Evolving Gaussian Mixture Models for Classification

Simon Reichhuber[a] and Sven Tomforde[b]
Intelligente Systeme, Christian-Albrechts-Universität zu Kiel, Kiel, Germany

Keywords: Gaussian Mixture Models, Evolutionary Algorithms, Classification, Human Activity Recognition.

Abstract: The combination of Gaussian Mixture Models and the Expectation Maximisation algorithm is a powerful tool for clustering tasks. Although there are extensions for the classification task, the success of the approaches is limited, in part because of instabilities in the initialisation method, as it requires a large number of statistical tests. To circumvent this, we propose an 'evolutionary Gaussian Mixture Model' for classification, where a statistical sample of models evolves to a stable solution. Experiments in the domain of Human Activity Recognition are conducted to demonstrate the sensibility of the proposed technique and compare the performance to SVM-based or LSTM-based approaches.

1 INTRODUCTION

In intelligent technical systems, e.g. referring to concepts from the Autonomic (Kephart and Chess, 2003) or Organic Computing (Müller-Schloer and Tomforde, 2017) domains, decisions about appropriate behaviour are typically taken based on a model of the current perceptions. In many cases, this requires continuous processing and classification of multi-dimensional sensor data. A large variety of techniques can be found in literature that is applicable to this task (D'Angelo et al., 2019) – but only a few techniques can provide an estimate of the associated uncertainty in addition to the classification decision. Classifiers based on Gaussian Mixture Models (GMM) (Heck and Chou, 1994) have been shown to come with a set of advantages including an inherent estimate if uncertainty and a probability distribution for the corresponding classes.

Consider human activity recognition (HAR) (Kong and Fu, 2018) as an example for a classification task in intelligent systems. Here, the behaviour of a human user is perceived by sensors, e.g. in terms of gyroscope, accelerometer or magnetometer available on a smartphone. The incoming data stream of the different sensors is pre-processed online, possibly segmented into smaller parts, and classified. The classification can consider basic activities such as sitting, walking, or running – but also more sophisticated activities such as cycling or playing football. Technically, these known activities are represented by expected patterns and compared to the currently observed patterns.

A standard approach for applying GMMs (or better CMM: classifiers based on GMM) to HAR is to run the Expectation Maximisation algorithm on large sets of training data. In this paper, we propose an alternative approach: We aim at evolving the CMM/GMM using evolutionary operators. Hence, the contribution of the paper is twofold: It defines the underlying evolutionary including the operators and presents a detailed study on GMM-based HAR data sets.

The remainder of this paper is organised as follows: Section 2 briefly reviews the related literature of GMMs, classification, and Evolutionary Algorithms. Next, we give a short recap of the used algorithms (Section 3) which also leads to the used reference baseline approach (Section 3.4). Then, in Section 4, we present our methodology of evolving GMMs for classification. Subsequently, we explain the experimental setup and present its results in Section 5. Finally, we give a short outlook to our further research (Section 6) and summarise our findings in Section 7.

2 RELATED WORK

Gaussian Mixture Models (GMMs) has been frequently used in the literature for the sake of a small-dimensional representation. For example, in the case of GMM-based adaptive knot placement for B-

[a] https://orcid.org/0000-0001-8951-8962
[b] https://orcid.org/0000-0002-5825-8915

Reichhuber, S. and Tomforde, S.
Evolving Gaussian Mixture Models for Classification.
DOI: 10.5220/0010984900003116
In *Proceedings of the 14th International Conference on Agents and Artificial Intelligence (ICAART 2022) - Volume 3*, pages 964-974
ISBN: 978-989-758-547-0; ISSN: 2184-433X

Splines (Zhao et al., 2011), projection pursuit with GMMs (Scrucca and Serafini, 2019), or as features for Support Vector Machines (in this context, the representation is also called "Universal Background Model" (You et al., 2010a)). Another field that uses the benefits of GMMs is the Organic Computing (OC) domain. The robustness against sensor noise and the advantage to analyse the knowledge represented by a GMM are two important characteristics making it attractive for constructors of such self-adapative and self-aware systems (Jänicke et al., 2016; Müller-Schloer and Tomforde, 2017). Some OC properties are guaranteed (with OC properties being a superset of self-* properties as originally defined for Autonomic Computing in (Kephart and Chess, 2003)), for instance adding new sensors to observation space (self-improving) and replacing defect ones (self-healing). In decision-making problems, GMMs are used for the detection of optimal actions which was formulated as a classification problem (Müller-Schloer and Tomforde, 2017). These GMM-based classifiers are denoted as CMMs and have been used in several applications throughout the last decades, such as speaker identification (del Alamo et al., 1996; Ozerov et al., 2011; Qi and Wang, 2011), language recognition (You et al., 2010b), Arabic sign language recognition (Deriche et al., 2019), face video verification (Li and Narayanan, 2011), limb motion classification using continuous myoelectric signals (Huang et al., 2005), or classification of sequences observed in electrocardiogram streams (Martis et al., 2009). Due to the inherent robustness against noise and explicit modelling of the underlying uncertainties in classification tasks, GMMs have been shown to be a powerful tool for human activity recognition (HAR) (Jänicke et al., 2016). In typical examples, the principles of using accelerometer data from smartphones (with a sampling rate of about 20 Hz to 40 Hz) for motion detection was shown in (Lau and David, 2010) and recently has been refined in (Garcia-Gonzalez et al., 2020). Thus, we rely on HAR as a use case in this article.

3 FOUNDATIONS OF GAUSSIAN MIXTURE MODELS AND EVOLUTIONARY ALGORITHMS

This section briefly summarises the technical basis of our contribution. It, therefore, introduces the concept of GMMs and explains how they can be turned into classifiers.

