Summation of the first 500 coefficients is 95.88% of the total summation. This shows that our projections (with 512 eigenvectors) captures the true underlying space of the original features. Using these reduced features speeds up the metric learning process as well.

Metric Learning We used implementation of [32] to learn LMNN metric(both version of linear and non-linear) and MLKR [5]. For the BoostMetric we slightly adjusted the implementation of [27]. For NCA we adopted its implementation by Fowlkes[6] to work on large scale feature vectors smoothly. For the case of ITML metric learning, we followed the original implementation of authors with the default setting. For the rest of methods, parameters are chosen through a grid search that finds the minimum nearest neighbor classification. Regarding the training time, learning the ITML metric was the fastest and learning NCA and LMNN were the slowest ones. Due to computational constrains we set the parameters of LMNN metric to reduce the size of features to 100. NCA metric reduces the dimension of features to the number of categories for each tasks: 27 for style classification, 23 for artist classification and 10 for genre classification. We randomly picked 3000 samples, which we used for metric learning. These samples follow the same distribution as original data and are not used for classification experiments.

4.2 Classification Experiments

For the purpose of metric learning, we conducted experiments with labels for three different tasks of style, genre and artist prediction. In following sections we investigate the performance of these metrics on different features for classification of aforementioned concepts.

We learned all the metrics in section3 for all 27 styles of paintings in our dataset (e.g. Expressionism, Realism, etc.). However, we did not use all the genres for learning metrics. In fact in our dataset we have 45 genres, some of which have less than 20 images. This makes the metric learning impractical and highly biased toward genres

[5] http://www.cse.wustl.edu/ kilian/index.html

[6] http://www.ics.uci.edu/ fowlkes/

Metric / Features	GIST	Classemes	Picodes	CNN	Dim.
Baseline	28.10	49.98	49.63	35.14	512
Boost	31.01	**57.87**	**57.35**	46.14	512
ITML	33.10	57.86	57.28	46.80	512
LMNN	**39.06**	54.96	54.42	**49.98**	100
MLKR	32.81	54.29	42.79	45.02	512
NCA	30.39	51.38	52.74	49.26	10

Table 3: Accuracy for the task of genre classification

with larger number of paintings. Because of this issue, we focus on 10 genres with more than 1500 paintings. These genres are listed in table 1. In all experiments we conducted 3 fold cross validation and reported the average accuracy over all partitions. We found the best value for penalty term in SVM (which is equal to 10) by three fold cross validation. In the next three sections, we explain settings and findings for each task independently.

Style Classification Table 2 contains the result (accuracy percentage) of style classification (SVM) after applying different metrics on a set of features. Columns correspond to different features and rows are different metrics that are used for projecting features before learning style classifiers. In order to quantify the improvement by learning similarity metrics, we conducted a baseline experiment (first row in the table) as the following: For each type of features, we learn a set of one-vs-all classifiers on raw feature vectors. Generally Boost metric learning and ITML approaches give the highest in accuracy for the task of style classification over different visual features. However the greatest improvement over the baseline is gained by application of Boost metric on Classeme features. We visualized the confusion matrix for the task of style classification, when we learn Boost metric on Classeme features.

Figure 5 shows this matrix, where red represents higher values. Further analysis of some confusions that are captured in this matrix result in interesting findings. In the rest of this paragraph we explain some of these cases. First, we found that there is a big confusion between “Abstract expressionism” (first row) and “Action paintings” (second column). Art historians verify the fact that this confusion is meaningful and somehow expected. “Action painting” is a type or subgenre of “abstract expressionism” and are characterized by paintings created through a much more active process– drips, flung paint, stepping on the canvas.

Another confusion happens between “Expressionism” (column 10) and “Fauvism” (row 11), which is actually expected based on art history literature. “Mannerism” (row 14) is a style of art during the (late)“Renaissance” (column 12), where they show unusual effect in scale and are less naturalistic than “Early Renaissance”. This similarity between “Mannerism” (row 14) and “Renaissance” (column 12) is captured by our system as well where results in confusion during style classification. “Minimalism” (column 15) and “Color field paintings”(6th row) are mostly confused with each other. We can agree on this finding as we look at members of these styles and figure out the similarity in terms of simple form and distribution of colors. Lastly some of the confusions are completely acceptable based on the origins of these styles (art movements) that are

Metric / Features	GIST	Classemes	Picodes	CNN	Dim.
Baseline	17.58	45.29	45.82	20.38	512
Boost	**25.65**	**57.76**	**55.50**	29.65	512
ITML	19.95	51.79	53.93	**31.04**	512
LMNN	20.41	53.99	53.92	30.92	100
MLKR	21.22	49.61	19.54	21.77	512
NCA	18.80	53.70	53.81	22.26	23

Table 4: Accuracy for the task of artist classification

noted in art history literature. For example, “Renaissance”(column 18) and “Early Renaissance”(row 9); “Post Impressionism” (column 21) and “Impressionism”(row 13); “Cubism” (8th row) and “Synthetic Cubism” (column 26). Synthetic cubism is the later act of cubism with more color continued usage of collage and pasted papers, but less linear perspective than cubism.

