axioms

MDPI

Article

Measure of Similarity between GMMs Based on Autoencoder-Generated Gaussian Component Representations

Vladimir Kalušev [1], Branislav Popović [2], Marko Janev [3], Branko Brkljač [2,*] and Nebojša Ralević [2]

[1] The Institute for Artificial Intelligence Research and Development of Serbia, Fruškogorska 1, 21000 Novi Sad, Serbia; vladimir.kalusev@ivi.ac.rs
[2] Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia; bpopovic@uns.ac.rs (B.P.); nralevic@uns.ac.rs (N.R.)
[3] Institute of Mathematics, Serbian Academy of Sciences and Arts, Kneza Mihaila 36, 11000 Belgrade, Serbia; markojan@uns.ac.rs
* Correspondence: brkljacb@uns.ac.rs

Abstract: A novel similarity measure between Gaussian mixture models (GMMs), based on similarities between the low-dimensional representations of individual GMM components and obtained using deep autoencoder architectures, is proposed in this paper. Two different approaches built upon these architectures are explored and utilized to obtain low-dimensional representations of Gaussian components in GMMs. The first approach relies on a classical autoencoder, utilizing the Euclidean norm cost function. Vectorized upper-diagonal symmetric positive definite (SPD) matrices corresponding to Gaussian components in particular GMMs are used as inputs to the autoencoder. Low-dimensional Euclidean vectors obtained from the autoencoder's middle layer are then used to calculate distances among the original GMMs. The second approach relies on a deep convolutional neural network (CNN) autoencoder, using SPD representatives to generate embeddings corresponding to multivariate GMM components given as inputs. As the autoencoder training cost function, the Frobenious norm between the input and output layers of such network is used and combined with regularizer terms in the form of various pieces of information, as well as the Riemannian manifold-based distances between SPD representatives corresponding to the computed autoencoder feature maps. This is performed assuming that the underlying probability density functions (PDFs) of feature-map observations are multivariate Gaussians. By employing the proposed method, a significantly better trade-off between the recognition accuracy and the computational complexity is achieved when compared with other measures calculating distances among the SPD representatives of the original Gaussian components. The proposed method is much more efficient in machine learning tasks employing GMMs and operating on large datasets that require a large overall number of Gaussian components.

Keywords: Gaussian mixture models; autoencoders; KL divergence; classification; machine learning

MSC: 68T10; 94A17; 62B10; 68T30; 68T45; 68P30; 65C60; 62G07

Citation: Kalušev, V.; Popović, B.; Janev, M.; Brkljač, B.; Ralević, N. Measure of Similarity between GMMs Based on Autoencoder-Generated Gaussian Component Representations. *Axioms* **2023**, *12*, 535. https://doi.org/10.3390/axioms12060535

Academic Editor: Simeon Reich

Received: 28 April 2023
Revised: 26 May 2023
Accepted: 26 May 2023
Published: 30 May 2023

1. Introduction

Gaussian mixture models have significant applications in various machine learning (ML) tasks, including computer vision and pattern recognition [1–8], as well as context-based image matching and texture recognition, since an arbitrary probability density function (PDF) could be successfully modeled by GMM, knowing the exact number of modes of that particular PDF. They can also be applied in a broad class of artificial intelligence (AI) tasks, such as automatic speech recognition [9], speaker verification and recognition [10], age and gender recognition [11], or used as input and/or output data representations in deep learning [12], and within an emerging field of variational autoencoders [4]. As many

of those systems are complex, i.e., they involve large amounts of high-dimensional multivariate GMM data representatives, developing less computationally demanding and, at the same time, reasonably accurate GMM similarity measures has become a crucial issue.

Current solutions can be divided into two main categories. The first and the main direction is based on the utilization of information distances between PDFs, as characteristics of information sources. This means that information distances can have various levels of computational complexity and can capture different aspects of the underlying PDFs that are driving the generation of observations from the considered information sources. Thus, the choice of a particular information distance is a matter of the level of computational complexity that is considered as acceptable, the availability and the amount of collected data from information sources, and the desired level of information distance effectiveness in solving a particular ML task. Early works are represented by [13–15], which are based on Chernoff, Bhattacharyya, and Matusita information distances, and the most effective Kullback–Leibler (KL) divergence [16,17], as given in [2], or [18,19]. The Bhattacharyya distance [14] is a special case of a Chernoff α-divergence [20], while the family of Chernoff-α divergences is related to the family of Rényi α-divergences by a simple relation described in [20]. On the other hand, Rényi α-divergences come from the Rényi entropy that generalizes the notion of the most commonly used Shannon entropy. Thus, for $\alpha = 1$, the Rényi entropy results in the Shannon entropy and consequently the Kullback–Leibler (KL) divergence, as the special case of a more general Rényi α-divergence, i.e., the corresponding cross-entropy. The most commonly used and adopted solution among the aforementioned solutions is the KL divergence, but the distance effectiveness in measuring the similarity between the considered probability distributions always depends on the specific task and conditions.

Although the KL divergence exists in the close (i.e., analytical) form as the information distance for the case when PDFs are two multivariate Gaussian components, there is no closed-form solution for the case of GMMs. Various methods approximating the KL divergence between GMMs have been developed, approximating the KL divergence as the KL divergence between their Gaussian components, while retaining acceptably low bounds on the approximation error, as well as high recognition accuracy [2,19]. For example, in [7], a straightforward, although unacceptably computationally expensive solution (especially when dealing with huge amounts of data and the large dimensionality of the underlying feature space) is proposed, based on the Monte-Carlo method calculation of the KL divergence between two GMMs. In the same work, lower and upper approximation bounds are delivered, including experiments conducted on synthetic data, as well as in speaker verification tasks. In [2], another approximation of the KL divergence is proposed and applied in an image retrieval task. In [1], the unscented transform is delivered and applied within a speaker recognition task. Besides these, we also highlight the applications of the KL divergence in the specific tasks of game design, where it can be utilized to establish variational autoencoder (VAE) models that generate new game levels [21], or to compare and analyze the game-levels [22], e.g., as the objective function of an evolutionary algorithm to evolve new levels.

The second group of methods relying on Earth mover's distance (EMD) also emerged [19,23], all based on the baseline Euclidean EMD method proposed in [24], designed to compare the cross bins of the histogram forms of one-dimensional PDFs. Either the KL divergence or some other information distance between the Gaussian components of GMMs is used as the actual ground or geometry-based distance between the symmetric positive definite (SPD) representations of GMM Gaussians obtained along the geodesic of Riemannian geometry of the SPD manifold, denoted as $\mathrm{Sym}_{++}(d)$. Nevertheless, the computational complexity of all those methods is of the order of $O(d^3)$, due to the computational complexity of covariance matrix inversion, where $d \in \mathbb{R}$ is the dimension of underlying feature space, i.e., it is computationally very expensive in the case of large GMMs, as well as large feature space dimensionality d, which is inherent to real application scenarios. The reduction in computational complexity is crucial in such systems, so various solu-

tions are able to significantly reduce the computational complexity without significantly decreasing the recognition accuracy and addressing the mentioned issue were proposed. Some of these EMD-based measures, as can be seen in [25–27], use the graph Laplacian-based procedures, modified to operate on the $\mathrm{Sym}_{++}(d)$ cone instead of $\mathbb{R}^d$, for example, the LPP [28] and NPE [29] dimensionality reduction, i.e., manifold learning techniques. Nevertheless, all mentioned embedding-based GMM similarity measures are based on the shallow ML models, as well as linear projection operators, which transform features from the original high-dimensional parameter space in which SPD representatives lie as elements of the $Sym_{++}(d)$ cone, to a low-dimensional Euclidean space of the transformed parameters, as used in the calculation formulas of GMM measures. Namely, in contrast to PCA, neighborhood preserving embedding (NPE) performs dimensionality reduction by preserving the local neighborhood structure on the data manifold by constructing an adjacency graph, computing the neighborhood graph weights by solving an optimization problem, and finally determining one "optimal" projection that best fits the manifold structure. In comparison to locality preserving projections (LPP), the objectives are different. On the other hand, in [25], graph weights in the applied LPP procedure are calculated using the KL information distance between the positive definite matrices, i.e., by using Lovric's positive definite embeddings of Gaussian components from all GMMs in a particular ML task. Thus, it is an extension of LPP to the manifold of $\mathrm{Sym}_{++}(d)$, utilized to compute the low-dimensional Euclidean representatives (vectors). Similarly, in [26], the NPE-like procedure, again involving Lovric's positive definite embeddings, results in obtaining the MAXDET optimization problem [30] that computes the neighborhood graph weights. Finally, the dimensionality reduction projection operator proposed in [26] is obtained in the same manner as in the LPP case. In both cases [25,26], the low-dimensional embeddings are then used to construct the proposed similarity measure between GMMs.

In this paper, a novel GMM-DeepAutoenc similarity measure between GMMs is proposed. It is based on similarities between the lower-dimensional representations (or representatives) of GMM components, obtained by utilizing deep autoencoder architectures, i.e., deep rather than shallow ML encoding approaches. Since nonlinear deep neural networks (DNNs) are used in such autoencoding embedding procedures, these allow representations of the original SPD matrices of GMM components in the low-dimensional and nonlinear submanifolds embedded in a $Sym_{++}(d)$ cone. Actually, we construct the low-dimensional Euclidean embeddings by the use of nonlinear autoencoder networks, which enable us to better incorporate the nonlinear structure of the manifold (even non-regular) into the feature vectors. In comparison to the previously discussed LPP, NPE, and DPLM-based GMM similarity measures reported in [25–27], the low-dimensional representatives used in this paper are obtained by a nonlinear projective procedure performed by the designed autoencoder, which can be broadly interpreted as the mapping to the tangent space of each particular element or sample, instead of having one common projection hyperplane for dimensionality reduction. Thus, at the roughly similar computational complexity, the method proposed in this paper can be expected to achieve higher effectiveness, i.e., a better trade-off between computational complexity and recognition accuracy.

Two different approaches regarding deep autoencoder architectures have been explored and utilized to obtain the low-dimensional representatives corresponding to the individual Gaussian components of GMMs.

The first approach uses a classical deep autoencoder architecture, utilizing the Euclidean norm between the input and output vectors as the model training cost function. Initially, vectorized upper-diagonal SPD matrices corresponding to individual Gaussian components in GMMs are obtained using the embedding procedure reported in [31] (as can also be seen in [19,23]), and fed as the training inputs to the autoencoder. Afterwards, during the encoder deployment, the low-dimensional Euclidean vectors obtained from the middle layer of the autoencoder are exploited to calculate the ground distances in the proposed similarity measure between GMMs in some particular machine learning tasks.