3.1 GMM

A Gaussian Mixture Model (GMM) is the superposition of multiple (here J) Gaussians called *components* (cf. Eq. (1)). Each component j has its own set of parameters: the mean vector $\boldsymbol{\mu}_j$ which describes its location and a covariance matrix $\boldsymbol{\Sigma}_j$ that describes its shape. To ensure that the GMM represents a valid density (i.e., $\int P(x)\mathrm{d}x = 1$) so-called *mixture coefficients* π_j are introduced for each component (cf. Eq. (2) and Eq. (3) for given constraints).

$$P(\boldsymbol{x}) = \sum_{j=1}^{J} \pi_j \cdot \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) \tag{1}$$

$$0 \leq \pi_j \leq 1 \tag{2}$$

$$1 = \sum_{j=1}^{J} \pi_j \tag{3}$$

$$\boldsymbol{\gamma}_{\boldsymbol{x}',j} = \frac{\pi_j \mathcal{N}(\boldsymbol{x}'|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{P(\boldsymbol{x}')} \tag{4}$$

The probabilities $\boldsymbol{\gamma}_{\boldsymbol{x}',j}$ (cf. Eq. (4)) are called *responsibilities* and indicate to what degree a given sample $\boldsymbol{x}'$ belongs to a component j.

The parameters for a GMM can be fitted using standard Expectation-Maximisation (EM), or variational inference (Bishop, 2006) on a suitable training set $\boldsymbol{X}_{train}$. An implementation for variational Bayesian inference for GMM can be found, e.g., in (Gruhl et al., 2021). Belonging to the class of generative probabilistic models, a GMM can be used to *sample* from it. That is, to generate *distinct* data sets that have the same statistical properties as the used training data. The sampling process is twofold. The first step is to draw $\hat{j}$ from a *categorical distribution* (also called *multinoulli*) described by the mixture coefficients π_j (cf. Eq. (5)). In the second step, $\hat{j}$ indicates from which of the Gaussian components a sample is drawn (cf. Eq. (6)). By repeating this procedure we can sample a whole data set $\boldsymbol{X}^*$ with the same statistical properties as the training set $\boldsymbol{X}_{train}$.

$$\hat{j} \simeq \mathrm{Cat}(\pi_1, \ldots, \pi_J) \tag{5}$$

$$\boldsymbol{x}^* \simeq \mathcal{N}(\boldsymbol{\mu}_{\hat{j}}, \boldsymbol{\Sigma}_{\hat{j}}) \tag{6}$$

3.2 CMM

Although GMMs are popular for clustering (McLachlan and Rathnayake, 2014; Chen et al., 2020), they have also been extended for classification. For example, in (Müller-Schloer and Tomforde, 2017), classifier based on Gaussian Mixture Models are denoted as CMMs.

Similar as in (Müller-Schloer and Tomforde, 2017), we define the classification model with GMMs, starting with the class conclusions $p(c|j)$, for each component j:

$$p(c|j) = \xi_{j,c} = \frac{1}{N_j} \sum_{x \in \mathcal{X}_c} \gamma_{\boldsymbol{x},j}, \tag{7}$$

Where $\mathcal{X}_c$ are the samples in $\mathcal{X}$ assigned with label c and $N_j = \sum_{i=1}^{N} \gamma_{\boldsymbol{x}_i,j}$ is the sum of the responsibilities of component j.

Then, the class posterior is given by:

$$p(c|\boldsymbol{x}) = \sum_{j=1}^{J} p(c|j) \cdot p(j|\boldsymbol{x}) = \sum_{j=1}^{J} \xi_{j,c} \cdot \gamma_{\boldsymbol{x},j}, \tag{8}$$

Finally, the discrimination function is constructed by calculating the a-posterior of the class probabilities:

$$h(\boldsymbol{x}') = \arg\max_{c} \left\{ p(c|\boldsymbol{x}' \right\} \tag{9}$$

The final classification procedure using CMMs consist of four steps:

(1) *Initialisation*

(2) *K-Means refinement*

(3) *EM-Updates*

(4) *Evaluation*

The first step, (1) *Initialisation*, shall provide initial values for the means $\boldsymbol{\mu}_j$ and the covariance matrix $\boldsymbol{\Sigma}_j$ of each component j in the CMM. This can either be randomly done (cf. Reference baseline in Section 3.4) or with an advanced procedure (cf. Evolved CMMS in Section 4). To guarantee that $\boldsymbol{\Sigma}_j$ is positive definite, we can use the steps in Eq. (10), Eq.(11), and Eq.(12) to generate random positive definite matrices:

$$M := (m_{i,j})_{1 \leq i,j \leq n} \qquad \text{where } m_{i,j} \sim U_{[-1,1]} \tag{10}$$

$$M \leftarrow 0.5 \cdot (M + M^T) \tag{11}$$

$$M \leftarrow M + n \cdot 1\!\!1_n \tag{12}$$

Subsequently, it is reasonable to stabilise the means and distribute them according to the density of the input space by applying a K-Means procedure resulting in step (2) *K-Means refinement*. Since K-Means does not refer to labels, some more complex classes are modelled by more components than other classes. Afterwards, we apply EM-updates in step (3) *EM-Updates* until a predefined stop criterion is reached. The latter can either be a total number of iterations, or a certain threshold denoting the change in the likelihood of drawing X from the current CMM to the one of the last EM-Update $\Delta P(X|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$. In the end, we evaluate in step (4) *Evaluation* the differences between the CMM model predictions of an unseen test set $\hat{y}_{test}$ and the provided labels of this test set y_{test}. Finally, the measured classification error is defined as our classification performance.

3.3 Evolutionary Algorithms

Evolutionary Algorithms (EAs) are a powerful tool for multi-dimensional minimisation/maximisation problems. Given an input space X and a multi-dimensional function $f : X^d -> \mathbb{R}$ the goal is to find an input x^* maximising the function f, i.e. $x^* = \text{argmax}_{x \in X} f(x)$. One of the pioneers of the idea of evolving some randomly found solutions by means of evolutionary operators (i.e. *selection*, *recombination*, and *mutation*) was John H. Holland who defined one of the first Genetic Algorithms in the year 1975(Holland, 1975). Given a set of $N_{\mathcal{P}}$ uniformly-distributed individuals, denoted as population $\mathcal{P}$, as seen in Algorithm 1 in each iteration the individuals' fitness is calculated (*CalculateFitness()*), based on the fitness value, the most powerful individuals are selected (*Selection()*) for a recombination procedure (*Recombination()*). In the end, random mutation is applied to all of the novel individuals *Mutation()* which form the next generation. Hence, these iterations are also called generations g. The algorithm terminates either after a previously-defined maximum number of generations $g < G$, or the best-found fitness value is above a certain threshold $\tau \in \mathbb{R}$.