Genre Classification We narrowed down the list of all genres in our dataset (45 in total) to get a reasonable number of samples for each genre (10 selected genres are listed in table 1). We trained ten one-vs-all SVM classifiers and compare their performance in Table 3. In this table columns represent different features and rows are different metric that we used to compute the distance. As table 3 shows we achieved the best performance for genre classification by learning Boost metric on top of Classeme features. Generally the performance of these classifiers are better than classifiers that we trained for style classification. This is expected as the number of genres is less than the number of styles in our dataset.

Figure 6 shows the confusion matrix for classification of genre by learning Boost metric, when we used Classeme features. Investigating the confusions that we find in this matrix, reveals interesting results. For example, our system confuses “Landscape” (5th row) with “Cityspace” (2nd column) and “Genre paintings” (3rd column). However, this confusion is expected as art historians can find common elements in these genres. On one hand “Landscape” paintings usually show rivers, mountains and valleys and there is no significant figure in them; frequently very similar to “Genre paintings” as they capture daily life. The difference appears in the fact that despite the “Genre paintings”, “Landscape” paintings are idealized. On the other hand, “Landscape” and “Cityspace” paintings are very similar as both have open space and use realistic color tonalities.

Artist Classification For the task of the artist classification, we trained one-vs-all SVM classifiers for each of 23 artists. For each test image, we determine its artist by finding the classifier that produces the maximum confidence. Table 4 shows the performance of different combinations of features and metrics for this task. In general learning Boost metric improves artist classification better than all other metrics, except the case of CNN features where learning ITML metric gained the best performance. We plotted the confusion matrix of this classification task in figure 7. In this plot, some confusions between artists are clearly reasonable. We investigated two cases:

Task / Features	GIST	Classemes	Picodes	CNN
Style	20.21	**37.33**	**33.27**	21.99
Genre	35.94	**58.29**	**56.09**	47.05
Artist	30.37	**59.37**	**55.65**	33.62

Table 5: Classification performance for metric fusion methodology.

First case, "Claude Monet"(5th row) and "Camille Pissaro"(3rd column). Both of these Impressionist artists who lived in the late nineteen and early twentieth centuries. Interestingly, based on art history literature Monet and Pissaro became friends when they both attended the "Académie Suisse" in Paris. This friendship lasted for a long time and resulted in some noticeable interactions between them. Second case, paintings of "Childe Hassam"(4th row) are mostly confused with ones from "Monet"(5th column). This confusion is acceptable as Hassam is an American Impressionist, who declared himself as being influenced by French Impressionists. Hassam called himself an "Extreme Impressionist", who painted some flag-themed artworks similar to Monet.

By looking at reported performances in tables 2- 4, we conclude that, all three classification tasks can benefit from learning the appropriate metric. This means that we can improve the accuracy of baseline classification by learning metrics independent of the type of visual feature or the concept that we are classifying painting based on. Experimental results show that, independent of the task, NCA and MLKR approaches are performing worse than other metrics. Additionally, Boost metric always gives the best or the second best results for all classification tasks.

Regarding analysis of importance of features, we can verify that Classeme and Picode features are better image representations for classification purposes. Based on these classification experiments, we claim that Classemes and Picodes features perform better than CNN features. This is rooted in the fact that amount of supervision for training Classeme and Picodes is more than CNN training. Also, unlike Classeme and Picodes, CNN feature is designed to categorize the object insides a given bounding box. However, in the case of paintings we cannot assume that all the bounding boxes around the objects are given.

Integration of Features and Metrics We investigated the performance of different metric learning approaches and visual features individually. In the next step, we find out the best performance for aforementioned classification tasks by combining different visual features. Toward this goal, we followed two strategies. First, for a given metric, we project visual features by applying the metric and concatenate these projected visual features together. Second, we fixed the type of visual feature that we use and project it with the application of different metrics and concatenate these projections all together. Having this larger feature vectors (either of two strategies), we train SVM classifiers for three tasks of Style, Genre and Artist classification. Table 6 shows the results of these experiments where we followed the earlier strategy and table 5 shows the results of the later case. In general we get better results by fixing the metric and concatenating the projected feature vectors (first strategy).

