# End-to-End Learning of Fisher Vector Encodings for Part Features in Fine-Grained Recognition

Dimitri Korsch[1(✉)], Paul Bodesheim[1], and Joachim Denzler[1,2,3]

[1] Computer Vision Group, Friedrich-Schiller-University Jena, Jena, Germany
dimitri.korsch@uni-jena.de
[2] Michael Stifel Center Jena for Data-Driven and Simulation Science, Jena, Germany
[3] German Aerospace Center (DLR), Institute for Data Science, Jena, Germany

**Abstract.** Part-based approaches for fine-grained recognition do not show the expected performance gain over global methods, although explicitly focusing on small details that are relevant for distinguishing highly similar classes. We assume that part-based methods suffer from a missing representation of local features, which is invariant to the order of parts and can handle a varying number of visible parts appropriately. The order of parts is artificial and often only given by ground-truth annotations, whereas viewpoint variations and occlusions result in not observable parts. Therefore, we propose integrating a Fisher vector encoding of part features into convolutional neural networks. The parameters for this encoding are estimated by an online EM algorithm jointly with those of the neural network and are more precise than the estimates of previous works. Our approach improves state-of-the-art accuracies for three bird species classification datasets.

**Keywords:** End-to-end learning · Fisher vector encoding · Part-based fine-grained recognition · Online EM algorithm

## 1 Introduction

Part- or attention-based approaches [9,10,15,48,49,51] are common choices for fine-grained visual categorization because they explicitly focus on small details that are relevant for distinguishing highly similar classes, e.g., different bird species. Quite surprisingly, methods that perform the categorization with global image features [7,27,33,34,41,52] also achieve excellent results. It is hard to tell from the empirical results reported in the literature which general approach (global or part-based) is superior, given that all of them show comparable results in terms of recognition performance. We hypothesize that part-based algorithms cannot exploit their full potential due to the problems that arise from

**Supplementary Information** The online version contains supplementary material available at https://doi.org/10.1007/978-3-030-92659-5_9.


C. Bauckhage et al. (Eds.): DAGM GCPR 2021, LNCS 13024, pp. 142–158, 2021.
https://doi.org/10.1007/978-3-030-92659-5_9

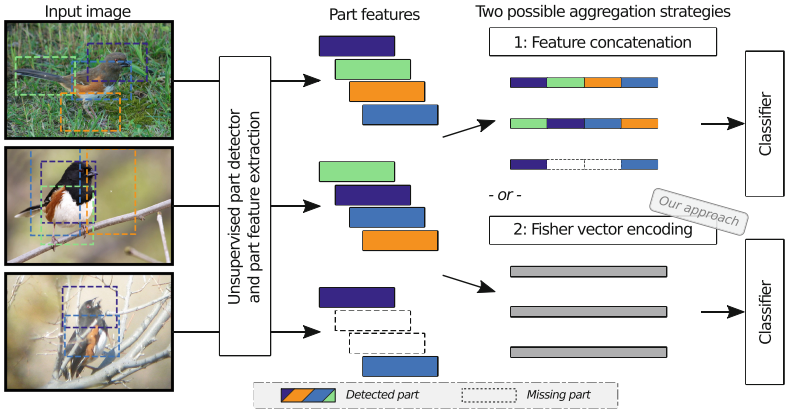



**Fig. 1.** When some parts can not be detected because they are not visible, the resulting gaps of missing features need to be filled when features are concatenated. Furthermore, the semantic meaning of extracted part features is not clear when applying unsupervised part detection algorithms that do not preserve a consistent order of part features. With our approach, we compute a Fisher vector encoding as a unified representation of fixed length for an arbitrary number of unordered part features, which can be used by any type of classifier, including simple linear classifiers and fully-connected layers in a deep neural network.

the initial detection of parts, especially regarding a unified representation of individual part features after the detection. Since learning individual part detectors requires part annotations and thus additional, time-consuming efforts by domain experts, methods for unsupervised part detection have been developed that already obtain remarkable classification results [10,24,49]. However, unsupervised part detection faces various challenges, such as missing parts caused by different types of occlusions and parts with ambiguous semantic meaning, as shown in Fig. 1. Hence, it remains unclear whether detected parts are reasonable and semantically consistent. Furthermore, part-based classifiers usually require a fixed number of parts to be determined for each image and a pre-defined order of the extracted part features. These are strong restrictions for the application, especially when considering varying poses and viewpoints that lead to hidden parts. We believe that a common way for representing a *varying number of unordered part features* obtained from every single image is rarely used in the context of fine-grained classification.