Algorithm 1: The canonical GA algorithm.

```
1:  function GA(X, f)
2:      P^(0) ← Initialisation()
3:      g ← 0
4:      while bestFitness(P^(g)) < τ AND g < G do
5:          Fitness ← CalculateFitness(f, P^(g))
6:          Parents ← Selection(P^(g))
7:          Offspring ← Recombination(Parents)
8:          P^(g+1) ← Mutation(Offspring)
9:          g ← g + 1
10:     end while
11:     return x*, f(x*), g
12: end function
```

3.4 Reference Baseline

As a reference baseline, we initialise a population of CMMs by means of uniform-randomly distributed configurations of the parameters: responsibilities $\gamma_{\boldsymbol{x}',j}$, means $\boldsymbol{\mu}_j$, and covariance matrices $\boldsymbol{\Sigma}_j$ The common way to find the most suitable CMM representation with $f(X|\Omega) \sim Y$ is to uniformly initiate the model parameters multiple times, and find the centres

of classes via K-Means iterations. Afterwards, the EM-Algorithm is applied and the best initialisation in terms of classification error is kept. Given the number of generations G and the population size $N_{\mathcal{P}}$, the evolution of CMMs requires $GN_{\mathcal{P}}$ evaluations. With the same number resulting in similar computing time, one can compare the evolutionary setting with the random initialisation and subsequent K-Means.

4 METHODOLOGY

4.1 Evolution of CMMs

By the term "Evolved Machine Learning" we mean the maximisation of the performance of a machine learning model through evolutionary algorithms. In this process, a population is created that consists of machine learning models with randomly selected hyperparameter configurations.

$$\max_{\theta \in \Theta} \pi(\theta) \tag{13}$$

Encoding

In the case of CMMs we define individual i of the population as:

$$\theta^{[i]} = \left(\boldsymbol{\pi}^{[i]}, \boldsymbol{\mu}^{[i]}, \boldsymbol{\Sigma}^{[i]}\right) \tag{14}$$

Fitness Function

Furthermore, the fitness function is given by the classification performance that we are interested in. That means, for each fitness call we have to follow the whole classification procedure as we defined in Section 3.2. Based on this fitness value, we are able to apply Evolutionary Algorithms and filter out the fittest individual after a maximum number of N_G generations has been evolved (cf. Figure 1. Since we are dealing with highly unbalanced datasets with $|\{y \in \mathcal{Y} | y = c_i\}| << |\{y \in \mathcal{Y} | y = c_j\}|$ for some classes $c_i, c_j \in \mathcal{C}$, we decided to mostly concern the balanced accuracy in our experiments. Given a set of true labels $\mathcal{Y} := \{y\}_{i=1}^N$ and model prediction $\hat{\mathcal{Y}} = \{f(x_i)\}_{i=1}^N = \{\hat{y}_i\}_{i=1}^N$, the balanced accuracy weights each accuracy is defined as in (Velez et al., 2007) (cf. Eq. 15).

$$\texttt{balanced-accuracy}(\mathcal{Y}, \hat{\mathcal{Y}}) = \sum_{i=1}^{N} \frac{\chi_{\{y_i\}}(\hat{y}_i)}{|\{y' \in \mathcal{Y} | y' = y_i\}|}, \tag{15}$$

$$\text{where } \chi_S(\omega) = \begin{cases} 1 & \text{if } \omega \in S \\ 0 & \text{if } \omega \notin S \end{cases} \tag{16}$$

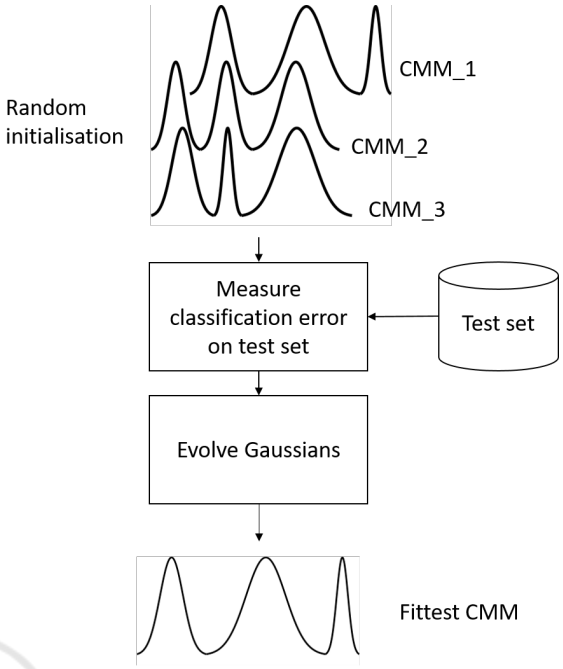


Figure 1: Schema of the procedure for evolving CMMs.

Selection

The parent population is found with respect to the fitness values of the individuals. The naïve selection strategy that takes the top k individuals into account leads to a poor diversity of the next generations. The latter is solved by stochastic selection strategies. For our approach, we used the selection strategy *remainder stochastic sampling* (Holland, 1975; Goldberg et al., 1990; Blanco et al., 2001) based on the relative fitness. *Remainder stochastic sampling* combines the strength of stochastic selection were even the weakest individual has a chance to be selected, similarly as stochastic universal sampling, with a guaranteed selection of individuals with a relative fitness above the average relative fitness which limits the diversity but stabilises the best solutions found so far.