The second approach uses a deep CNN autoencoder (as can be seen in [32–34]), to which the same type of embedded SPD representatives of GMM Gaussian components are applied. However, the Frobenious matrix norm between the two-dimensional matrix input and autoencoder matrix output is employed as the cost function used in the learning procedure. Moreover, as a regularizer in the actual training, various pieces of information, as well as Riemannian manifold-based distances between the SPD matrices belonging to $\mathrm{Sym}_{++}(d+1)$ and corresponding to the statistics of the computed autoencoder feature maps, are also used, where $d \in \mathbb{R}$ is the dimension of the underlying Euclidean feature space (which is space of the original GMM components). Thus, the low-dimensional Euclidean representatives of original GMM components can be obtained, where the autoencoder embedding procedure is learned by the additional enforcing of similarities among the PDFs, i.e., the statistics of the feature space observations (computed autoencoder feature maps) corresponding to the encoder and the decoder network, respectively. Since the obtained embedded representatives of the original GMM components are low-dimensional Euclidean vectors, the computational complexity of calculating distances between them is of the order of $O(d)$, while the computational complexity of various measures utilizing ground distances between the original SPD matrices (representatives denoting individual GMM components) is of the order of $O(d^3)$. As a result, in the machine learning tasks utilizing the proposed autoencoder-based GMM similarity measure (GMM-DeepAutoenc), a significantly better trade-off between the achieved recognition accuracy and the computational complexity can be obtained, when compared with the classical similarity measure methods. This was experimentally confirmed on various texture recognition tasks, as well as image matching datasets. In case of ML tasks that are characterized by the presence of GMMs with a large overall number of Gaussian components, such an approach would be much more efficient when compared with the existing ones. Moreover, when compared with other methods utilizing the low-dimensional representations of Gaussian components, i.e., when compared to shallow ML based embedding frameworks (see, e.g., [25–27]), the proposed deep autoencoder-based measure (GMM-DeepAutoenc) obtains better results in the mentioned tasks and for various datasets, especially in cases with a large number of categories.

2. Materials and Methods

2.1. Baseline GMM Similarity Measures

KL divergence, defined as $KL(p||q) = \int_{\mathbb{R}^d} p(x) \ln \frac{p(x)}{q(x)} dx$, is the most natural measure between two probability distributions p and q. Unfortunately, it can be expressed in a closed (analytical) form only for a limited number of distributions, and GMM is not one of these. Let us denote two GMMs as $f = \sum_{i=1}^{n} \alpha_i f_i$ and $g = \sum_{j=1}^{m} \beta_j g_j$, with $f_i = \mathcal{N}(\Sigma_i, \mu_i)$ and $g_j = \mathcal{N}(\Sigma_j, \mu_j)$ representing the Gaussian components of the corresponding mixtures. Where $\alpha_i > 0$, $\beta_j > 0$ are corresponding weights, satisfying $\sum_i \alpha_i = 1$, $\sum_j \beta_j = 1$, while μ_i, μ_j are the mean vectors of f_i, g_j, and Σ_i, Σ_j are their full covariance matrices. There are various proposed measures between GMMs, almost all of which are based on some specific approximation of the KL divergence. The straightforward and most computationally expensive measure (especially in real-world applications where there are high feature space dimensionalities) is based on using the standard Monte Carlo method (see [35]). The idea is to sample the probability distribution f by using i.i.d. samples x_i, $i = 1, \ldots, N$, such that $E_f\left[\ln \frac{f(x)}{g(x)}\right] = KL(f||g)$. This is given by:

$$KL_{MC}(f||g) \approx \frac{1}{N} \sum_{i=1}^{N} \ln \frac{f(x_i)}{g(x_i)}. \tag{1}$$

Many of the proposed measures are based on approximations utilizing KL divergence between two Gaussian components $KL(f_i||g_j)$, which exist in the closed form given by

$$KL(f_i||g_j) = \ln \frac{|\Sigma_{f_i}|}{|\Sigma_{g_j}|} + Tr\left[\Sigma_{g_j}^{-1}\Sigma_{f_i}\right] + (\mu_{f_i} - \mu_{g_j})^T \Sigma_{g_j}^{-1}(\mu_{f_i} - \mu_{g_j}) - d \tag{2}$$

The roughest approximation (by an upper bound) is based on the convexity of the KL divergence [36]. Thus, for two GMMs $f = \sum_{i=1}^{n} \alpha_i f_i$ and $g = \sum_{j=1}^{m} \beta_j g_j$, it holds that the weighted average approximation is given by:

$$KL(f||g) \leq KL_{WA}(f||g) = \sum_{i,j} \alpha_i \beta_j KL(f_i||g_j), \tag{3}$$

where $f_i = \mathcal{N}(\Sigma_i, \mu_i)$ and $g_j = \mathcal{N}(\Sigma_j, \mu_j)$ are the Gaussian components of the corresponding mixtures, while $\alpha_i > 0$, $\beta_j > 0$ are the corresponding weights, satisfying $\sum_i \alpha_i = 1$, $\sum_j \beta_j = 1$. We assume that the KL divergences $KL(f_i||g_j)$ between the corresponding Gaussian components exist in the closed-form, e.g., given by (2).

Matching-based (MB) approximation, proposed in [2] as:

$$KL_{MB}(f||g) \approx \sum_i \alpha_i \left[\min_j KL(f_i||g_j) + \log\left(\frac{\alpha_i}{\beta_j}\right)\right] \tag{4}$$

was based on the assumption that the element g_j in g that is the most proximal to f_i dominates the integral $\int f_i \log g$ in $KL(f||g)$. Motivated by (4), the more efficient matching-based approximation can be obtained by

$$KL_{MBS}(f||g) \approx \sum_i \alpha_i \min_j KL(f_i||g_j), \tag{5}$$

which shows good performance in the case when components in f and g are far apart, but is inappropriate when there is significant overlap between them.

In [37], the unscented transform-based approximation was proposed in order to deal with the described overlapping situations. This is a method for calculating the statistics of a random variable that undergoes a nonlinear transformation. As it holds $KL(f||g) = \int_{\mathbb{R}^d} f \ln f - \int_{\mathbb{R}^d} f \ln g$, the unscented transform approximates $\int_{\mathbb{R}^d} f_i \ln g$ as:

$$\begin{aligned} &\int_{\mathbb{R}^d} f_i \ln g \approx \frac{1}{2d} \sum_{k=1}^{2d} \ln g(x_{i,k}) \\ &x_{i,k} = \mu_i + \left(\sqrt{\Sigma_i}\right)_k, \; k = 1, \ldots, d \\ &x_{i,d+k} = \mu_i - \left(\sqrt{\Sigma_i}\right)_k, \; k = 1, \ldots, d, \end{aligned} \tag{6}$$

where $\left(\sqrt{\Sigma_i}\right)_k$ is the k-th column of the matrix square root of Σ_i. In the case of $\int_{\mathbb{R}^d} f \ln g$, we obtain:

$$\int_{\mathbb{R}^d} f \ln g \approx \frac{1}{2d} \sum_{i=1}^{n} \alpha_i \sum_{k=1}^{2d} \ln g(x_{i,k}),$$

thus acquiring $KL_{UC}(f||g)$. In [35], the approximation based on variational bound is proposed as:

$$KL_{VB}(f||g) = \sum_i \alpha_i \frac{\sum_{\hat{i}} \alpha_{\hat{i}} e^{-KL(f_i||f_{\hat{i}})}}{\sum_j \beta_j e^{-KL(f_i||g_j)}}. \tag{7}$$

In order to obtain an approximate *KL* divergence between two GMMs, all these approximations utilize KL divergences between Gaussian components that are given in the closed-form expression. On the other hand, Earth mover's distance (EMD)-based GMM distances stem from the Earth mover's methodology, which emerges in various recognition tasks, such as those reported in [3,24,38]. In [19], textures are classified using EMD to measure the distributional similarity of sets of Gaussian components that represent the categories, while in [23], the ground distances between Gaussian components given by (2) are incorporated within a sparse EMD-based SR-EMD procedure [23]. The authors utilized various ground distances based on the Riemannian geometry, as well as in combination with the Gaussian component embedding reported in [31]. The best performance among these methods was achieved by the Riemannian metric defined on the Riemannian manifold representing the product of Lie groups $\mathbb{R}^d$ and $\mathrm{Sym}_{++}(d)$. According to [23], such a ground distance measure between the Gaussian component f_i and g_j can be motivated by the product of the mentioned Lie groups and defined as:

$$d_{f_i,g_j}(\xi) = (1-\xi)a_{f_i,g_j} + \xi b_{f_i,g_j}, \tag{8}$$

where the individual components of measure $d_{f_i,g_j}(\xi)$ are defined by the following expressions:

$$a_{f_i,g_j} = \left((\mu_{f_i} - \mu_{g_j})^T(\Sigma_{f_i}^{-1} + \Sigma_{g_j}^{-1})(\mu_{f_i} - \mu_{g_j})\right)^{\frac{1}{2}},$$
$$b_{f_i,g_j}(\Omega) = \|\Omega\big(\ln(\Sigma_{f_i}) - \ln(\Sigma_{g_j})\big)\|_F.$$

We note that the matrix Ω denotes the result of the decomposition of the matrix M that represents the corresponding matrix Mahalanobis norm, i.e., $M = \Omega^T\Omega$ and:

$$b_{f_i,g_j}(\Omega) = tr\Big(\big(\ln(\Sigma_{f_i}) - \ln(\Sigma_{g_j})\big)^T M\big(\ln(\Sigma_{f_i}) - \ln(\Sigma_{g_j})\big)\Big). \tag{9}$$

Thus, when $d_{f_i,g_j}(\xi)$ is incorporated into a sparse representation EMD procedure (SR-EMD), one obtains the similarity measure between GMMs as the minimum value of the cost function of the optimization problem given by:

$$\min_{M,x_{pq}} \sum_{p=1}^{m_f}\sum_{q=p+1}^{m_g}\alpha_{pq}\Big(\|y_{pq} - A_{pq}C_{pq}^{-1}(M)x_{pq}\|_{l_2}^2 + \rho\|x_{pq}\|_{l_1}\Big). \tag{10}$$
$$s.t.\ M \succ 0, \forall x_{pq} \succeq 0$$

The described SR-EMD problem involves the joint optimization of the ground distance matrix $C_{pq}(M)$ and vector x_{pq}, which formulate an alternating dictionary learning and sparse representation procedure. Thus, the optimal transport vector x_{pq} is found under the constraints vector $y_{pq} = \Big[\alpha_1, \ldots, \alpha_{m_f}, \beta_1, \ldots, \beta_{m_g}\Big]$, defined by the weights of all $(m_f + m_g)$ GMM components, and some fixed $\rho > 0$. We note that the term A_{pq} represents the $(m_f + n_g) \times m_f m_g$ matrix with 0 and 1 entries. In (8)–(10), terms $\|\cdot\|_{l_1}$, $\|\cdot\|_{l_2}$, $\|\cdot\|_F$, represent l_1, l_2 vector norms, and matrix Frobenius norm F, respectively.