Fisher vector encoding (FVE) [30,31] is a well-known feature transformation method typically applied to local descriptors of key points. When applied to part features, it allows for computing a unified representation of fixed length for each image, independent of the number of detected parts (Fig. 1). Note that the problem of missing parts can also be observed for ground-truth part annotations of fine-grained bird species datasets [3,42,44]. Due to various poses and occlusions, some bird parts are not visible and cannot be annotated. These datasets are often used to evaluate the performance of classifiers without taking the part

detection into account, and this requires a proper gap-filling strategy for missing parts. In contrast, no gap-filling is needed when using the FVE. Furthermore, FVE allows for neglecting an artificial order of parts since there is usually no natural order of parts, and each part should be treated equally.

Applying FVE together with CNNs has been done differently in the past: either GMM and FVE are computed after training the CNN model [6,36,50] such that only FVE parameters are adapted to the CNN features and not vice versa, or the FVE has been realized as a trainable layer [1,11,40,45]. Although the latter allows for end-to-end training of GMM parameters, artificial constraints are required to obtain reasonable values (e.g., positive variances). Furthermore, only the classification loss influences the mixture estimations, and there is no objective involved for modeling the data distribution correctly. In contrast, we propose to realize the FVE as a differentiable feature transformation based on a GMM estimated with an iterative EM algorithm using mini-batch updates of the parameters. First, the differentiable transformation allows end-to-end training of the feature extraction and classification weights. Next, the FVE directly influences the feature extraction within the CNN such that the features adapt to the encoding. Finally, since the parameters of the GMM are estimated with an EM algorithm instead of gradient descent, the resulting GMM describes the input data more precisely.

In our experiments, we show that the FVE outperforms other feature aggregation methods and that our approach estimates the GMM parameters more precisely than gradient descent methods. Furthermore, our approach described in Sect. 3 improves state-of-the-art accuracies for three fine-grained bird species categorization datasets: CUB-200-2011 [44] (from 90.3% to 91.1%), NA-Birds [42] (from 89.2% to 90.3%), and Birdsnap [3] (from 84.3% to 84.9%).

## 2 Related Work

First, we discuss related work on FVE in the context of deep neural networks. Some of these approaches either do not allow for learning the GMM parameters end-to-end but rather estimate parameters separately after neural network training. Others treat GMM parameters as conventional network parameters learned without any clustering objective by artificially enforcing reasonable values for the mixture parameters. Second, we review existing algorithms for iterative EM algorithms since we borrow ideas from these approaches. Surprisingly, none of these iterative approaches has been integrated in a CNN yet. Third, we list current state-of-the-art techniques for fine-grained categorization, mainly focusing on part-based methods. Some approaches [6,50] use FVE in fine-grained approaches, but all of them deploy the encoding in an offline manner, i.e., the FVE is used *after* learning CNN parameters for feature extraction.

### 2.1 Variants of Deep Fisher Vector Encoding (Deep FVE)

Simonyan *et al.* [35] presented a Fisher vector layer for building deep networks. They encode an input image or pre-extracted SIFT features in multiple lay-

ers with an FVE, but the entire network is trained greedily layer-by-layer and not end-to-end due to some restrictions. Sydorov *et al.* [37] suggested another deep architecture that learns an FVE by updating GMM parameters based on gradients that are backpropagated from a Hinge loss of an SVM classifier. Cimpoi *et al.* [6] proposed a CNN together with an FVE for local CNN features, and the resulting feature representation is used by a one-vs-rest SVM. Song *et al.* [36] improved this approach, but still without end-to-end learning.