Recombination

From the selection, there will be drawn two parents $\boldsymbol{x}^{p_1}$ and $\boldsymbol{x}^{p_2}$ for pairing which results in two novel children $\boldsymbol{x}^{c_1}$ and $\boldsymbol{x}^{c_2}$. Since GMMs are probabilistic generative models, the simplest way to generate a new child from two GMM parents would be to linearly combine both parents and draw a new GMM from the linear combination, which is also a GMM. Here In order to keep track of the class imbalance also during recombination. The ratio of the components per class should remain. Therefore, we first identify the

components of a parent that is most likely generating class c. Given a class $c \in \mathcal{C}$, the class components of parent p are defined as:

$$\iota^{p_1}(c) := \{j \in \mathbb{N}(J) | \arg\max_{j' \in \mathbb{N}(J)} p(c_i|j') = j\} \quad (17)$$

Using the class components in Eq. 17, Parent 1 induces her knowledge about the class complexity of class c which can be estimated from the magnitude of the class components $|\iota^{p_1}(c)|$ to child 1, and parent 2 analogous to child 2. This number is used to determine the number of draws from the combined component indices of parent 1 and parent 2, i.e. $\iota^{p_1}(c) \cup \iota^{p_2}(c)$, representing the knowledge about class c. Since $\sum_c |\iota^{p_1}(c)| = J$, an iteration over all classes sufficiently defines two new J-component CMM children.

Mutation

Since each of the three CMM parameters $\boldsymbol{\pi}^{[i]}$, $\boldsymbol{\mu}^{[i]}$, and $\boldsymbol{\Sigma}^{[i]}$ have different effects on the CMM, the mutation should also be divided into three types. Hence, there is the need for three mutation probabilities: (1) a mixture-coefficient mutation probability $p_{\boldsymbol{\pi}}$, (2) a mean-mutation probability $p_{\boldsymbol{\mu}}$, and (3) a correlation-matrix mutation probability $p_{\boldsymbol{\Sigma}}$. For the mean-mutation, we simply implemented a uniformly drawn one-step mutation. Because only a single dimension in a mean vector of the component of a CMM is altered, the probability $p_{\boldsymbol{\mu}}$ has to be adequately small. The correlation-matrix mutation is applied by replacing a randomly selected row or column with index i of the correlation matrix with a randomly drawn row or column with values in $[-1,1]$ where to the i-th entry is added the number J. The addition keeps the positive definiteness. Once an individual was chosen for mixture coefficient mutation, we apply a pairwise mixture coefficient exchange of the components i and j of an individual as in Eq. 18

$$s \leftarrow U[-s,s] \quad \text{, where } s := \frac{1}{J} \quad (18)$$

$$\pi_i \leftarrow s \quad (19)$$

$$\pi_j \leftarrow -s \quad (20)$$

Elitism

Only, a small amount of the best individuals in the population, denoted as elites, are immune to the mutation operation and are added directly to the next generation. Therefore, the parameter r_{elite} defines the fraction of elites in the population.

Algorithm 2: Evolved-CMMs as a combination of an EA and CMMs. The negative loss function $-\mathcal{L}$ is set to the balanced accuracy.

function EVOLVED-CMMS($\boldsymbol{X}_{train}$, $\boldsymbol{y}_{train}$, $\boldsymbol{X}_{test}$, $\boldsymbol{y}_{test}$)
  $\boldsymbol{X}_{train} \leftarrow$ Z-norm$(\boldsymbol{X}_{train}, \mu_{train}, \sigma_{train})$
  $\boldsymbol{X}_{test} \leftarrow$ Z-norm$(\boldsymbol{X}_{test}, \mu_{train}, \sigma_{train})$
  Initialise population of CMMs $P^{(0)}$
  while $g < G$ **do**
    for each $i \in \mathbb{N}(N_P)$ **do**
      CMM$_i$[fitness] $\leftarrow -\mathcal{L}(\boldsymbol{y}_{test}, \text{CMM}_i(\boldsymbol{X}_{test}))$
    end for
    Parents $\leftarrow$ RemainderStochasticalSampling $(\{CMM_i\}_{i=1}^{N_P})$
    Offspring $\leftarrow$ Recombination(Parents)
    NextGeneration $\leftarrow$ Mutation(Parents $\cup$ Offspring) $\cup$ Elite
    $g \leftarrow g+1$
    $P^{(g)} \leftarrow$ NextGeneration
  end while
  CMM* $\leftarrow$ $\arg\max_{CMM \in CMM^{(g)}} -\mathcal{L}(\boldsymbol{y}_{test}, \text{CMM}_i(\boldsymbol{X}_{test}))$
  score $\leftarrow -\mathcal{L}(\boldsymbol{y}_{test}, \text{CMM}_i^*(\boldsymbol{X}_{test}))$
  return CMM*, score
end function

4.2 Evolving Time-consuming Machine Learning Models

One question is how to reduce the huge amount of floating-point operations (FLOPs). Besides the individuals that are marked as elitists, there is also a percentage of selected parents that are included untouched in the next generation. Since their model parameters already have been evaluated once, it makes sense to keep track of unchanged individuals and prevent a time-consuming re-computation of the fitness. Another runtime optimisation can be found in parallelisation of the individuals of one generation within the GPU as explained in (Chen et al., 2020).

5 EXPERIMENTS

In this section, we explain the conducted experiments in the domain of HAR. The dataset was taken from the contribution of Garcia-Gonzalez et. al (Garcia-Gonzalez et al., 2020) and contains sensor records from 19 smartphone-users who wear their smart phone in daily-life and annotated their state of activity by giving labels, i.e. *inactive*, *active*, *walking*,

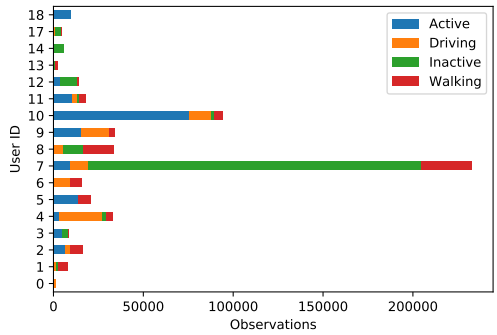


Figure 2: Total number of samples provided for each activity of each user in the dataset HAR_1.