Recently, approaches for GMM similarity measurements based on dimensionality reduction have been proposed, as can be seen in [25,26], where each embedded SPD representative corresponding to some Gaussian component of GMMs (obtained by using [31]) is replaced by its low-dimensional Euclidian representation (projection). These exploit the shallow ML approach based on graph manifold learning (LPP in [25] or NPE in [26]), which is used in order to obtain the linear projectors, but with kernels containing symmetrized KL divergence between SPD representatives, instead of the Euclidian norm. Similarly, in [27], the dimensionality reduction preceding GMMs' similarity measurements was also applied using the shallow ML approach proposed in [39] in order to obtain the low-dimensional SPD representatives of the Gaussian components. Some standard KL divergence-based methods (for example [1,2,7]) were then used in order to obtain the similarity measures

between GMMs, but now with low-dimensional SPDs. We refer to this type of method as DPLM, inspired by the title of [39], where the corresponding dimensionality reduction based on the distance preservation to the local mean (DPLM) was proposed. Finally, the GMM similarity measure in all [25–27] is defined using a fuzzy aggregation of Euclidian distances between low-dimensional representatives, or using an EMD-based distance metric (see [38]) between the sets of the corresponding low-dimensional representatives. Concerning the DPLM-based GMM similarity measures that we use as the baselines in Section 3 with experimental results, and also discuss in Section 2.3 regarding computational complexity, we will denote these by $uDPLM_{MB}$, $uDLPM_{WA}$, and $uDLPM_{VB}$ throughout the paper. The prefix u indicates that these are unsupervised approaches, as can be seen in [27], while the suffixes WA, MB, and VB denote that the similarity between newly obtained GMMs with low-dimensional SPD matrices (Lovric embeddings) is measured using the standard KL divergence approximation measures given by (3), (4), and (7), respectively. From the results comparing CPU times and recognition accuracies in [27], it can be seen that, by the dimensionality reduction applied in [27], a significantly better trade-off between the recognition accuracy and the computational complexity is achieved when compared with the baseline approaches [1,2,7].

2.2. GMM Similarity Measures Based on Autoencoder-Generated Representations

GMM similarity measures based on the dimensionality reduction, as reported in [25–27], are all shallow ML methods utilizing linear approximations, i.e., projectors in order to approximate the underlying sub-manifold of manifold $\mathrm{Sym}_{++}(d+1)$ on which the parameters of the SPD representatives (Gaussian components) lie, which is an assumption of the underlying approximation problem. Moreover, all these methods, in their own way, learn linear projectors, that fit the sub-manifold defined by the SPD representatives of the Gaussian components of GMMs in some particular ML problem (i.e., the SPD representatives obtained using the Lovric's embedding proposed in [31]). However, in this work, we propose a novel approach whereby, using the deep autoencoder network architecture, we learn to encode the mentioned SPD representatives of Gaussian components into the middle layer of the autoencoder, and thus obtain the embedding of SPD matrices into the low-dimensional Euclidean space. If provided with a sufficient amount of learning data, the designed nonlinear embedding aims to obtain an even better trade-off between recognition accuracy and computational complexity. As a result, by using a deep autoencoder network architecture, we can model nonlinear, and even non-regular sub-manifold structures in $\mathrm{Sym}_{++}(d+1)$.

2.2.1. Autoencoder Architectures

In this work, we use deep autoencoder network architectures to construct the novel GMM similarity measure that is computationally efficient, and at the same time, retains recognition accuracy. The idea is that, by using a deep autoencoder, we can more efficiently "capture" and characterize the nonlinear sub-manifold that contains the parameters of GMMs used in a particular ML application. Thus, the nonlinear dimensionality reduction performed by the encoding process of the autoencoder network that implicitly performs the mentioned sub-manifold characterization. Such a learning process and consequent dimensionality reduction lead to achieving a more computationally efficient GMM similarity measurement at the same or better recognition accuracy, i.e., a better trade-off between these requirements if a sufficient amount of training data are available.

The autoencoder network comprises nonlinear mappings for the encoder and the decoder, learned on the set of input SPD matrices or their vectorized counterparts. After the autoencoder training, we calculate the proposed GMM similarity measure between two GMMs by aggregating the Euclidean distances between the obtained low-dimensional representations of GMM components, i.e., the low-dimensional representatives of input SPD matrices that correspond to Gaussian components. Described aggregation is performed by various fuzzy aggregation measures, as in [25–27], or by using the EMD optimal transport

problem described in [24]. We note that in the presented experiments, Gaussian components are used to characterize both training and testing samples, which are instances of the categories in the actual ML task.

We use two different types of deep autoencoders. The first one is the classical fully connected, regularized deep autoencoder, Figure 1.

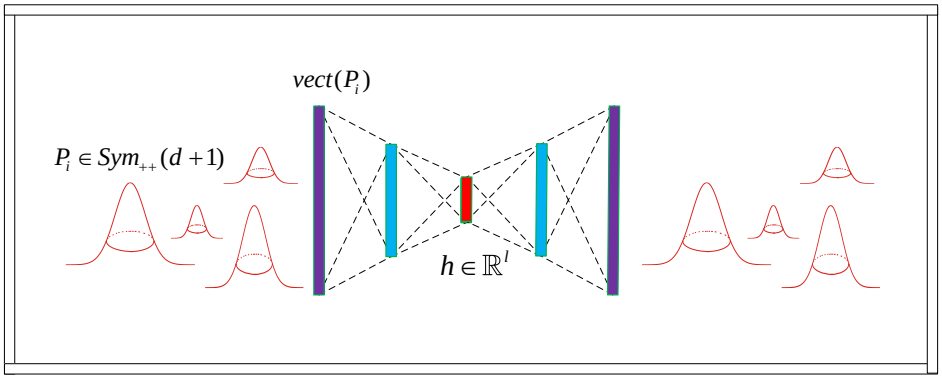

Figure 1. Illustration of the proposed fully connected autoencoder architecture for the low-dimensional embedding of Gaussian components represented by $vect(P_i)$ into $h \in \mathbb{R}^l$. Colors indicate symmetric architecture of the network and the goal of learning unique embedding of the original Gaussian components (middle vector h shown in red).

This usually comprises two sections—an encoder that maps the input data into a lower-dimensional representation, and a subsequent decoder that maps the output of the encoder to the same space as the input data. Furthermore, it is usually built to have symmetrical encoder and decoder layers. Namely, if we denote the encoder and the decoder networks by F and G, respectively, where $n = (d+1)(d+2)/2$ is the size of the input layer, and we supply the vectorized upper triangular $vect(P)$ of SPD representative $P \in \mathrm{Sym}_{++}(d+1)$ obtained by the Lovric embedding [31] of Gaussian component $g = \mathcal{N}(x; \mu, \Sigma) \in \{f_i\}_1^{n_f}$ of some GMM $f = \sum_{i=1}^{n_f} \alpha_i f_i$ in $\mathbb{R}^d$, which is given by:

$$g \hookrightarrow P = |\Sigma|^{-\frac{1}{d+1}} \begin{bmatrix} \Sigma + \mu\mu^T & \mu \\ \mu^T & 1 \end{bmatrix}, \tag{11}$$

the autoencoder training procedure can be described by the following objective:

$$\hat{F}, \hat{G} = \mathrm{argmin}_{F,G}\, \mathbb{E}_{P\sim p(P)}\left[\|(G \circ F)(vect(P)) - vect(P)\|_{l_2}\right] + \lambda \mathcal{R}(F, G), \tag{12}$$

where $\mathbb{E}[\cdot]$ stands for the mathematical expectation, $\circ$ for the composition of the functions, $\|\cdot\|_{l_2}$ denotes the Euclidean l_2 norm, while $P \sim p(P)$ denotes that the SPD representatives P given by the Lovric embedding (11), with PDF given by p, and $\mathcal{R}(F, G)$ is a regularization term used for parameters of F and G. We recall that, for any matrix $B \in \mathbb{R}^{m\times n}$ and its vectorized version $vect(B) \in \mathbb{R}^{mn}$, obtained by the serialization of the matrix B along the vertical or horizontal dimensions, it holds that $\|B\|_F = \sum_{i=1}^{m}\sum_{j=1}^{n} b_{ij}^2 = \|vect(B)\|_{l_2}$. The term $\lambda > 0$ is a regularization parameter. Thus, we estimate the loss function in (12) as:

$$\widehat{\mathbb{E}}_{P\sim p(P)}[\|(G \circ F)(vect(P)) - P\|_F] = \frac{1}{M}\sum_{i=1}^{M}\left[\|(G \circ F)(vect(P_i)) - vect(P_i)\|_{l_2}\right], \tag{13}$$

where M stands for the overall number of SPD matrices that are constructed as Lovric-embedding matrices $P_i, i = 1, \ldots, M$, i.e., the SPD representatives of the Gaussian components of all GMMs that are available for the training of the autoencoder model (e.g., per each

data training instance, there is one GMM with several GMM components). For $\mathcal{R}(F,G)$, we use the l_2 regularizer defined by:

$$\mathcal{R}(F,G) = \sum_{l=1}^{L} \sum_{i=1}^{n_l} \sum_{j=1}^{k_l} \left(w_{ij}^{(l)}\right)^2, \tag{14}$$

where L denotes the overall number of hidden layers (including F and G), n_l and k_l are the output and the input size of layer l, respectively, while $w_{ij}^{(l)}$ denote all weights of the autoencoder network, corresponding to the l-th layer. Term $\lambda > 0$ is considered as fixed in the optimization problem (12).

For the second type of deep autoencoder architecture, we use a deep CNN autoencoder, where we observe the model inputs, i.e., the entries of some particular Lovric embedding representative P_i (2D SPD matrix), as "pixels" of some 2D image (as usual in vision-based CNNs). This embedding framework consists of the encoder and the decoder CNN networks F and G, respectively, as well as the middle layer $h \in \mathbb{R}^l$, as shown in Figure 2.