In contrast to the methods above, Wieshollek *et al.* [45] and Tang *et al.* [40] deploy the FVE directly in a neural network. As a result, features are learned jointly with the classification and mixture model parameters. However, although GMM parameters are estimated jointly with other network parameters using gradient descent, the training procedure has some drawbacks. First, artificial constraints have to be applied to obtain reasonable mixture parameters, e.g., positive variances. Next, due to the formulas for the FVE, the resulting gradients cause numerically unstable computations. Finally, these approaches require a proper initialization of the mixture parameters, which implies the computation of the features of the entire dataset. Performing such an initialization on large dataset results in an unreasonable computation and storage overhead. Arandjelovic *et al.* [1] employ a simplified version of the FVE, called Vector of Locally Aggregated Descriptors (VLAD) [20], and rewrite its computation in terms of trainable network parameters that are optimized end-to-end via gradient descent. However, for this NetVLAD-Layer [1] as well as for [40,45], it is arguable whether estimated GMM parameters are reasonable due to gradient-based optimization with back-propagation. In contrast, our proposed method does not require any preliminary initialization since it uses an iterative EM algorithm to estimate the mixture parameters end-to-end, leading to meaningful cluster representations.

### 2.2 EM Algorithms and GMM Estimation

The standard technique for estimating GMM parameters is the EM algorithm. It is an iterative process of alternating between maximum likelihood estimation of mixture parameters and computing soft assignments of samples to the mixture components. In the default setting, all samples are used in both steps, but this leads to an increased runtime for large-scale datasets. To reduce the computational costs, one can only use a subset of samples for the parameter estimation or rely on existing online versions of the EM algorithm [4,29]. For example, Cappé and Moulines [4] approximate the expectation over the entire dataset with an exponential moving average over batches of the data. Based on this work, Chen *et al.* [5] propose a variance reduction of the estimates in each step, which results in faster and more stable convergence. A comparison of these algorithms is carried out by Karimi *et al.* [21], who also introduce a new estimation algorithm. Further approaches [2,13,26] propose similar solutions with different applications and motivations for the iterative parameter update.

In our work, we employ the ideas of iterative parameter update coupled with a bias correction. Furthermore, we demonstrate how to integrate these ideas in a neural network and estimate the parameters jointly with the network weights.

### 2.3 Fine-grained Visual Categorization

In the literature, two main directions can be observed for fine-grained recognition: global and part-based methods. Global methods use the input image as a whole and employ clever strategies for pre-training [7], augmentation [25,41], or pooling [27,33,34,52]. In contrast, part- or attention-based approaches apply sophisticated detection techniques to determine interesting image regions and to extract detailed local features from these patches. It results in part features as an additional source of information for boosting the classification performance.

He *et al.* [15] propose a reinforcement learning method for estimating how many and which image regions are helpful for the classification. They use multi-scale image representations for localizing the object and afterward estimate discriminative part regions. Ge *et al.* [10] present the current state-of-the-art approach on the CUB-200-2011 dataset. Based on weakly supervised instance detection and segmentations, part proposals are generated and constrained by a part model. The final classification is performed with a stacked LSTM classifier and context encoding. The method of Zhang *et al.* [49] also yields good results on the CUB-200-2011 and the NA-Birds dataset. Expert models arranged in multiple stages predict class assignments and attention maps that the final expert uses to crop the image and refine the observed data. Finally, a gating network is used to weigh the decisions of the individual experts.

Compared to the previous approaches, we use a different part detection method described at the beginning of the next section before presenting the details of our proposed FVE for part features.

## 3 Fisher Vector Encoding (FVE) of Part Features

In this section, we present our approach for an FVE of part features, which allows for joint end-to-end learning of all parameters, i.e., the parameters of the underlying GMM and the parameters of the CNN that computes the part features. It can be applied to any set of extracted parts from an image. Hence it is possible to combine it with different part detection algorithms. In this paper, we use the code[1] for a part detection method provided by Korsch *et al.* [24]. The authors use an initial classification of the entire input image to identify features used for this classification. Then, the pixels in the receptive field of these features are clustered and divided into candidate regions. Bounding boxes are estimated around these regions and used as parts in the final part-based classification.