The work of Bar et al [4] is the most similar to ours and we compare our final results of these experiments with their reported performance. [4] only performed the task

Concept / Metric	Boost	ITML	LMNN	MKLR	NCA
Style	41.74	**45.05**	**45.97**	38.91	40.61
Genre	**58.51**	**60.28**	58.48	55.79	54.82
Artist	**61.24**	60.46	**63.06**	53.19	55.83

Table 6: Classification results for feature fusion methodology.

of style classification on half of the images in our dataset and achieved the accuracy of 43% by using two variations of PiCoDes features and two layers of CNN. However we outperform their approach by achieving 45.97 % accuracy for the task of style classification when we used LMNN metric to project GIST, Classeme, PiCoDes and CNN features and concatenate them all together as it is reported in the third column of table 6.

Our contribution goes beyond outperforming state-of-the-art by learning a more compact feature representation. In this work, our best performance for style classification happens when we concatenate four 100-dimensional feature vectors. This results in a 400 dimensional feature vectors that we train SVM classifiers on top of them. However [4] extract a 3882 dimensional feature vector to their best reported performance. As a result we not only outperform the state-of-the-art, but presented a better image representation that reduces the amount of space by 90%. Our efficient feature vector is an extremely useful image representation that gains the best classification accuracy and we consider its application for the task of image retrieval as future work.

To qualitatively evaluate extracted visual features and learned metrics, we did a prototype image search task. As the feature fusion with application of LMNN metric gives the best performance for style classification, we used this setting as our similarity measurement model. Figure 8 shows some sample output of this image search task. For each pair, the image on the left is the query image, which we find the closest match(image on the right) to it based on LMNN and feature fusion. However we force the system to pick the closest match that does not belong to the same style as the query image. This verifies that although we learn the metric based on style labels, the learned projection can find similarity across styles.

5 Conclusion and Future Works

In this paper we investigated the applicability of metric learning approaches and performance of different visual features for learning similarity in a collection of fine-art paintings. We implemented meaningful metrics for measuring similarity between paintings. These metrics are learned in a supervised manner to put paintings from one concept close to each other and far from others. In this work we used three concepts: Style, Genre and Artist. We used these learned metrics to transform raw visual features into another space that we can significantly improve the performance for three important tasks of *Style, Genre* and *Artist classification*. We conducted our comparative experiments on the largest publicly available dataset of fine-art paintings to evaluate the performance for the aforementioned tasks.

We conclude that:

- Classeme features show the superior performance for all three tasks of Style, Genre or Artist classification. This superior performance is independent of the type of metric that has been learned.
- In the case of working on individual type of visual features, Boost metric and Information Theoretic Metric Learning(ITML) approaches improve the accuracy of classification tasks across all features.
- For the case of using different types of features all together(feature fusion), Large-Margin Nearest-Neighbor(LMNN) metric learning achieves the best performance for all classification experiments.
- By learning LMNN metric on Classeme features, we find an optimized representation that not only outperforms state-of-the art for the task of style classification, but reduce the size of feature vector by 90%.

We consider verification of applicability of this representation for the task of image retrieval and recommendation systems as future work. As other future works we would like to learn metrics based on other annotation(e.g. time period).