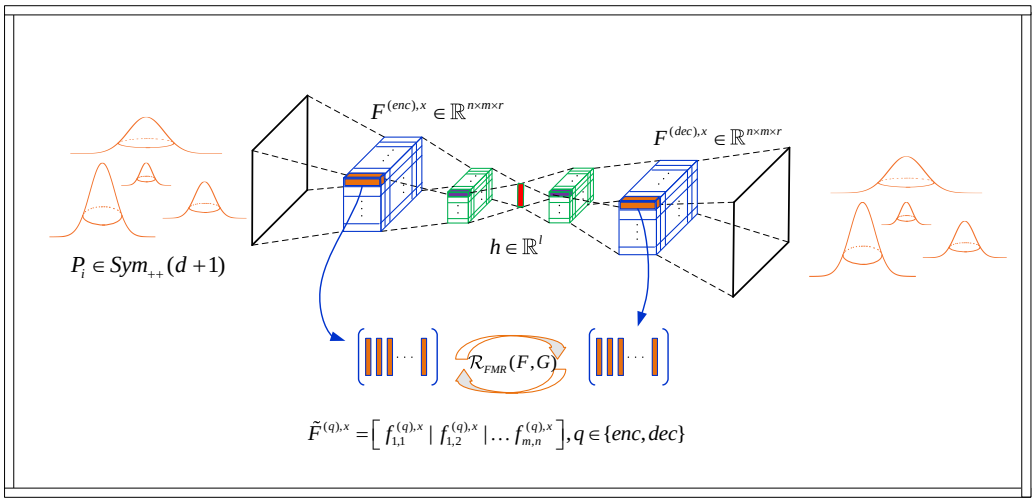

Figure 2. Illustration of the proposed FMR regularized CNN autoencoder for the low-dimensional embedding of Gaussian components represented by SPD matrices $P_i \in \mathrm{Sym}_{++}(d+1)$ into $h \in \mathbb{R}^l$.

The training procedure can be given by the same optimization problem (12) where we use the Frobenious matrix norm $\|\cdot\|_F$ instead of the Euclidean $\|\cdot\|_{l_2}$ in the objective, as we deal with matrices at the input (as well as at the output) of the CNN autoencoder. Nevertheless, it still requires the strong regularization provided by an additional cost term that forces the similarity between the corresponding feature map tensors in the encoder and the decoder, i.e., the outputs of the convolutional layers corresponding to the encoder and the decoder (between the first feature map of the encoder and the last feature map of the decoder, etc.). Namely, we add the feature map regularization (FMR) penalty term $\mathcal{R}_{FMR}$ in the form of a similarity measure between the PDFs that model statistics of the computed feature maps in F and G. Thus, we obtain the CNN autoencoder learning problem as follows:

$$\begin{aligned} \hat{F}, \hat{G} &= \mathrm{argmin}_{F,G}\, \mathbb{E}_{P\sim p(P)}[\|(G\circ F)(P) - P\|_F] + \lambda \mathcal{R}(F,G), \\ &+ \lambda_{FMR}\mathcal{R}_{FMR}(F,G) \end{aligned} \tag{15}$$

with some fixed $\lambda_{FMR} > 0$, where $\|\cdot\|_F$ is the Frobenius norm defined for $A = [a_{ij}] \in \mathbb{R}^{m\times n}$ as $\|A\|_F = \sum_{i=1}^{m}\sum_{j=1}^{n} a_{ij}^2$.

For measuring the feature map PDFs, i.e., formulating the $\mathcal{R}_{FMR}$, we consider some of the information-based, as well as the Riemannian manifold-based distances reported in [19,23]. For the low-dimensional Euclidean representation of SPD instances $P_i \in$

$\mathrm{Sym}_{++}(d+1)$ obtained by the Lovric embedding of Gaussian components, we take the vectors obtained from the middle layer of the deep autoencoder networks, i.e., bottleneck $h \in \mathbb{R}^l$, as described herein. These nonlinear projection quantities are then used to formulate the novel GMM-Autoenc similarity measures that we invoke.

2.2.2. Ground Distances Regularization

We present ground distances between multidimensional (k-dimensional) Gaussians, that we use as similarity measures in order to regularize the training of an autoencoder network, i.e., to form $\mathcal{R}_{FMR}(F,G)$. Let us consider k-dimensional $f = \mathcal{N}(x;\mu_1,\Sigma_1)$ and $g = \mathcal{N}(x,\mu_2,\Sigma_2)$, which are elements of $\mathrm{Sym}_{++}(k)$, or $k+1$-dimensional SPD Lovric embedding representatives given by (11) as elements of $\mathrm{Sym}_{++}(k+1)$, which are both Riemannian manifolds (see [19,23]), equipped with different Riemannian metrics, and inducing geodesic distances. Out of such ground distances that do not take into account the information geometric properties of multidimensional Gaussians, we consider the l_2-based distance:

$$d_{l_2}(f,g) = \|\Sigma_1 - \Sigma_2\|_F + \eta\|\mu_1 - \mu_2\|_{l_2}, \tag{16}$$

as well as the robust l_1-based distance given by:

$$d_{l_1}(f,g) = \|\Sigma_1 - \Sigma_2\|_{l_1} + \eta\|\mu_1 - \mu_2\|_{l_1}, \tag{17}$$

where $\|\cdot\|_{l_1}$ is the robust l_1 norm in $\mathbb{R}^k$. We note that the distance functions involving the $\|\cdot\|_{l_2}$ and $\|\cdot\|_F$ norm in (16) emerge from the assumption that the prior function, i.e., PDF, in the formulation of the regularization problem, is of the Gaussian type. These can be considered as more appropriate ones when there is a large amount of training data, while the ones based on the l_1 norm in (17), which correspond to the prior distribution of Laplace type, are usually used if we tend to obtain a more robust estimation.

Considering that there is a closed-form expression for the KL divergence between two arbitrary Gaussians f and g, denoted by $d_{KL}(f,g) = KL(f||g)$, with $KL(f||g)$ given by (2) (where the symbols are replaced by: $d \mapsto k$, $f_i \mapsto f$, $g_j \mapsto g$), for the information-based ground distance, we use the symmetrized version of the KL divergence given by:

$$d_{KL_{sym}}(f,g) = \frac{1}{2}(d_{KL}(f,g) + d_{KL}(g,f)). \tag{18}$$

Although the KL divergence is not symmetric, we note that it does not mean that it is antisymmetric. Namely, we give the following counterexample: for two discrete Bernoulli distributions $\mathcal{B}_\alpha(x)$ and $\mathcal{B}_\beta(x)$, parameterized by $\alpha = p$ and $\beta = 1-p$, respectively, it holds that $KL(\mathcal{B}_\alpha||\mathcal{B}_\beta) = KL(\mathcal{B}_\beta||\mathcal{B}_\alpha)$.

For the geometrical ground distance, we use Log-Euclidean metric as geodesic distance (see [40]) on $\mathrm{Sym}_{++}(k+1)$, forming the efficient geodesic distance invariant to similarity transformation, given by:

$$d_{le,nonsym}(P_f,P_g) = \|\ln(P_f) - \ln(P_g)\|_F,$$

for Lovric embedding SPD representatives $P_f, P_g \in \mathrm{Sym}_{++}(k+1)$ of f and g, respectively. Actually, we use the symmetrized version:

$$d_{le}(P_f,P_g) = \frac{1}{2}\Big(d_{le,nonsym}(P_f,P_g) + d_{le,nonsym}(P_g,P_f)\Big), \tag{19}$$

2.2.3. Forming the Feature Map Regularizer $\mathcal{R}_{FMR}(F,G)$

Let us denote the first feature map tensor of the encoder and the last feature map tensor of the decoder of the CNN autoencoder network, obtained for the input matrix $x \in \mathbb{R}^{(d+1)\times(d+1)}$, as $F^{(enc),x} \in \mathbb{R}^{n\times m\times r}$, and $F^{(dec),x} \in \mathbb{R}^{n\times m\times r}$, respectively, (both tensors are of the convolution layers type, and are of identical format, size $m \times n$ and depth r,

since the encoder and the decoder are assumed to be symmetric). In a general case, for the r-dimensional feature map or layer indexed by q, we will reshape or reformat the $F^{(q),x}$ 3D tensor into the following 2D matrix: $\tilde{F}^{(q),x} = \left[f_{1,1}^{(q),x} | f_{1,2}^{(q),x} | \dots f_{m,n}^{(q),x}\right]$, $q \in \{enc, dec\}$, where $f_{i,j}^{(q),x} \in \mathbb{R}^r$, $i \in \{1, \dots, n\}$, $j \in \{1, \dots, m\}$ are r-dimensional observations (or values) of the autoencoder feature maps in layer q (generated by the corresponding statistics of the autoencoder layer q with depth r).

Next, under the assumption that the underlying PDFs are r-dimensional multivariate Gaussians, i.e., that the unknown PDFs rendering the r-dimensional observations (values in the described feature map tensors) are normally distributed, $f_{i,j}^{(q),x} \sim \mathcal{N}(\Sigma^{(q),x}, \mu^{(q),x})$, we obtain the maximum likelihood estimates (MLEs) of their covariances and centroids, i.e., $\hat{\Sigma}^{(q),x}$, $\hat{\mu}^{(q),x}$, $q \in \{enc, dec\}$ from the columns of matrices $\tilde{F}^{(q),x}$, $q \in \{enc, dec\}$ as:

$$\begin{aligned} \hat{\Sigma}^{(q),x} &= \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} (f_{i,j}^{(q),x} - \mu^{(q),x})(f_{i,j}^{(q),x} - \mu^{(q),x})^T, \\ \hat{\mu}^{(q),x} &= \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j}^{(q),x}, \end{aligned} \tag{20}$$

Finally, we form the feature map regularizer $\mathcal{R}_{FMR}(F, G)$ in (15) as follows:

$$\mathcal{R}_{FMR}(F, G) = \mathbb{E}_{P \sim p(P)} \left[d_{gd}(f^P, g^P) \right] \tag{21}$$

where $f^P = \mathcal{N}(\Sigma^{(enc),P}, \mu^{(enc),P})$, $g^P = \mathcal{N}(\Sigma^{(dec),P}, \mu^{(dec),P})$, and the ground distances $gd \in \{l_2, l_1, KL_{sym}, le\}$ are all defined in the previous section. $P \in \mathrm{Sym}_{++}(r+1)$ stands for the SPD matrices generated by $p(P)$, i.e., by the distribution p of Lovric embedding SPD matrices representing the feature map Gaussians described by statistics in (20). Note that each of these Gaussians P, i.e., $(r+1) \times (r+1)$ SPD matrices made of $\hat{\Sigma}^{(q),x}$ and mean $\hat{\mu}^{(q),x}$ in (20), correspond to one of the original Gaussian components brought to the input of the autoencoder, i.e., one input x. In other words, estimates in (20) correspond to the autoencoder feature map observations in $\mathbb{R}^r$, while the input Gaussian components are in $\mathbb{R}^d$. Since it is assumed that there are M such $(d+1) \times (d+1)$ SPD matrices corresponding to the individual components of all GMMs present in the training phase of some particular ML task, we estimate the expectation in (21) as:

$$\mathbb{E}_{P \sim p(P)} \left[d_{gd}(f^P, g^P) \right] = \frac{1}{M} \sum_{i=1}^{M} d_{gd}(f^{P_i}, g^{P_i}), \tag{22}$$

Thus, M is the overall number of Gaussians of GMMs used for the autoencoder training, and each of the M input Gaussians results in feature map tensors $F^{(q),x}$, $q \in \{enc, dec\}$, i.e., their statistics in (20) are represent by the Lovric embedding $P \in \mathrm{Sym}_{++}(r+1)$ in (21).