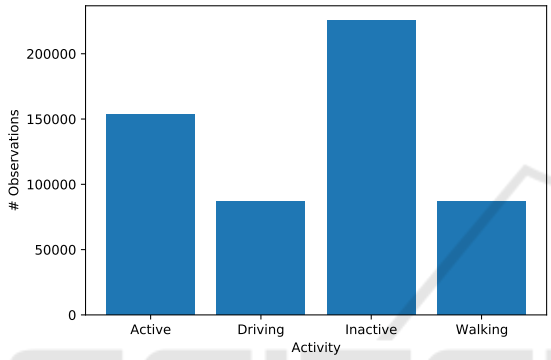


Figure 3: Total number of samples provided for each of the classes in the dataset HAR_1.

and *driving*. They sampled the streams with a sliding window approach into 20 sec-chunks with an overlap of 19 sec. In Figure 2, the distribution of the samples per user is visualised.

Another characteristic of this dataset is the class imbalance, which can be seen in Figure 3. These imbalance should be taken into account by the classification and underlines the necessity of the balanced accuracy, i.e. a naïve classifier always predicting the activity *inactive* would reach a accuracy of 43 %.

For the analysis of the optimal sensors, the authors provided three different sets based on different sensors, which we abbreviate HAR_1, HAR_2, and HAR_3 in the remainder. The detailed sensor equipment of each dataset can be seen in Table 1. The differencies in the number of samples between HAR_1, HAR_2, and HAR_3 is due to different user equipment. The claimed objective of the authors of (Garcia-Gonzalez et al., 2020) is to provide a dataset from which machine learning algorithms can create a orientation-independent, placement-independent, and subject-independent predictor for the activity. Even if further experiments are needed to affirm the claim for more than the included 19 subjects, more placements, and more orientations (which were not been defined,

Table 1: Sensor equipment, support, and involved users of each of the three datasets HAR_1, HAR_2, and HAR_3 provided by (Garcia-Gonzalez et al., 2020).

Name	Used sensors	#samples	User IDs
HAR_1	Accelerometer GPS	182387	All users (ID 0 to ID 18)
HAR_2	Accelerometer Magnetometer GPS	177246	Missing IDs: 14, 15, 16, 18
HAR_3	Accelerometer Gyroscope Magnetometer GPS	164762	Missing Ids: 4, 14-18

hence, determined by the individual behaviour of the users), the large variety of these attributes induces a high noise in the dataset. In contrast to this noise which is not well understood, for the reduction of the pure sensor noise, there are simple methods available in machine learning. For this reason, the author provide a preprocessing with low-pass filter and suggest to extract the following statistical features for each sensor:

- Mean
- Minimum
- Maximum
- Variance
- Mean absolute deviation
- Interquartile range

Counting the number of extracted values per sensor per timestamp, there are three for the accelerometer, three for the gyroscope, and three for the magnetometer. Additionally, for the GPS, there can be extracted six values: latitude, longitude, altitude, speed, bearing, and accuracy.

Given three values measured by the accelerometer, six values extracted from the GPS, and for each six statistical features, it sums up to $6 \cdot (3+6) = 54$ features for the HAR_1 dataset, analogous for HAR_2 $6 \cdot (3+3+6) = 72$ features, and for HAR_3 $6 \cdot (3+3+3+6) = 90$ features. Using these features, the fitness function requires a training phase of a CMM on 67 % of the dataset with the following parameters in Table 2 and a test phase calculating the balanced accuracy on 33 % of the dataset. Before the training or testing, the samples were z-normalised based on variance and mean found in the training set.

With this CMM setting, we initialised the evolutionary algorithm with the parameters in Table 3. Because of the limtited computing time, the parameters were manually optimised. Further improvements in the efficiency of the algorithm may allow a comprehensive hyperparameter tuning.

Table 2: Parameters used for training CMMs without evolutionary adaptation.

Parameter	Description	Value
J	Number of Gaussian components	30
`covariance_type`	Characteristics of the positive definite covariance matrix (e.g. *full*, *diagonal*, *constant*)	*full*
`em_updates_max`	Maximum number of EM updates	100
`em_tolerance`	Stop criterion for EM updates if the change of the likelihood $\Delta P(X\|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$ is below this threshold	1×10^{-3}

Table 3: Parameters used for the evolutionary algorithm.

Parameter	Description	Value
$N_{\mathcal{P}}$	Population size	100
N_G	Number of generations	100
r_p	Parents ratio	50 %
r_e	Elites ratio	1 %
$p_{\boldsymbol{\pi}}$	Mutation probability for mixture-coefficient	1 %
$p_{\boldsymbol{\mu}}$	Mutation probability for means	1 %
$p_{\boldsymbol{\Sigma}}$	Mutation probability for covariance matrix	1 %

Time Complexity versus Performance Trade-off

The setting of the parameters in Table 2 comes with a trade-off decision between accuracy (cf. Figure 4) and computing time (cf. Figure 5). Therefore, we analysed the balanced accuracy of models of varying complexity. The higher the number of components in a single CMM, the larger is its model complexity. As a drawback, this model has to be called for each fitness call.

Therefore, a good trade-off between the performance reaching a level of 81.108% balanced accuracy and limiting the computing time to 2347 sec (for comparison, using 40 components the computing time increases to 3209 sec) are 30 components. It is noteworthy that also the number of features influences the computing time and there is a break point between 54 features of HAR_1 and 72 features of HAR_2, whereas the balanced accuracy is quit high for all datasets.

In comparison to the authors of the dataset who applied support vector machines for the classification (Garcia-Gonzalez et al., 2020) with a mean accuracy of HAR_1 67.53 %, HAR_2 74.39 %, and HAR_3 69.28 %, the evolved CMMs predict up to a balanced accuracy of 80.03 % (cf. Figure 4). The baseline (see Section 4) eventually reached a maximum balanced accuracy of 79.324 %. But on average it only got 75.916 % with a standard deviation of 1.464 %. With this, the baseline is in the near of the evolutionary algorithm but the chances of having an appropriate CMM initialisation are in general not sufficient for a secure guarantee. In contrast to the random initialisation, because of elitism in the evolving CMMs, the fitness value history is monotonically increasing. Therefore, in all of the runs, the evolved CMMs reached a stable solution above 80 % balanced accuracy as seen in Figure 6. We also assume that the larger sensor sets HAR_2 and HAR_3 would increase further with more generations and more computing time. With respect to the computing time and the best results, we would decide to use dataset HAR_1 for HAR classification with Gaussians and statistical features.