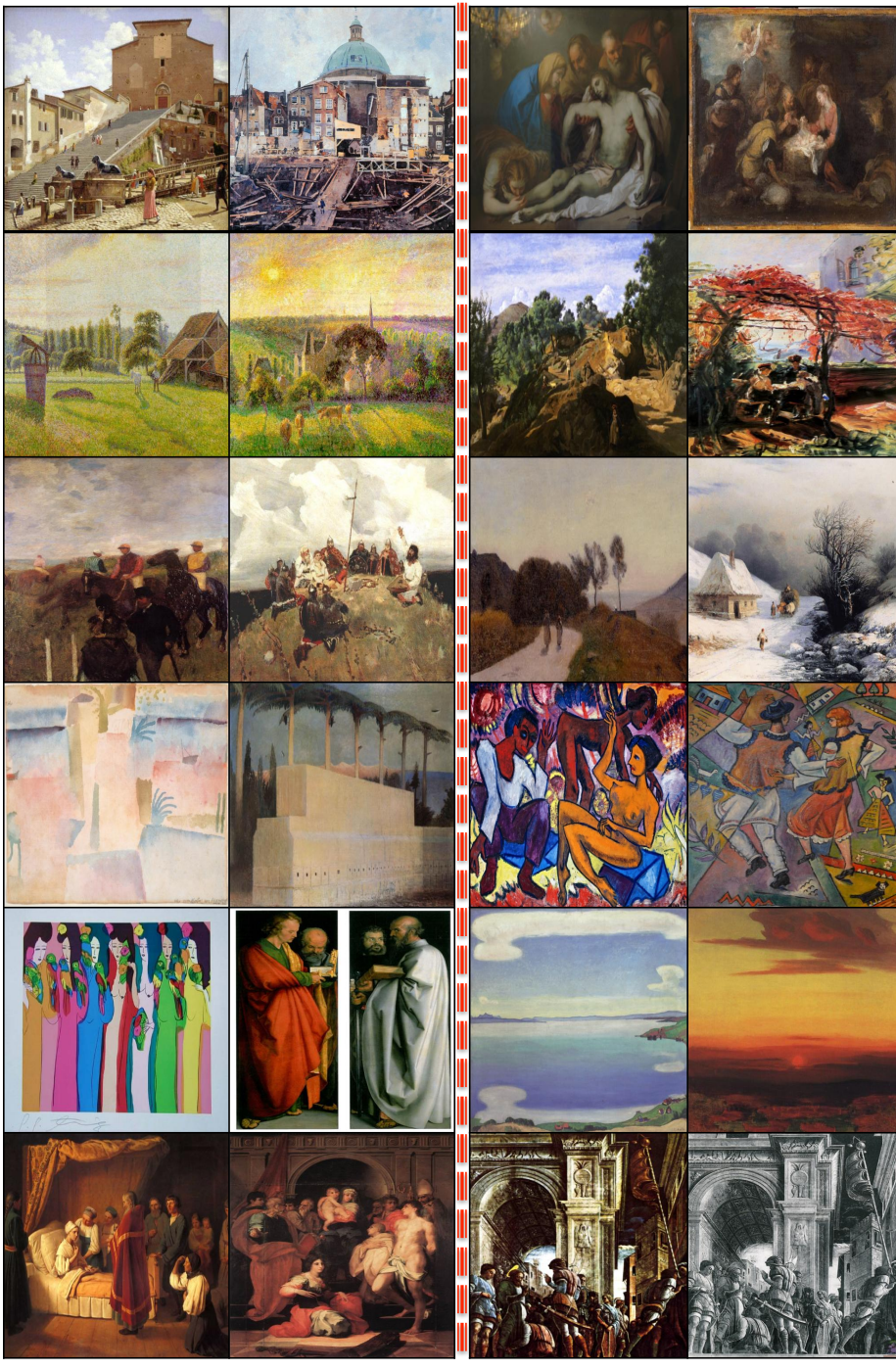

Fig. 8: Sample output for the tasks of image search. In each pair, the image on the left is the query and image on the right is the closest match, but not from the same style (LMNN plus feature fusion.)

Art name	Artist	Style	Art name	Artist	Style
The marble staircase which leads up to S. Maria in Aracoeli in Rome	Christoffer Wilhelm Eckersberg	Neoclassicism	Corner Paleissingel Straat In Amsterdam	Cornelis Vreedenburgh	Impressionism
Brickworks at Eragny	Camille Pissarro	Pointillism	Countryside and Eragny Church and Farm	Camille Pissarro	Impressionism
At the races	Edgar Degas	Impressionism	Bayan	Viktor Vasnetsov	Romanticism
View towards the port of Hammamet	Paul Klee	Cubism	Sacrificial stone in Baalbek	Tivadar Kosztka Csontvary	Post-Impressionism
Ladies in a row	Walasse Ting	Pop Art	The four Apostles	Albrecht Durer	Northern Renaissance
Communion of dying	Alexey Venetsianov	Realism	Madonna Enthroned and ten saints	Rosso Fiorentino	Mannerism (Late Renaissance)
The lamentation	Alexey Venetsianov	Realism	Adoration of the shepherds	Bartolome Esteban Murillo	Baroque
A chestnut wood among the rocks	Camille Corot	Realism	Artist's children in garden	Max Slevogt	Impressionism
A road in the countryside, near lake leman	Camille Corot	Realism	Little Russian ox cart in winter	Ivan Aivazovsky	Romanticism
Hungarian gipsies	Endre Bartos	Expressionism	Composition with Romanian motifs	Corneliu Michailescu	Cubism
Lake Geneva from Chexbres	Ferdinand Hodler	Post-Impressionism	Sunset in the winter. A coast of the sea	Arkhip Kuindzhi	Impressionism
St. Jacques leads to martyrdom	Andrea Mantegna	High Renaissance	St. James the Great on his way to execution (painted on : 1448)	Andrea Mantegna	Early Renaissance

Table 7: Annotation of paintings in Figure 8. Each row corresponds to one pair of images, labeled with the name of painting, its style and its artist. First six rows correspond to the six pairs on the left in Figure 8 and next six rows correspond to the pairs on the right.