2.2.4. Forming the GMM Similarity Measure Based on Autoencoder-Generated Representations

We form the proposed novel GMM-Autoenc similarity measures by following the approach similar to that described in [26]. Namely, the similarity measure between two particular GMMs is formed by aggregating the non-negative real values that represent the measure of similarity between corresponding components, i.e., their nonlinear embeddings. Actually, we have to compare the transformed "clouds" of lower l-dimensional Euclidean parameter vectors, with $l \ll n$, $n = (d+1)(d+2)/2$, which are computed from the original Gaussian components of two GMMs. In all approaches that we utilize, for the particular m_f-component GMM $f = \sum_{i=1}^{m_f} \alpha_i f_i$ with $f_i = \mathcal{N}(\mu_i, \Sigma_i)$, we represent GMM by the unique tupple $F = (v_1, \dots, v_{m_f}, \alpha_1, \dots, \alpha_{m_f})$, which consists of the embedding vectors v_i corresponding to the low-dimensional outputs from the middle (bottleneck)

layer of the trained embedding autoencoder described in Section 2.2.1. In this process, the belonging mixture weights α_i are also taken into account. We note that the autoencoder inputs are the Lovric embeddings of f_i, defined by (11), i.e., SPD matrices P_i or their vectorized counterparts.

Thus, the similarity measure between two GMMs, denoted by $f = \sum_{i=1}^{m_f} \alpha_i f_i$ and $g = \sum_{j=1}^{m_g} \beta_j g_j$, is introduced as the similarity between the corresponding representatives $F = \left(v_1, \ldots, v_{m_f}, \alpha_1, \ldots, \alpha_{m_f}\right)$ and $G = \left(u_1, \ldots, u_{m_g}, \beta_1, \ldots, \beta_{m_g}\right)$, respectively. It is measured in the autoencoder embedding space, as represented by the weighted low-dimensional Euclidean vectors $v_i, u_i \in \mathbb{R}^l$. By doing so, we utilize two different approaches. The first one employs a form of fuzzy union or intersection operation (see, e.g., [41]). Namely, we use the weighted max–min operator as follows:

$$\begin{aligned} p_i &= \min\{\beta_j \|v_i - u_j\|_2 \mid j = 1, \ldots, m_f\} \\ a &= \max\{\alpha_i p_i \mid i = 1, \ldots, m_g\} \\ q_j &= \min\{\alpha_i \|v_i - u_j\|_2 \mid i = 1, \ldots, m_g\} \\ b &= \max\{\beta_j q_j \mid j = 1, \ldots, m_f\} \\ D_1(F, G) &= \frac{1}{2}(a + b), \end{aligned} \tag{23}$$

as well as the maximum of the positive weighted sums

$$\begin{aligned} a &= \max\{\alpha_i \sum_{j=1}^{n} \beta_j \|v_i - u_j\|_2 \mid i = 1, \ldots, m_g\} \\ b &= \max\{\beta_j \sum_{i=1}^{m} \alpha_i \|v_i - u_j\|_2 \mid j = 1, \ldots, m_f\} \\ D_2(F, G) &= \frac{1}{2}(a + b). \end{aligned} \tag{24}$$

In the following, the invoked GMM measures induced by D_1 will be denoted by GMM-Autoenc$_{1,AE}$, for the classical autoencoder architecture and GMM-Autoenc$_{1,AECNN}$ for the CNN autoencoder. Similarly, the GMM measures induced by D_2 will be denoted by GMM-Autoenc$_{2,AE}$ and GMM-Autoenc$_{2_AECNN}$, respectively. Both types satisfy self-similarity, positivity, and are symmetric; however, on the contrary to KL divergence, self-identity is not satisfied.

The second aggregation approach utilizes the EMD distance on the F and G representatives of two GMMs that are mutually compared. It is based on the work proposed in [24] and the EMD transportation problem. Namely, the computed tuples F and G are interpreted as the sets of low-dimensional Euclidean vectors equipped with the weights or masses $\alpha_i, i = 1 \ldots m_f$, and $\beta_j, j = 1 \ldots m_g$, respectively. Thus, we introduce the measure of similarity between two GMMs f and g to be the measure between weighted "clouds" F and G consisting of the Euclidean vectors in $\mathbb{R}^l$ (e.g., see [24]). Taking into account the distance or cost between all pairs of $\mathbb{R}^l$ vectors from F and G, i.e., by computing the matrix $[d_{ij}]$, it follows that the optimal flow $[\zeta_{ij}]$ (with the lowest overall cost) when moving $\mathbb{R}^l$ vectors from F to G is obtained as the solution to the following LP type minimization problem:

$$\begin{aligned}
&\min_{\zeta_{ij}} \sum_{i=1}^{m_f} \sum_{j=1}^{m_g} d_{ij}\zeta_{ij}, \\
&s.t. \\
&\zeta_{ij} \geq 0,\ i = 1, \ldots, m_f,\ j = 1, \ldots, m_g, \\
&\sum_{j=1}^{m_g} \zeta_{ij} \leq \alpha_i,\ i = 1, \ldots, m_f, \\
&\sum_{i=1}^{m_f} \zeta_{ij} \leq \beta_j,\ j = 1, \ldots, m_g, \\
&\sum_{i=1}^{m_f} \sum_{j=1}^{m_g} \zeta_{ij} = 1.
\end{aligned} \tag{25}$$

Based on the optimal flow $[\zeta_{ij}]$, it follows that the aggregate EMD distance between the sets of the Euclidean vectors in F and G can be computed as:

$$D_{EMD}(F, G) = \frac{\sum_{i=1}^{m_f} \sum_{j=1}^{m_g} d_{ij}\zeta_{ij}}{\sum_{i=1}^{m_f} \sum_{j=1}^{m_g} \zeta_{ij}}. \tag{26}$$

Regarding the LP in (25), $[d_{ij}]$ is the matrix of Euclidean distances between vectors v_i and u_j in $\mathbb{R}^l$, i.e., $d_{ij} = \|v_i - u_j\|_2$. The first constraint imposed on the solution $[\zeta_{ij}]$ defines the direction of mass or "supplies" flow, i.e., it allows moving supplies from the cloud or set F to the cloud or set G, but not vice versa. We also note that α_i as well as β_j sums up to be one. The second constraint in (25) limits the amount of supplies that can be sent from F to G, by imposing the restriction that the elements from F individually cannot supply more goods in total to the elements of G than the amount corresponding to the value of their weight in F; while the third constraint puts the same type of restrictions on the elements of G (that they cannot receive more supplies in total than the value of their individual weight, i.e., capacity). The fourth constraint in the form of equality serves the purpose to exploit all available flow from the cloud F to the cloud G (i.e., to move the maximum amount of supplies as possible). Once the transportation problem is solved, the optimal values of $[\zeta_{ij}]$ are determined, and the Earth mover's distance $D_{EMD}(F, G)$ in (26) is defined as the total work or cost (resulting value of the objective function) that is necessary in order to move, by optimal flow $[\zeta_{ij}]$, the maximum amount of supplies possible, from the "cloud" F to the "cloud" G, which is normalized by the total flow (as in the optimal transportation problem).

Moreover, the fact that the EMD distance is a metric (see [24]), implies that the measure of similarity between two GMMs defined by (26) is also a metric. We denote the GMM measures induced by D_{EMD} as GMM-Autoenc$_{3,AE}$ and GMM-Autoenc$_{3,AECNN}$, respectively.

2.3. Computational Complexity

In large-scale ML systems relying on GMMs as a way of modeling distributions of interest, the number of high-dimensional Gaussian components is usually large, which can induce a significant computational burden in cases when the similarity between the GMMs is measured. Therefore, it is of interest to have low-computational complexity, and at the same time, preserve the accuracy of performing the ML task, e.g., recognition, which can be achieved by the baseline KL divergence-based GMM similarity measures. Reducing the computational burden is of the utmost importance in providing efficient services based on such GMM-based ML systems.

Let us assume that the full covariance GMMs f and g have the same number of components m. We denote by d the dimension of the original feature space corresponding to some recognition task. The computational complexity of the Monte Carlo approximation

KL_{MC}, defined by relation (1), is estimated as $O(Nmd^3)$, where N is the number of samples. Namely, there are N observations for which two mixtures are calculated, each with m Gaussian components, and the computational complexity for calculating each component is of the order of $O(d^3)$, so the overall computational complexity is of the order of $O(Nmd^3)$. However, to obtain an efficient KL divergence approximation, the number of i.i.d. samples N must be large, making it inefficient. For the computational complexity of all KL-based measures, KL_{WA}, KL_{MB}, and KL_{VB}, defined in Section 2.1 by relations (3), (4) and (7), respectively, the rationale is as follows. Their complexities are dominated by the term $KL(f_i||g_j)$, and there are m^2 such terms, as it is assumed that both mixtures in the pair have m d-variate Gaussian components, so $i, j = 1, \ldots, m$. As $KL(f_i||g_j)$ is defined by (2), it can be seen that calculations of these are dominated by the inversion of d-dimensional SPD matrices, which is of the order of $O(d^3)$. Thus, the overall computation is of the order of $O(m^2d^3)$. We use all these in the experiments as the baseline measures.

Concerning the complexity of similarity measures reported in [27], they are estimated as $O(m\tilde{l}d^2) + O(md\tilde{l}^2) + O(m^2\tilde{l}^3)$ where $\tilde{l} \times \tilde{l}$ is the dimension of low-dimensional SPD representatives. We denote the mentioned similarity measures (unsupervised versions) by $uDPLM_{WA}$, $uDPLM_{MB}$, and $uDPLM_{VB}$ and use those in the experiments as baseline measures.