As shown in Fig. 2, given a set of parts specified by their corresponding image regions, we propose the computation of a set of local features for each part with a CNN. We denote the output of a CNN as a *ConvMap* $C \in \mathbb{R}^{H \times W \times D}$, that consists of $N = H \cdot W$ local $D$-dimensional features $\mathcal{X} = \{\vec{x}_1, \ldots, \vec{x}_N\}$. Usually, CNNs contain *global average pooling (GAP)* to reduce $\mathcal{X}$ to a single feature representation: $GAP(\mathcal{X}) = \frac{1}{N} \sum_{n=1}^{N} \vec{x}_n$. Common part-based approaches [10, 24,46,47] extract a set of *ConvMaps* $\mathcal{C} = \{C_1, \ldots, C_T\}$ from a single image

[1] https://github.com/cvjena/l1_parts.

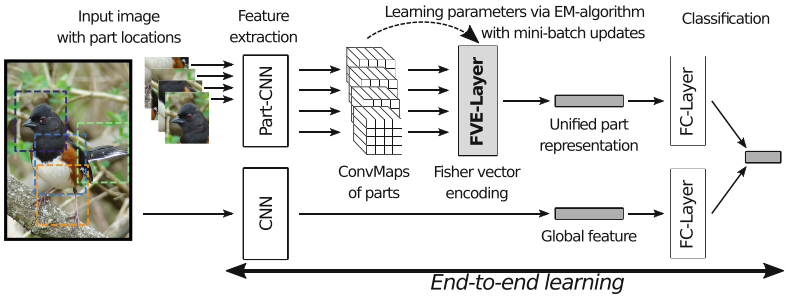



**Fig. 2.** Overview of our proposed method. During training, we estimate the parameters of the GMM that leads to the FVE using an EM-algorithm with mini-batch updates described in Sect. 3.2. The resulting FVE-Layer, which is explained in Sect. 3.3, can be integrated in any deep network architecture. We use this new layer for computing a unified part representation that aggregates local features extracted from *ConvMaps* of a CNN. Our approach enables joint end-to-end learning of both CNN parameters and GMM parameters for the FVE.

$I$ by processing $T$ image regions and use GAP for each *ConvMap* followed by concatenation of the resulting $T$ part features. Since each part $t$ is represented by $N$ local features of the corresponding *ConvMap* $C_t$, an image $I$ can be described by a set of $N \cdot T$ local *ConvMap* features $\mathcal{X}_I = \{\vec{x}_{1,1}, \ldots, \vec{x}_{n,t}, \ldots, \vec{x}_{N,T}\}$. We then use FVE to transform this set into a single feature: $FVE(\mathcal{X}_I) = \vec{f} \in \mathbb{R}^{\hat{D}}$. Note that the set $\mathcal{X}_I$ is just another representation for the set of *ConvMaps* $\mathcal{C}$, and we can also directly write $FVE(\mathcal{C})$ instead of $FVE(\mathcal{X}_I)$ to indicate that the FVE is a transformation of the *ConvMaps* $\mathcal{C}$.

### 3.1 Fisher Vector Encoding

First, we assume that the CNN computes local features *i.i.d.* from an input image. We further assume that all local descriptors $\mathcal{X}_I$ of the extracted *ConvMaps* $\mathcal{C}$ come from the same distribution with density function $p(\vec{x}_{n,t})$, represented by a finite mixture model with $K$ components: $p(\vec{x}_{n,t}|\Theta) = \sum_{k=1}^{K} \alpha_k p_k(\vec{x}_{n,t}|\theta_k)$ with mixture weights $\alpha_k$ that add up to 1 ($\sum_{k=1}^{K} \alpha_k = 1$), and model parameters $\Theta = \{\alpha_1, \theta_1, \ldots, \alpha_K, \theta_K\}$. Without any prior knowledge, let the density function of each component be a Gaussian distribution with mean vector $\vec{\mu}_k$ and diagonal covariance matrix $\vec{\sigma}_k$: $p_k(\vec{x}_{n,t}|\theta_k) = \mathcal{N}(\vec{x}_{n,t}|\vec{\mu}_k, \vec{\sigma}_k)$ leading to a Gaussian Mixture Model (GMM) with parameters $\Theta = \{\alpha_1, \vec{\mu}_1, \vec{\sigma}_1 \ldots \alpha_K, \vec{\mu}_K, \vec{\sigma}_K\}$.