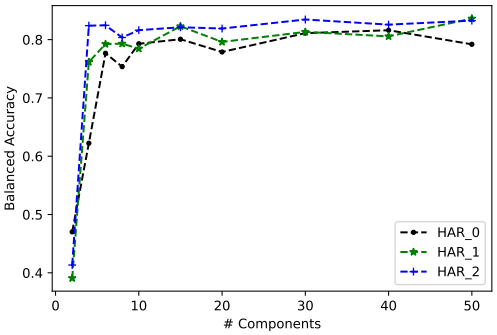


Figure 4: Balanced accuracy of evolved GCC trained on each of the data sets, i.e. `HAR_0`, `HAR_1`, `HAR_2`.

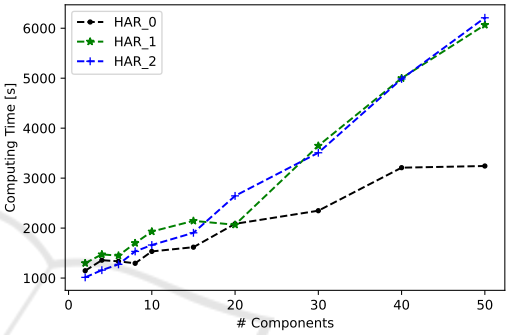


Figure 5: Computing time of evolved GCC trained on each of the data sets, i.e. `HAR_0`, `HAR_1`, `HAR_2`.

Table 4: Classification results of evolved CMMs tested on 33 % of the samples in the datasets HAR_1, HAR_2, and HAR_3.

	precision	recall	f1-score	support
Active	0.85	0.58	0.69	50741
Driving	0.87	0.83	0.85	28768
Inactive	0.79	0.93	0.86	74398
Walking	0.66	0.76	0.70	28480
accuracy	0.79	0.79	0.79	1.0
macro avg	0.79	0.78	0.78	182387
weighted avg	0.80	0.79	0.79	182387

(a) Classification results of evolved CMMs tested on 182k samples of HAR dataset with features based on triaxial accelerometer sensor and GPS.

	precision	recall	f1-score	support
Active	0.79	0.62	0.70	47107
Driving	0.73	0.92	0.81	28613
Inactive	0.80	0.95	0.87	72676
Walking	0.90	0.57	0.70	28850
accuracy	0.79	0.79	0.79	1.0
macro avg	0.80	0.77	0.77	177246
weighted avg	0.80	0.79	0.79	177246

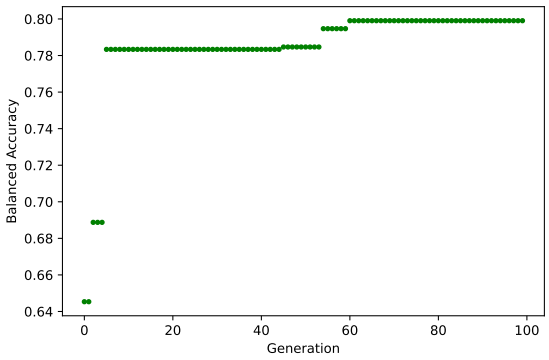
(b) Classification results of evolved CMMs tested on 177k samples of HAR dataset with features based on triaxial accelerometer, triaxial gyroscope, and GPS.

	precision	recall	f1-score	support
Active	0.72	0.66	0.69	46386
Driving	0.91	0.75	0.82	20332
Inactive	0.81	0.91	0.85	70456
Walking	0.75	0.69	0.72	27588
accuracy	0.78	0.78	0.78	1.0
macro avg	0.79	0.75	0.77	164762
weighted avg	0.78	0.78	0.78	164762

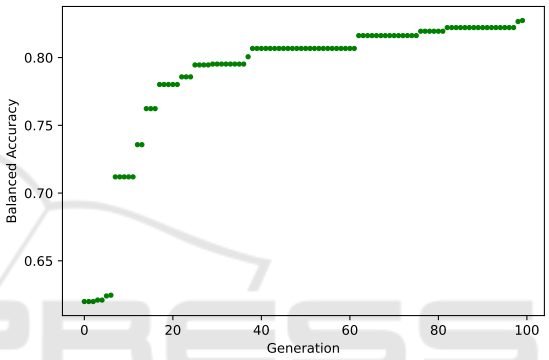
(c) Classification results of evolved CMMs tested on 164k samples of HAR dataset with features based on triaxial accelerometer, triaxial gyroscope, GPS, and magnetometer.

6 FURTHER RESEARCH

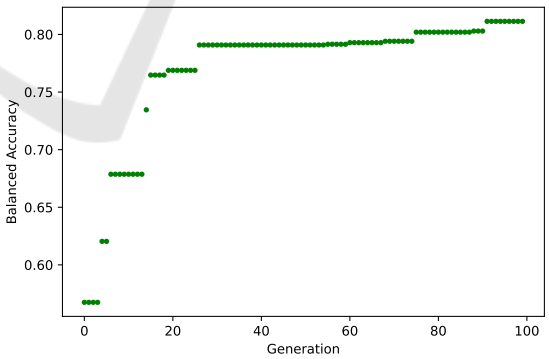
The presented approach offers a potential interface to adapt the evolution principles towards control of diversity using, e.g., self-betting or external-betting mechanisms as proposed in (Reichhuber and Tomforde, 2021). There are two options to enrich the algorithm with betting capabilities. Either each individual is given the possibility to bet on itself (self-betting Evolutionary Algorithm), or we introduce another bet population which is evolved simultaneously to the GCCs and its only objective is to learn how to bet on the individuals of the current population.



(a) Best fitness value history of evolved CMMs tested on 182k samples of HAR dataset with features based on triaxial-accelerometer sensor and GPS.