The measures reported in [25,26] have complexity estimated by $O(md^2l) + O(m^2l)$, where l is the dimensionality of the corresponding low-dimensional embedding. This also holds for GMM measures induced by D_1 and D_2, as given by (23) and (24), respectively, while D_{EMD}, given by (26), induces the GMM similarity measure with the complexity of the order of $O(md^2l) + O(m^2l) + O(m^5)$ (as can be seen, e.g., in [25,26]).

Since the proposed GMM-Autoenc$_{i,AE}$, $i = 1, 2$ has a single layer encoder, its computational complexity is influenced by the described aggregation operators, and can be estimated in the same way, i.e., as $O(md^2l) + O(m^2l)$, where l is the size of the low-dimensional autoencoder embedding (in the case of the architecture with the input layer, hidden layer, and output layer which is used in the experiments). On the other hand, for GMM-Autoenc$_{i,CNNAE}$, $i = 1, 2$, if we assume that there are two convolutional feature map layers in the encoder as well as the decoder of the embedding CNN autoencoder, in the case when using D_1 and D_2 given by (23) and (24), the computational complexity of the CNN-based GMM similarity measure can be estimated as $O(2mw_1h_1{k_1}^2n_1) + O(2mw_2h_2{k_2}^2n_1n_2) + O(2mw_3h_3{k_3}^2n_2n_3) + O(m^2l)$. Note that $n_i, w_i, h_i, k_i < d$ are the dimensions of the corresponding convolutional layers in the encoder (depth, width, height, and kernel spatial size).

For the proposed EMD-based similarity measure GMM-Autoenc$_{3,AE}$ induced by (26), the computational complexity is estimated as $O(md^2l) + O(m^2l) + O(m^5)$. Namely, the term $O(m^2l)$ corresponds to the computational complexity of calculating $\|v_i - u_j\|_{l2}$ between all pairs of l-dimensional representatives of Gaussian components f_i and g_j, respectively, $i, j = 1, \ldots, m$. The term $O(md^2l)$ comes from the encoding process in the autoencoder network. At the input, we have $d(d+1)/2$-dimensional vectors that are transformed by matrix multiplication and nonlinearity into l-dimensional output vectors (from the computational complexity perspective nonlinearity can be neglected if considered to be a piecewise affine). The third term, $O(m^5)$, comes from the computational complexity of the LP problem given by (25), where the number of iterations in the EMD computation is $O(2m)$, as described in [23]. Regarding the GMM-Autoenc$_{3,CNNAE}$, which is based on the CNN autoencoder architecture, the computational complexity can be estimated as $O(2mw_1h_1{k_1}^2n_1) + O(2mw_2h_2{k_2}^2n_1n_2) + O(2mw_3h_3{k_3}^2n_2n_3) + O(m^2l) + O(m^5)$, which has the similar complexity as the previously described Autoenc$_{i,CNNAE}$, $i = 1, 2$, with only one difference, namely that it has an additional term $O(m^5)$, originating from solving the LP problem in (25).

It can be seen that the computational complexity of the proposed GMM-Autoenc$_{i,AE}$ for $i = 1, 2, 3$, i.e., when using D_1 and D_2 and D_{EMD}, is similar to the corresponding GMM similarity measures reported in [25,26], which is thus computationally significantly more

efficient when compared with KL_{WA}, KL_{MB}, and KL_{VB}. Depending on the relation between $\tilde{l}$ and l, the proposed autoencoder-based measures are also potentially more efficient when compared with unsupervised $uDPLM_{WA}$, $uDPLM_{MB}$, $uDPLM_{VB}$, and the supervised $sDPLM_{WA}$, $sDPLM_{MB}$, and $sDPLM_{VB}$. The computational complexity of the proposed GMM-Autoenc$_{i,CNNAE}$, $i = 1, 2, 3$ is also significantly more efficient when compared with KL_{WA}, KL_{MB}, and KL_{VB}, but less efficient than GMM-Autoenc$_{i,AE}$, $i = 1, 2, 3$.

3. Experimental Results

In this section, the network architecture used in the experiments, as well as the experimental results of applications of the proposed novel GMM-Autoenc similarity measures are presented. The comparisons include several baseline measures given in Section 2.1, over various datasets within a texture recognition task.

3.1. Network Architectures

In the case of the fully connected autoencoder utilized in GMM-Autoenc$_{i,AE}$, $i = \{1, 2\}$, we used a simple fully connected architecture consisting of input and output layers with one hidden representation layer between them and the logistic sigmoid transfer function. The training consisting of scaled conjugate gradient descent optimization was driven by the MSE loss function defined in (13), accompanied by the l_2 weight regularization term $\mathcal{R}(\mathcal{F}, \mathcal{G})$ defined in (14), and the regularization penalty $\lambda = 10$. The number of nodes in the input and output layers was determined by the dimensionality of the input and output feature space. The CNN encoder utilized in GMM-Autoenc$_{i,CNNAE}$, $i = \{1, 2\}$ consists of three convolutional layers narrowing down the higher-dimensional input (matrices of size $d \times d$) into lower-dimensional vectors $\mathbb{R}^l$, while preserving the relevant clustering information. The depths of CNN layers are $n_1 = 32$, $n_2 = 64$, and $n_3 = 128$, with the square kernel sizes of $k_1 = 14$, $k_2 = 7$, and $k_3 = 3$. The dimensions of the layers' width w_i and height h_r, $r = \{1, 2, 3\}$ are implicitly determined by the size of input SPD matrices, but we note that the stride value was 2. The bottleneck layer of size $\mathbb{R}^l$ is connected to the last CNN layer by one flattening layer and two additional linear layers with a ReLU component in between. ReLU components are also added among convolutional layers, including one batch norm regularizer between the last two convolutional layers. The CNN decoder architecture replicates the above-described pattern in the backward order, decompressing the latent feature space into the original matrix features. For CNN autoencoder training, the Adam gradient descent optimization algorithm is applied. Adaptive optimization algorithms, such as Adam, have shown a better optimization performance in the means of faster convergence, due to the adaptive learning rate, as can be seen in [42]. Moreover, empirical results demonstrate that Adam works well in practice and still achieves a competitive performance when compared with newer methods, as can be seen in [43]. The procedure is performed by using the objective defined in (15) over 30 epochs, with a learning rate of 0.001, a learning decay rate decay of $1e-05$, and the feature map regularization term $\mathcal{R}_{\mathcal{FMR}}(\mathcal{F}, \mathcal{G})$ defined in (21), with a regularization penalty value of $\lambda_{FMR} = 10$. The first regularization term $\mathcal{R}(\mathcal{F}, \mathcal{G})$ in (15) refers to network weights and is of the same type as in the case of previously described GMM-Autoenc$_{i,AE}$. In both cases, all input matrices (from both the training and test data instances) have been normalized to the [0,1] range before the actual training.

3.2. Performances

Experimental results are given for the proposed GMM-Autoenc-based similarity measures invoked in Sections 2.2 and 2.2.4, compared to baseline *KL*-based measures KL_{WA}, KL_{MB} and KL_{VB} given by (3), (4), and (7), respectively, and proposed in [1,2]; as well as to the baseline EMD-based unsupervised $uDPLM_{WA}$, $uDPLM_{MB}$, $uDPLM_{VB}$ measures, proposed in [27]. Experiments were conducted within the texture recognition tasks on the UIUC texture dataset, the KTH-TIPS image dataset, and the UMD texture dataset. A five-class recognition problem is considered in all experiments. Concerning the UIUC

dataset, the following classes are considered: wood, water, granite, marble, and floor, with images of 640×480 pixel resolution. In the case of the KTH-TIPS dataset, the following classes were considered: aluminum foil, brown bread, corduroy, cotton, and cracker, with images of 1280×960 pixel resolution. Finally, for the UMD texture dataset, we used images from the following classes: paint cans, stones, brick walls, apples, and textile patterns, sampled at 1280×960 pixels. Selected samples from all three databases are shown in Figure 3.

Concerning the underlying texture features used for the experiments as texture descriptors, the region covariance descriptors proposed in [30] are employed, since they have already shown good performance in various texture recognition tasks (see for example [19,23,27,30]). For any given $N \times M$ image, they are formed in the following way: patches of size $m \times m$, $m < \min\{N, M\}$ are cropped ($m = 128$ with the step size of 16 for the UIUC dataset, $m = 40$ with the step size of 5 for the KTH-TIPS dataset and $m = 256$ with the step size of 32 for the UMD dataset). For every pixel positioned at (x, y), features were calculated as elements of $\mathbb{R}^{\hat{d}}$, $\hat{d} = 5$, i.e., $[I, |I_x|, |I_y|, |I_{xx}|, |I_{yy}|]$, where I represents illumination, I_x and I_y are the first-order derivatives and I_{xx} and I_{yy} are the second-order derivatives. Covariance matrices were calculated for the mentioned features and their upper triangular part is vectorized into $d = \hat{d}(\hat{d}+1)/2 = 15$-dimensional feature vectors. The parameters of GMMs are then estimated using the EM algorithm (as can be seen in [44]) applied over the pool of feature vectors obtained as previously described. During experiments, for every $n \times m$-size image in all mentioned datasets, the uniform sampling of the N_{smpl} sub-images of the size of $(n/2) \times (m/2)$ is performed in order to augment every database, which is needed to train a nonlinear autoencoder in order to perform the GMM-Autoenc measure (deep autoencoder in the case of GMM-Autoenc$_{1,AECNN}$), because originally, we do not have a sufficient amount of data in order to train them. Every GMM (or its low-dimensional Euclidean or SPD representatives for the proposed GMM-Autoenc or DPLM GMM similarity measures, respectively) is compared to all other GMMs in the training set, and its class is determined using the KNN method.

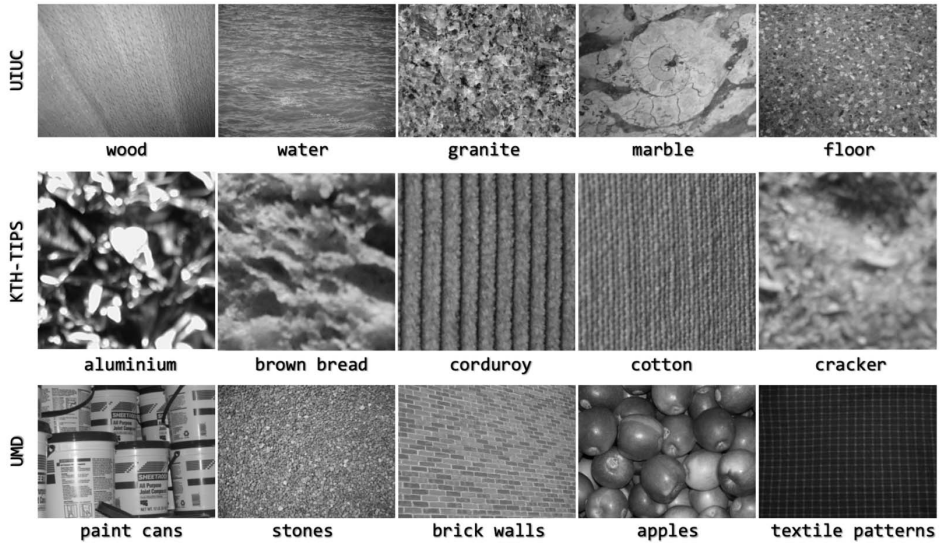

Figure 3. UIUC, KTH-TIPS, and UMD database samples.