Following Jaakola and Haussler [19], and Perronnin *et al.* [30,31], the FVE is derived by considering the gradients of the log-likelihood with respect to the GMM parameters $\Theta$ and assuming independence of the part features: $\mathcal{F}_\Theta(\mathcal{C}) = \sum_{n,t=1}^{N,T} \nabla_\Theta \log p(\vec{x}_{n,t}|\Theta)$. These gradients, also called Fisher scores, describe how parameters contribute to the process of generating a particular feature. We use the approximated normalized Fisher scores introduced by [30,31]:

$$\mathcal{F}_{\vec{\mu}_k}(\mathcal{C}) = \frac{1}{\sqrt{NT\alpha_k}} \sum_{n,t=1}^{N,T} w_{n,t,k} \left( \frac{\vec{x}_{n,t} - \vec{\mu}_k}{\vec{\sigma}_k} \right), \tag{1}$$

$$\mathcal{F}_{\vec{\sigma}_k}(\mathcal{C}) = \frac{1}{\sqrt{2\,NT\alpha_k}} \sum_{n,t=1}^{N,T} w_{n,t,k} \left( \frac{(\vec{x}_{n,t} - \vec{\mu}_k)^2}{\vec{\sigma}_k^2} - 1 \right) \tag{2}$$

as feature encoding with $w_{n,t,k} = \frac{\alpha_k p(\vec{x}_{n,t}|\theta_k)}{\sum_{l=1}^{K} \alpha_l p(\vec{x}_{n,t}|\theta_l)}$ denoting the soft assignment of the feature $\vec{x}_{n,t}$ to a component $k$.

Finally, these scores can be computed for all parameters $\vec{\mu}_k$ and $\vec{\sigma}_k$ of the estimated GMM. We use the concatenation of the scores as FVE of the part features, which results in a unified representation of dimension $\hat{D} = 2KD$ that is independent of the number of part features $T$.

### 3.2 Estimation of the Mixture Model Parameters

The computations of the FVE require a GMM, and we illustrate two ways for estimating its parameters from data jointly with the CNN parameters. In our experiments, we use both methods and compare them by the classification accuracy and quality of the estimated GMM parameters.

**Gradient-Based Estimation.** This idea is covered in different works [1,40,45]: since all of the operations in Eqs. (1) and (2) are differentiable, it is straightforward to implement the FVE as a differentiable FVE-Layer such that the parameters $\Theta$ are estimated via gradient descent. However, the GMM constraints (positive variances and prior weights adding up to one) need to be enforced. Wieschollek *et al.* [45] propose to model the variances $\sigma_k^2 = \epsilon + \exp(s_k)$ and the mixture weights $\alpha_k = \frac{\text{sigm}(a_k)}{\sum_j \text{sigm}(a_j)}$ by estimating $s_k$ and $a_k$ instead of $\sigma_k^2$ and $\alpha_k$.

**Online-EM Estimation.** Different variants for an online EM algorithm can be found in the literature [2,4,5,21,26,29]. The main idea is to approximate the expectations over the entire dataset with exponential moving averages (EMAs) over batches of the data: $\Theta[t] = \lambda \cdot \Theta[t-1] + (1-\lambda) \cdot \Theta^{\text{new}}$ with $\lambda \in (0,1)$ and $[t]$ indicating the training step $t$. We follow this approach and propose an FVE-Layer with online parameter estimation via EMAs. It is worth mentioning that the parameters of the widely used batch-normalization layer [18] are also estimated with EMAs. However, our FVE-Layer differs from a batch-normalization layer in two ways. First, we estimate a mixture of Gaussians and not only the mean and variance of the inputs. Second, we encode the input according to Eqs. (1) and (2) instead of whitening the inputs. Additionally, we perform bias correction via $\hat{\Theta}[t] = \frac{\Theta[t]}{1-\lambda^t}$ since plain EMAs are biased estimators. Similar bias correction is also done in the Adam optimizer [23].

Finally, we have observed that some of these $D$-dimensional local feature vectors ($H \cdot W$ vectors for each of the $T$ parts) have low $L^2$-norm, especially if the corresponding receptive field mainly contains background pixels. However, since we are only interested in using local features that exceed a certain

activation level, i.e., that carry important information, we include an additional filtering step for the local feature vectors before both the estimation of the GMM parameters and the computation of the FVE. We only use local features with an $L^2$-norm greater than the mean $L^2$-norm of all local features obtained from the same image. We found that this filtering leads to more stable and balanced estimates for the GMM parameters during our experiments.