Bibliography

[1] A. E. Abdel-Hakim and A. A. Farag. Csift: A sift descriptor with color invariant characteristics. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2006.

[2] R. Arnheim. *Visual thinking*. Univ of California Press, 1969.

[3] R. S. Arora and A. M. Elgammal. Towards automated classification of fine-art painting style: A comparative study. In *ICPR*, 2012.

[4] Y. Bar, N. Levy, and L. Wolf. Classification of artistic styles using binarized features derived from a deep neural network. 2014.

[5] A. Bentkowska-Kafel and J. Coddington. *Computer Vision and Image Analysis of Art: Proceedings of the SPIE Electronic Imaging Symposium, San Jose Convention Center, 18-22 January 2010*. PROCEEDINGS OF SPIE. 2010.

[6] I. E. Berezhnoy, E. O. Postma, and H. J. van den Herik. Automatic extraction of brushstroke orientation from paintings. *Machine Vision and Applications*, 20(1):1–9, 2009.

[7] A. Bergamo and L. Torresani. Classemes and other classifier-based features for efficient object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1, 2014.

[8] A. Bergamo, L. Torresani, and A. W. Fitzgibbon. Picodes: Learning a compact code for novel-category recognition. In *Advances in Neural Information Processing Systems*, pages 2088–2096, 2011.

[9] G. Carneiro, N. P. da Silva, A. D. Bue, and J. P. Costeira. Artistic image classification: An analysis on the printart database. In *ECCV*, 2012.

[10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *International Conference on Computer Vision & Pattern Recognition*, volume 2, pages 886–893, June 2005.

[11] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *ICML*, 2007.

[12] M. V. Fahad Shahbaz Khan, Joost van de Weijer. Who painted this painting? 2010.

[13] L. Fichner-Rathus. *Foundations of Art and Design*. Clark Baxter, 2008.

[14] J. Goldberger, S. Roweis, G. Hinton, and R. Salakhutdinov. Neighbourhood components analysis. In *NIPS*, 2004.

[15] C. R. Johnson, E. Hendriks, I. J. Berezhnoy, E. Brevdo, S. M. Hughes, I. Daubechies, J. Li, E. Postma, and J. Z. Wang. Image processing for artist identification. *Signal Processing Magazine, IEEE*, 25(4):37–48, 2008.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[18] J. Li and J. Z. Wang. Studying digital imagery of ancient paintings by mixtures of stochastic models. *Image Processing, IEEE Transactions on*, 13(3):340–353, 2004.

[19] J. Li, L. Yao, E. Hendriks, and J. Z. Wang. Rhythmic brushstrokes distinguish van gogh from his contemporaries: Findings via automated brushstroke extraction. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2012.

[20] T. E. Lombardi. The classification of style in fine-art painting. *ETD Collection for Pace University. Paper AAI3189084.*, 2005.

[21] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 2004.

[22] S. Lyu, D. Rockmore, and H. Farid. A digital technique for art authentication. *Proceedings of the National Academy of Sciences of the United States of America*, 101(49):17006–17010, 2004.

[23] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *IJCV*, 2001.

[24] G. Polatkan, S. Jafarpour, A. Brasoveanu, S. Hughes, and I. Daubechies. Detection of forgery in paintings using supervised learning. In *16th IEEE International Conference on Image Processing (ICIP)*, 2009.

[25] R. Sablatnig, P. Kammerer, and E. Zolda. Hierarchical classification of paintings using face- and brush stroke models. 1998.

[26] B. Saleh, K. Abe, and A. Elgammal. Knowledge discovery of artistic influences: A metric learning approach. In *ICCC*, 2014.

[27] C. Shen, J. Kim, L. Wang, and A. van den Hengel. Positive semidefinite metric learning using boosting-like algorithms. *Journal of Machine Learning Research*, 13:1007–1036, 2012.

[28] D. G. Stork. Computer vision and computer graphics analysis of paintings and drawings: An introduction to the literature. In *Computer Analysis of Images and Patterns*, pages 9–24. Springer, 2009.

[29] L. Torresani, M. Szummer, and A. Fitzgibbon. Efficient object category recognition using classemes. In *ECCV*, 2010.

[30] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. *CoRR*, abs/1412.4564, 2014.

[31] K. Weinberger and G. Tesauro. Metric learning for kernel regression. In *Eleventh international conference on artificial intelligence and statistics*, pages 608–615, 2007.

[32] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *JMLR*, 2009.