In Tables 1–3, the performances of the novel GMM-Autoenc GMM similarity measures are presented when compared with *KL*-based as well as DPLM-based measures, for the UIUC, KTH-TIPS, and UMD datasets, respectively. In all experiments, the number of Gaussian components used to represent any particular training or testing image was set to $m \in \{1, 5, 10\}$. For the case of DPLM-based and GMM-Autoenc measures, $K = 3$ for the KNN, and the projection dimension for DPLM-based methods was set to $\tilde{l} \in \{5, 7\}$.

Results in Tables 1–3 for the proposed GMM-Autoenc$_{i,AECNN}$ refer to the model training procedure with FMR regularization based on the log-Euclidean ground distance $gd = le$ given in (19), as we obtained slightly better results with this regularizer when compared with other ground distances.

From the experimental results presented in Tables 1–3, which correspond to recognition accuracy, and taking into account the computational complexity analyses given in Section 2.3, it can be seen that all of the proposed novel GMM-Autoenc measures obtained a better trade-off between recognition accuracy and computational complexity for all datasets when compared with the baseline *KL*-based as well as DPLM-based measures mentioned in Section 2.2. We note that GMM-Autoenc$_{i,AE}$ variants could be the preferred choice when compared with GMM-Autoenc$_{i,AECNN}$ models, probably due to the number of training samples in all datasets, which is relatively small for the training of a CNN architecture.

Table 1. Recognition accuracies for the proposed GMM-Autoenc-based measures when compared with *KL*-based as well as DPLM-based GMM similarity measures on UIUC database.

GMM Sim. Meas.	**Accuracy**					
	$m = 1$		$m = 5$		$m = 10$	
KL_{MB}	0.82		0.80		0.80	
KL_{WA}	0.82		0.82		0.82	
KL_{VB}	0.82		0.82		0.82	
	$\tilde{l} = 5$	$\tilde{l} = 7$	$\tilde{l} = 5$	$\tilde{l} = 7$	$\tilde{l} = 5$	$\tilde{l} = 7$
$uDPLM_{MB}$	0.72	0.81	0.73	0.74	0.79	0.79
$uDPLM_{WA}$	0.72	0.81	0.73	0.74	0.80	0.80
$uDPLM_{VB}$	0.72	0.81	0.73	0.74	0.80	0.80
	$l = 20$	$l = 30$	$l = 20$	$l = 30$	$l = 20$	$l = 30$
GMM-Autoenc$_{1,AE}$	0.75	0.81	0.75	0.76	0.80	0.80
GMM-Autoenc$_{2,AE}$	0.75	0.81	0.76	0.76	0.80	0.80
GMM-Autoenc$_{3,AE}$	0.76	0.80	0.76	0.77	0.81	0.81
GMM-Autoenc$_{1,AECNN}$	0.75	0.80	0.73	0.77	0.81	0.80
GMM-Autoenc$_{2,AECNN}$	0.76	0.81	0.71	0.78	0.81	0.81
GMM-Autoenc$_{3,AECNN}$	0.77	0.81	0.71	0.79	0.81	0.80

Table 2. Recognition accuracies for the proposed GMM-Autoenc-based measures when compared with *KL*-based as well as DPLM-based GMM similarity measures on KTH-TIPS database.

GMM Sim. Meas.	**Accuracy**					
	$m = 1$		$m = 5$		$m = 10$	
KL_{MB}	0.78		0.74		0.75	
KL_{WA}	0.78		0.78		0.78	
KL_{VB}	0.78		0.78		0.78	
	$\tilde{l} = 5$	$\tilde{l} = 7$	$\tilde{l} = 5$	$\tilde{l} = 7$	$\tilde{l} = 5$	$\tilde{l} = 7$
$uDPLM_{MB}$	0.57	0.73	0.69	0.71	0.63	0.72
$uDPLM_{WA}$	0.57	0.73	0.72	0.75	0.64	0.75
$uDPLM_{VB}$	0.57	0.73	0.72	0.75	0.63	0.75
	$l = 20$	$l = 30$	$l = 20$	$l = 30$	$l = 20$	$l = 30$
GMM-Autoenc$_{1,AE}$	0.71	0.75	0.73	0.75	0.72	0.74
GMM-Autoenc$_{2,AE}$	0.71	0.74	0.72	0.74	0.72	0.75
GMM-Autoenc$_{3,AE}$	0.72	0.75	0.73	0.75	0.73	0.76
GMM-Autoenc$_{1,AECNN}$	0.72	0.75	0.73	0.74	0.73	0.73
GMM-Autoenc$_{2,AECNN}$	0.73	0.75	0.71	0.75	0.74	0.76
GMM-Autoenc$_{3,AECNN}$	0.73	0.76	0.72	0.77	0.74	0.77

Table 3. Recognition accuracies for the proposed GMM-Autoenc-based measures when compared with *KL*-based as well as DPLM-based GMM similarity measures on UMD database.

GMM Sim. Meas.	**Accuracy**					
	$m = 1$		$m = 5$		$m = 10$	
KL_{MB}	0.75		0.73		0.72	
KL_{WA}	0.75		0.75		0.75	
KL_{VB}	0.75		0.75		0.75	
	$\tilde{l} = 5$	$\tilde{l} = 7$	$\tilde{l} = 5$	$\tilde{l} = 7$	$\tilde{l} = 5$	$\tilde{l} = 7$
$uDPLM_{MB}$	0.73	0.74	0.72	0.72	0.70	0.72
$uDPLM_{WA}$	0.73	0.74	0.73	0.74	0.71	0.75
$uDLPM_{VB}$	0.73	0.74	0.73	0.74	0.71	0.75
	$l = 20$	$l = 30$	$l = 20$	$l = 30$	$l = 20$	$l = 30$
GMM-Autoenc$_{1,AE}$	0.74	0.74	0.73	0.73	0.71	0.72
GMM-Autoenc$_{2,AE}$	0.73	0.74	0.73	0.74	0.73	0.74
GMM-Autoenc$_{3,AE}$	0.74	0.75	0.74	0.75	0.73	0.75
GMM-Autoenc$_{1,AECNN}$	0.74	0.75	0.73	0.73	0.72	0.72
GMM-Autoenc$_{2,AECNN}$	0.75	0.75	0.74	0.75	0.74	0.74
GMM-Autoenc$_{3,AECNN}$	0.74	0.75	0.74	0.75	0.74	0.75

4. Conclusions

In this work, a novel similarity measure between GMMs based on autoencoder-generated Gaussian component representations is proposed. Low-dimensional Euclidean vectors obtained using nonlinear autoencoder embedding demonstrated higher similarity measurement capabilities when compared with the existing baseline models. Two different approaches were explored and utilized to encode the original Gaussian mixture components. The first one focused on the simple architecture of the classical autoencoder, where the vectorized versions of the SPD matrices obtained by the Lovric embedding of GMM Gaussians were used to generate inputs for the fully connected autoencoder. In the second approach, a more complex CNN autoencoder was utilized in combination with the novel regularization training in the feature space. In terms of trade-off between the recognition accuracy and the computational complexity, autoencoder embedding proved to be a better solution when compared with the existing GMM similarity measurement baselines in texture recognition tasks. In general, any domain in which some random processes, data instances, or groups of entities are described by PDFs in the form of Gaussian mixture models can be considered as possible fields of application that would benefit from the proposed computationally efficient GMM similarity measures. Besides ML recognition tasks based on decision rules in the form of minimum distance classifiers, we also foresee various applications in other domains such as clustering, probabilistic modeling, kriging methods, learning regularization, and feature space alignment as possible extensions of the current study and proposed similarity measures. For more effective dimensionality reduction, the utilized autoencoder networks should be trained on large datasets, but this could be performed in various settings, online or offline, or by designing encoders for specific sub-tasks and data modalities. In that sense, described methods are scalable and could be adapted to specific system requirements. Thus, in order to fully explore the capabilities of the CNN-based embedding approach, in future work, we plan to conduct experiments on tasks with much larger datasets and different data modalities.

Author Contributions: Conceptualization, V.K. and M.J.; methodology, V.K. and B.P.; software, B.P.; validation, B.B.; formal analysis, M.J. and N.R.; investigation, V.K., M.J. and B.P.; resources, B.P.; data curation, M.J.; writing—original draft preparation, M.J. and B.P.; writing—review and editing, B.B.; visualization, B.P. and B.B.; supervision, N.R.; project administration, B.P.; funding acquisition, B.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Serbian Ministry of Education, Science and Technological Development through project no. 45103-68/2020-14/200156: "Innovative Scientific and Artistic Research from the Faculty of Technical Sciences Activity Domain", and the internal project of the Faculty of Technical Sciences in 2023: "Research aimed at improving the teaching process and development of scientific and professional areas of the Department of Power, Electronic and Telecommunication engineering". This research also received publication fee support from the H2020 project INCOMING (ID 856967): "Innovation and excellence in massive-scale communications and information processing".

Data Availability Statement: Publicly available datasets were analyzed in this study. The UIUC data description can be found in [45], and data can be found here: http://www.cvr.ai.uiuc.edu/ponce_grp, accessed on 1 September 2016. KTH-TIPS data description can be found in [46], and data can be found here: https://www.csc.kth.se/cvap/databases/kth-tips, accessed on 1 October 2022. UMD data description can be found in [47], and data can be found here: http://users.umiacs.umd.edu/~fer/website-texture/texture.htm, accessed on 3 December 2022.