### 3.3 Training with the FVE-Layer

We implement the proposed FVE-Layer utilizing the calculations from Eqs. (1) and (2) as well as the online parameter estimation introduced in the previous section. For end-to-end training of the CNN layers preceding the FVE-Layer, we estimate the gradients of the encoding w.r.t. the inputs similar to [45]:

$$\frac{\partial \mathcal{F}_{\mu_{k,d}}(\mathcal{C})}{\partial x_{n,t,d_*}} = \frac{1}{\sqrt{NT\alpha_k}} \left[ \frac{\partial w_{n,t,k}}{\partial x_{n,t,d_*}} \left( \frac{x_{n,t,d} - \mu_{k,d}}{\sigma_{k,d}} \right) + \delta_{d,d_*} \frac{w_{n,t,k}}{\sigma_{k,d_*}} \right], \tag{3}$$

$$\begin{aligned} \frac{\partial \mathcal{F}_{\sigma_{k,d}}(\mathcal{C})}{\partial x_{n,t,d_*}} &= \frac{1}{\sqrt{2\,NT\alpha_k}} \left[ \frac{\partial w_{n,t,k}}{\partial x_{n,t,d_*}} \left( \frac{(x_{n,t,d} - \mu_{k,d})^2}{(\sigma_{k,d})^2} - 1 \right) \right. \\ &== \left. \frac{1}{\sqrt{2\,NT\alpha_k}} + \delta_{d,d_*} \frac{2 w_{n,t,k} (x_{n,t,d_*} - \mu_{k,d_*})}{(\sigma_{k,d_*})^2} \right]. \end{aligned} \tag{4}$$

In both equations, we use $\delta_{d,d_*}$ to denote the Kronecker delta being 1 if $d = d_*$ and 0 else, as well as the derivative of $w_{n,t,k}$ w.r.t. $x_{n,t,d_*}$ that is given by $\frac{\partial w_{n,t,k}}{\partial x_{n,t,d_*}} = w_{n,t,k} \left( -\frac{(x_{n,t,d_*} - \mu_{k,d_*})}{(\sigma_{k,d_*})^2} + \sum_{\ell=1}^{K} w_{n,t,\ell} \frac{(x_{n,t,d_*} - \mu_{\ell,d_*})}{(\sigma_{\ell,d_*})^2} \right)$. Further details for the derivation of these gradients can be found in the supplementary material.

Though these gradients are computed within the deep learning framework by the autograd functionality, it is important to mention that we observed some numerical instabilities during the training, especially with high dimensional mixture components. To circumvent this issue, we perform an auxiliary classification on the inputs of the FVE-Layer, similar to Szegedy *et al.* [38]. The auxiliary classification branch consists of a global average pooling and a linear layer. Finally, we combine the resulting auxiliary loss with the loss computed from the prediction on the encoded part features: $\mathcal{L}_{parts} = \beta \cdot \mathcal{L}_{aux} + (1 - \beta) \cdot \mathcal{L}_{FVE}$. We set $\beta$ to 0.5 and multiply it by 0.5 after 20 epochs, such that the effect of the auxiliary classification decreases over time. Our motivation behind the initial value of $\beta$ is to give both losses equal impact at the beginning and to increase the impact of the encoded part features as the training goes on. We tested other initial values, but they had minor effects on the classification performance, except that disabling the auxiliary loss resulted in degraded training stability, as mentioned before. Hence, we have chosen the $\beta$ value that matches best the arithmetic mean of the losses. Furthermore, we omit the auxiliary branch for the final classification and perform the part classification entirely on the features encoded by the FVE.

The final loss, consisting of the losses computed from the global and the part predictions, is computed in a similar way: $\mathcal{L}_{final} = \frac{1}{2}(\mathcal{L}_{parts} + \mathcal{L}_{global})$. For cross-entropy, this combination is equivalent to computing the final prediction as a geometric mean of the class probabilities or the arithmetic mean of the normalized log-likelihoods (see Sect. S2 in the supplementary material for more details).