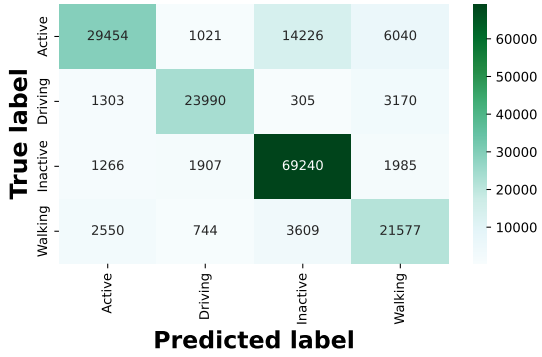


(b) Best fitness value history of evolved CMMs tested on 177k samples of HAR dataset with features based on triaxial accelerometer, triaxial gyroscope, and GPS.

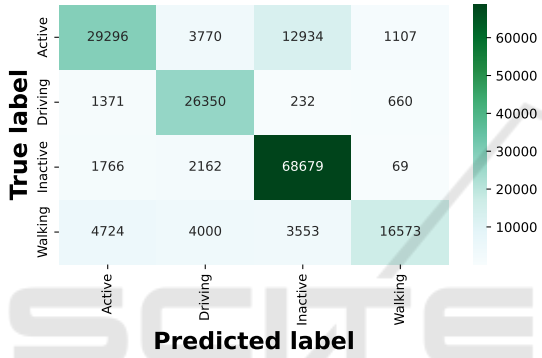


(c) Best fitness value history of evolved CMMs tested on 164k samples of HAR dataset with features based on triaxial accelerometer, triaxial gyroscope, GPS, and magnetometer.

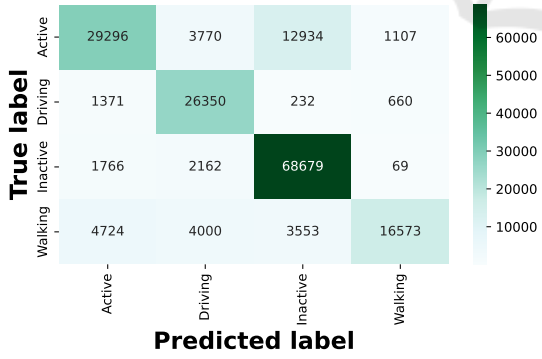
Figure 6: Fitness value history over 100 generations of evolving CMMs tested on the datasets HAR_1, HAR_2, and HAR_3.



(a) Confusion matrix of evolved CMMs tested on 182k samples of HAR dataset with features based on triaxial-accelerometer sensor and GPS.



(b) Confusion matrix of evolved CMMs tested on 177k samples of HAR dataset with features based on triaxial accelerometer, triaxial gyroscope, and GPS.



(c) Confusion matrix of evolved CMMs tested on 164k samples of HAR dataset with features based on triaxial accelerometer, triaxial gyroscope, GPS, and magnetometer.

Figure 7: Confusion matrices of evolving CMMs tested on the datasets HAR_1, HAR_2, and HAR_3.

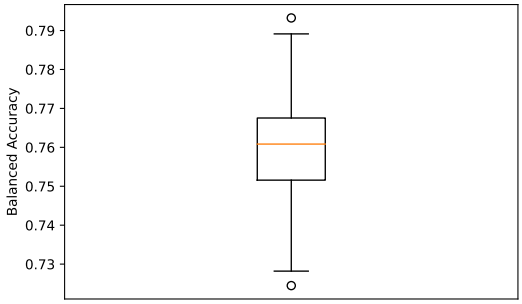


Figure 8: Boxplot diagram with the 25th and the 75th percentile of the balanced accuracy distribution found with the baseline reference. In 100000 iterations the best-found initialisations of GCCs resulted in a maximum balanced accuracy of 79.324 % and a median (orange line) of 76.083 %. The tests were driven on the data set `HAR_0`.

Since the latter would increase the time complexity of the algorithm, a solution for reducing the time complexity of evolving GCCs is required. Due to the matrix inversion, the time complexity of training a single CMM with n samples of dimensions d in k iterations of EM updates is $O(nkd^3)$. Since the latter is required for each fitness update of the individuals in a population, there are two more factors: (i) the number of generations g and (ii) the population size N that needs to be considered. In total, the time complexity for the evolved GCC algorithm is $O(gN(nkd^3))$.

However, for the experiments, we were able to limit the actual costs to an affordable amount of computing time. That is, we limited the number of EM iterations to a maximum of 100 and also used a minimum update delta of $1e-3$ as stop criterion, which reduces the actual duration to a spendable amount of computing time.

Regarding the high time complexity, a further improvement of the evolved CMM algorithm would be a replacement of the EM updates with a low-cost approximation. Using fast Incremental Gaussian Mixture Models, as presented in (Pinto and Engel, 2015), and limiting the stream size s for each individual in the population, the time complexity of the algorithm can be reduced to $O(gN(skd^2))$. However, the increasing performance lost needs to be evaluated in further experiments.

For general improvements of the HAR prediction, there might be taken more features into account. These might be adapted in an automated feature selection procedure tailored for a specific set of HAR activities. A good starting point for this, can be found in the library Time Series Feature Extraction Library (TSFEL) (Barandas et al., 2020) where the authors provide a list of general purpose time series features. For further improvements to the prediction performance, the datasets might be uniformed to a bench-

mark set including other daily life activities, like in (Leutheuser et al., 2013).

7 SUMMARY

This article presented a novel approach of evolving Gaussian Mixture Models (GMMs) that are applied to classification tasks, therefore called Classifier based on GMM (CMM). We explained that GMM and their particular variant of CMM are powerful tools that have been shown to obtain very good results in many domains and data sets. The current state of the art in training them lies in the usage of k-means and expectation maximisation, resulting in the most appropriate shape of the Gaussians. However, this is characterised by strong efforts in the training procedure. In contrast, our approach aims at utilising the principles of evolutionary computation. Hence, we presented our methodology in detail, including the encoding scheme, the definition of two novel genetic operators (i.e. *mutation* and *recombination*), and the resulting steps of the evolutionary process. Furthermore, a baseline reference of random K-Means initialisation is presented. In an experimental section, we demonstrated the potential benefit of evolved GMMs/CMMs using human activity recognition (HAR) as challenging use case. In considering various data sets of HAR, we analysed the capability of evolved CMMs to predict the correct activities. In total, a balanced accuracy of above 80 % has been achieved, which is particularly comparable to other approaches of the state-of-the-art while simultaneously allowing for novel advantages from the evolutionary process.