Acknowledgments: The authors would also like to acknowledge the ICONIC centre—Centre for Intelligent Communications, Networking and Information Processing (https://iconic.ftn.uns.ac.rs/); established through H2020 project INCOMING (ID 856967) at the Faculty of Technical Sciences in Novi Sad.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

AI	Artificial Intelligence
CNN	Convolutional Neural Network
DNN	Deep Neural Network
DPLM	Distance Preservation to the Local Mean
EM	Expectation Maximization
EMD	Earth Mover's Distance
FMR	Feature Map Regularized
GMM	Gaussian Mixture Model
KL	Kullback–Leibler
KNN	K-Nearest Neighbors
KTH	Royal Institute of Technology in Stockholm
LP	Linear Programming
MB	Matching-Based
MC	Monte Carlo
ML	Machine Learning
MLE	Maximum Likelihood Estimate
MSE	Mean Squared Error
PDF	Probability Density Function
SPD	Symmetric Positive Definite
TIPS	Textures Under Varying Illumination, Pose and Scale
UC	Unscented Transform
UIUC	University of Illinois Urbana-Champaign
UMD	University of Maryland
VB	Variational Bound
VAE	Variational AutoEncoder
WA	Weighted Average

References

1. Goldberger, J.; Aronowitz, H. A distance measure between GMMs based on the unscented transform and its application to speaker recognition. In Proceedings of the INTERSPEECH, Lisbon, Portugal, 4–8 September 2005; pp. 1985–1988.
2. Goldberger, J.; Gordon, S.; Greenspan, H. An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures. In Proceedings of the International Conference on Computer Vision, Nice, France, 13–16 October 2003; Volume 3, pp. 487–493.
3. Wu, Y.; Chan, K.L.; Huang, Y. Image texture classification based on finite Gaussian mixture models. In Proceedings of the 3rd Int. Workshop on Text. Anal. and Synth., Int. Conf. on Computer Vision, Nice, France, 13–16 October 2003; pp. 107–112.
4. Dilokthanakul, N.; Mediano, P.A.; Garnelo, M.; Lee, M.C.; Salimbeni, H.; Arulkumaran, K.; Shanahan, M. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *arXiv* **2017**, arXiv:1611.02648v2.
5. Gangodkar, D. A novel image retrieval technique based on semi supervised clustering. *Multimed. Tools Appl.* **2021**, *80*, 35741–35769. [CrossRef]
6. Asheri, H.; Hosseini, R.; Araabi, B.N. A new EM algorithm for flexibly tied GMMs with large number of components. *Pattern Recognit.* **2021**, *114*, 107836. [CrossRef]
7. Durrieu, J.L.; Thiran, J.P.; Kelly, F. Lower and upper bounds for approximation of the Kullback-Leibler divergence between Gaussian mixture models. In Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Kyoto, Japan, 25–30 March 2012; pp. 4833–4836.
8. Brkljač, B.; Janev, M.; Obradović, R.; Rapaić, D.; Ralević, N.; Crnojević, V. Sparse representation of precision matrices used in GMMs. *Appl. Intell.* **2014**, *41*, 956–973. [CrossRef]
9. Kaur, A.; Sachdeva, R.; Singh, A. Classification approaches for automatic speech recognition system. In *Artificial Intelligence and Speech Technology*; CRC Press: Boca Raton, FL, USA, 2021; pp. 1–7.
10. Demir, K.E.; Eskil, M.T. Improved microphone array design with statistical speaker verification. *Appl. Acoust.* **2021**, *175*, 107813. [CrossRef]
11. Yücesoy, E. Two-level classification in determining the age and gender group of a speaker. *Int. Arab J. Inf. Technol.* **2021**, *18*, 663–670. [CrossRef]
12. Narasimhan, H.; Vinayakumar, R.; Mohammad, N. Unsupervised deep learning approach for in-vehicle intrusion detection system. *IEEE Consum. Electron. Mag.* **2023**, *12*, 103–108. [CrossRef]
13. Chernoff, H. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *Ann. Math. Stat.* **1952**, *23*, 493–507. [CrossRef]
14. Bhattacharyya, A. On a measure of divergence between two statistical populations defined by their probability distribution. *Bull. Calcutta Math. Soc.* **1943**, *35*, 99–110.
15. Matusita, K. Decision rules, based on the distance, for problems of fit, two samples, and estimation. *Ann. Math. Stat.* **1955**, *26*, 631–640. [CrossRef]
16. Kullback, S.; Leibler, R.A. On Information and Sufficiency. *Ann. Math. Stat.* **1951**, *22*, 79–86. [CrossRef]
17. Kullback, S. *Information Theory and Statistics*; Dover Publications Inc.: Mineola, NY, USA, 1968.
18. Minh, H.Q.; Murino, V. Covariances in computer vision and machine learning. *Synth. Lect. Comput. Vis.* **2017**, *7*, 1–170. [CrossRef]
19. Hao, H.; Wang, Q.; Li, P.; Zhang, L. Evaluation of ground distances and features in EMD-based GMM matching for texture classification. *Pattern Recognit.* **2016**, *57*, 152–163. [CrossRef]
20. Nielsen, F. Chernoff information of exponential families. *arXiv* **2011**, arXiv:1102.2684.
21. Mak, H.W.L.; Han, R.; Yin, H.H. Application of variational autoEncoder (VAE) model and image processing approaches in game design. *Sensors* **2023**, *23*, 3457. [CrossRef]
22. Lucas, S.M.; Volz, V. Tile pattern KL-divergence for analysing and evolving game levels. In Proceedings of the Genetic and Evolutionary Computation Conference, Prague, Czech Republic, 13–17 July 2019; pp. 170–178.
23. Li, P.; Wang, Q.; Zhang, L. A novel earth mover's distance methodology for image matching with Gaussian mixture models. In Proceedings of the IEEE International Conference on Computer Vision, Sydney, Australia, 1–8 December 2013; pp. 1689–1696.
24. Rubner, Y.; Tomasi, C.; Guibas, L.J. The earth mover's distance as a metric for image retrieval. *Int. J. Comput. Vis.* **2000**, *40*, 99. [CrossRef]
25. Krstanović, L.; Ralević, N.M.; Zlokolica, V.; Obradović, R.; Mišković, D.; Janev, M.; Popović, B. GMMs similarity measure based on LPP-like projection of the parameter space. *Expert Syst. Appl.* **2016**, *66*, 136–148. [CrossRef]
26. Popović, B.; Cepova, L.; Cep, R.; Janev, M.; Krstanović, L. Measure of similarity between GMMs by embedding of the parameter space that preserves KL divergence. *Mathematics* **2021**, *9*, 957. [CrossRef]
27. Popović, B.; Janev, M.; Krstanović, L.; Simić, N.; Delić, V. Measure of similarity between GMMs based on geometry-aware dimensionality reduction. *Mathematics* **2022**, *11*, 175. [CrossRef]
28. He, X.; Niyogi, P. Locality preserving projections. In Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 9–14 December 2003; Volume 16.
29. He, X.; Cai, D.; Yan, S.; Zhang, H.J. Neighborhood preserving embedding. In Proceedings of the International Conference on Computer Vision, Beijing, China, 17–21 October 2005; Volume 2, pp. 1208–1213.
30. Sivalingam, R.; Boley, D.; Morellas, V.; Papanikolopoulos, N. Tensor sparse coding for region covariances. In Proceedings of the European Conference on Computer Vision, Heraklion, Greece, 5–11 September 2010; pp. 722–735.

31. Lovrić, M.; Min-Oo, M.; Ruh, E.A. Multivariate normal distributions parametrized as a Riemannian symmetric space. *J. Multivar. Anal.* **2000**, *74*, 36–48. [CrossRef]
32. Roy, S.S.; Hossain, S.I.; Akhand, M.; Murase, K. A robust system for noisy image classification combining denoising autoencoder and convolutional neural network. *Int. J. Adv. Comput. Sci. Appl.* **2018**, *9*, 224–235. [CrossRef]
33. Ahmed, A.S.; El-Behaidy, W.H.; Youssif, A.A. Medical image denoising system based on stacked convolutional autoencoder for enhancing 2-dimensional gel electrophoresis noise reduction. *Biomed. Signal Process. Control* **2021**, *69*, 102842. [CrossRef]
34. Munir, N.; Park, J.; Kim, H.J.; Song, S.J.; Kang, S.S. Performance enhancement of convolutional neural network for ultrasonic flaw classification by adopting autoencoder. *NDT E Int.* **2020**, *111*, 102218. [CrossRef]
35. Hershey, J.R.; Olsen, P.A. Approximating the Kullback Leibler divergence between Gaussian mixture models. In Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Honolulu, HI, USA, 15–20 April 2007; Volume 4, pp. 317–320.
36. Cover, T.M. *Elements of Information Theory*; Wiley Series in Telecommunications; John Wiley & Sons: New York, NY, USA, 1991.
37. Julier, S.J. *A General Method for Approximating Non-Linear Transformations of Probability Distributions*; Technical Report; Robotics Research Group, Department of Engineering Science, University of Oxford: Oxford, UK, 1996.
38. Ling, H.; Okada, K. An efficient earth mover's distance algorithm for robust histogram comparison. *IEEE Trans. Pattern Anal. Mach. Intell.* **2007**, *29*, 840–853. [CrossRef]
39. Davoudi, A.; Ghidary, S.S.; Sadatnejad, K. Dimensionality reduction based on distance preservation to local mean for symmetric positive definite matrices and its application in brain–computer interfaces. *J. Neural Eng.* **2017**, *14*, 036019. [CrossRef]
40. Arsigny, V.; Fillard, P.; Pennec, X.; Ayache, N. Fast and simple calculus on tensors in the log-Euclidean framework. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Palm Springs, CA, USA, 26–29 October 2005; pp. 115–122.
41. Klir, G.J.; Yuan, B. *Fuzzy Sets and Fuzzy Logic: Theory and Applicaions*; Prentice Hall New Jersey: Hoboken, NJ, USA, 1995; Volume 4.
42. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
43. Schmidt, R.M.; Schneider, F.; Hennig, P. Descending through a crowded valley-benchmarking deep learning optimizers. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual, 18–24 July 2021; pp. 9367–9376.
44. Webb, A.R. *Statistical Pattern Recognition*; John Wiley & Sons: Hoboken, NJ, USA, 2003.
45. Lazebnik, S.; Schmid, C.; Ponce, J. A sparse texture representation using local affine regions. *IEEE Trans. Pattern Anal. Mach. Intell.* **2005**, *27*, 1265–1278. [CrossRef] [PubMed]
46. Fritz, M.; Hayman, E.; Caputo, B.; Eklundh, J.O. *The Kth-Tips Database*; Technical Report; Computational Vision and Active Perception Laboratory, Department of Numerical Analysis and Computer Science: Stockholm, Sweden, 2004. Available online: https://www.csc.kth.se/cvap/databases/kth-tips/doc/ (accessed on 1 October 2022).
47. Xu, Y.; Ji, H.; Fermüller, C. Viewpoint invariant texture description using fractal analysis. *Int. J. Comput. Vis.* **2009**, *83*, 85–100. [CrossRef]