REFERENCES

Barandas, M., Folgado, D., Fernandes, L., Santos, S., Abreu, M., Bota, P., Liu, H., Schultz, T., and Gamboa, H. (2020). Tsfel: Time series feature extraction library. *SoftwareX*, 11:100456.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag.

Blanco, A., Delgado, M., and Pegalajar, M. C. (2001). A real-coded genetic algorithm for training recurrent neural networks. *Neural networks*, 14(1):93–105.

Chen, C., Wang, C., Hou, J., Qi, M., Dai, J., Zhang, Y., and Zhang, P. (2020). Improving accuracy of evolving gmm under gpgpu-friendly block-evolutionary pattern. *International Journal of Pattern Recognition and Artificial Intelligence*, 34(03):2050006.

D'Angelo, M., Gerasimou, S., Ghahremani, S., Grohmann, J., Nunes, I., Pournaras, E., and Tomforde, S. (2019). On learning in collective self-adaptive systems: state of practice and a 3d framework. In *Proc. of 14th Int. Symp. on Software Engineering for Adaptive and Self-Managing Systems*, pages 13–24.

del Alamo, C. M., Gil, F. C., Gomez, L. H., et al. (1996). Discriminative training of gmm for speaker identification. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 1, pages 89–92. IEEE.

Deriche, M., Aliyu, S. O., and Mohandes, M. (2019). An intelligent arabic sign language recognition system using a pair of lmcs with gmm based classification. *IEEE Sensors Journal*, 19(18):8067–8078.

Garcia-Gonzalez, D., Rivero, D., Fernandez-Blanco, E., and Luaces, M. R. (2020). A public domain dataset for real-life human activity recognition using smartphone sensors. *Sensors*, 20(8):2200.

Goldberg, D. E. et al. (1990). *Real-coded genetic algorithms, virtual alphabets and blocking*. Citeseer.

Gruhl, C., Sick, B., and Tomforde, S. (2021). Novelty detection in continuously changing environments. *Future Generation Computer Systems*, 114:138–154.

Heck, L. P. and Chou, K. C. (1994). Gaussian mixture model classifiers for machine monitoring. In *Proceedings of ICASSP'94. IEEE International Conference on Acoustics, Speech and Signal Processing*, volume 6, pages VI–133. IEEE.

Holland, J. (1975). Adaptation in natural and artificial systems, univ. of mich. press. *Ann Arbor*.

Huang, Y., Englehart, K., Hudgins, B., and Chan, A. (2005). A gaussian mixture model based classification scheme for myoelectric control of powered upper limb prostheses. *IEEE Transactions on Biomedical Engineering*, 52(11):1801–1811.

Jänicke, M., Tomforde, S., and Sick, B. (2016). Towards self-improving activity recognition systems based on probabilistic, generative models. In *2016 IEEE International Conference on Autonomic Computing (ICAC)*, pages 285–291. IEEE.

Kephart, J. and Chess, D. (2003). The Vision of Autonomic Computing. *IEEE Computer*, 36(1):41–50.

Kong, Y. and Fu, Y. (2018). Human action recognition and prediction: A survey. *arXiv preprint arXiv:1806.11230*.

Lau, S. L. and David, K. (2010). Movement recognition using the accelerometer in smartphones. In *2010 Future Network & Mobile Summit*, pages 1–9. IEEE.

Leutheuser, H., Schuldhaus, D., and Eskofier, B. M. (2013). Hierarchical, multi-sensor based classification of daily life activities: comparison with state-of-the-art algorithms using a benchmark dataset. *PloS one*, 8(10):e75196.

Li, M. and Narayanan, S. (2011). Robust talking face video verification using joint factor analysis and sparse representation on gmm mean shifted supervectors. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1481–1484. IEEE.

Martis, R. J., Chakraborty, C., and Ray, A. K. (2009). A two-stage mechanism for registration and classifica-

tion of ecg using gaussian mixture model. *Pattern Recognition*, 42(11):2979–2988.

McLachlan, G. J. and Rathnayake, S. (2014). On the number of components in a gaussian mixture model. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 4(5):341–355.

Müller-Schloer, C. and Tomforde, S. (2017). *Organic Computing-Technical Systems for Survival in the Real World*. Springer.

Ozerov, A., Lagrange, M., and Vincent, E. (2011). Gmm-based classification from noisy features. In *International Workshop on Machine Listening in Multisource Environments (CHiME 2011)*.

Pinto, R. C. and Engel, P. M. (2015). A fast incremental gaussian mixture model. *PloS one*, 10(10):e0139931.

Qi, P. and Wang, L. (2011). Experiments of gmm based speaker identification. In *2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, pages 26–31. IEEE.

Reichhuber, S. and Tomforde, S. (2021). Bet-based evolutionary algorithms: Self-improving dynamics in offspring generation. In *ICAART (2)*, pages 1192–1199.

Scrucca, L. and Serafini, A. (2019). Projection pursuit based on gaussian mixtures and evolutionary algorithms. *Journal of Computational and Graphical Statistics*, 28(4):847–860.

Velez, D. R., White, B. C., Motsinger, A. A., Bush, W. S., Ritchie, M. D., Williams, S. M., and Moore, J. H. (2007). A balanced accuracy function for epistasis modeling in imbalanced datasets using multifactor dimensionality reduction. *Genetic Epidemiology: the Official Publication of the International Genetic Epidemiology Society*, 31(4):306–315.

You, C. H., Lee, K. A., and Li, H. (2010a). Gmm-svm kernel with a bhattacharyya-based distance for speaker recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1300–1312.

You, C. H., Li, H., and Lee, K. A. (2010b). A gmm-supervector approach to language recognition with adaptive relevance factor. In *2010 18th European Signal Processing Conference*, pages 1993–1997. IEEE.

Zhao, X., Zhang, C., Yang, B., and Li, P. (2011). Adaptive knot placement using a gmm-based continuous optimization algorithm in b-spline curve approximation. *Computer-Aided Design*, 43(6):598–604.