## 4 Experimental Results

### 4.1 Datasets

We evaluate our method on widely used datasets for fine-grained categorization. First, we use three datasets for bird species recognition: *CUB-200-2011* [44], *NA-Birds* [42], and *Birdsnap* [3], since this is the most challenging domain when considering current state-of-the-art results with accuracies of around 90 % or less (see Table 1). For other fine-grained domains like aircraft, cars, or flowers, the methods already achieve accuracies above 95 %. The three bird datasets contain between 200 and 555 different species. CUB-200-2011 is the most popular fine-grained dataset for benchmarking because of its balanced sample distribution, but it is also the smallest one with only 5994 training and 5794 test images. The other two datasets are more imbalanced but contain much more training images: 23929 and 40871, respectively. Besides class labels, bounding boxes and part annotations are available for all three datasets.

Additionally, we evaluate our method on datasets for dogs and moths species to show the applicability of our approach for other domains. *Stanford Dogs* [22] consists of 120 classes and 20580 images with class labels and bounding box annotations. Since the entire dataset is part of the ImageNet dataset [8], we only use neural networks pre-trained on the iNaturalist 2017 dataset [43] to avoid pre-training on the test images. The *EU-Moths* dataset[2] contains 200 moth species common in Central Europe. Each of the species is represented by approximately 11 images. The insects are photographed manually and mainly on a relatively homogeneous background. We manually annotated bounding boxes for each specimen and used the cropped images for training. We trained the CNN on a random balanced split of 8 training and 3 test images per class.

### 4.2 Implementation Details

As a primary backbone of the presented method, we take the InceptionV3 CNN architecture [39]. We use the pre-trained weights proposed by Cui *et al.* [7]. They have pre-trained the network on the iNaturalist 2017 dataset [43] and could show that this is more beneficial for animal datasets than pre-training on ImageNet. For some experiments (Sect. 4.4 and 4.5), we also use a ResNet-50 CNN architecture [14] pre-trained on the ImageNet dataset [32].

[2] https://www.inf-cv.uni-jena.de/eu_moths_dataset.

**Table 1.** Comparison of our proposed FVE for part features with various state-of-the-art methods on three bird datasets (**bold** = best per dataset).

| Method | CUB-200-2011 | NA-Birds | Birdsnap |
|---|---|---|---|
| Cui *et al.* [7] | 89.3 | 87.9 | – |
| Stacked LSTM [10] | 90.3 | – | – |
| FixSENet-154 [41] | 88.7 | 89.2 | 84.3 |
| CS-Parts [24] | 89.5 | 88.5 | – |
| MGE-CNN [49] | 89.4 | 88.6 | – |
| WS-DAN [16] | 89.4 | – | – |
| PAIRS [12] | 89.2 | 87.9 | – |
| API-Net [53] | 90.0 | 88.1 | – |
| No Parts (baseline) | 89.5 ± 0.2 | 86.9 ± 0.1 | 81.9 ± 0.5 |
| GAP (parts of [24]) | 90.9 ± 0.1 | 89.9 ± 0.1 | 84.0 ± 0.2 |
| Gradient-based FVE (parts of [24]) | **91.2** ± 0.3 | **90.4** ± 0.1 | **85.3** ± 0.2 |
| EM-based FVE (parts of [24]) | **91.1** ± 0.2 | **90.3** ± 0.1 | **84.9** ± 0.2 |

For a fair comparison, we use fixed hyperparameters for every experiment. We train each model for 60 epochs with an AdamW optimizer [28], setting the learning rate to 2e-3 and $\alpha$ to 0.01. Due to limited GPU memory, we apply the gradient accumulation technique. We use a batch size of 12 and accumulate the gradients over four training iterations before we perform a weight update, which results in an effective batch size of 48. Furthermore, we repeat each experiment at least 5 times to observe the significance and robustness of the presented approach. The source code for our approach is publicly available on GitHub[3].

### 4.3 Fine-grained Classification

In our first experiment, we test our proposed FVE-Layer together with an unsupervised part detector that provides classification-specific parts (CS-Parts) [24]. However, in contrast to Korsch *et al.* [24], we use a separate CNN for calculating part features. This part-CNN is fine-tuned on the detected parts, and the extracted features are adapted to the FVE. Besides the part-CNN, we also extract features from the global image with another CNN. For the part-CNN, the prediction is performed based on the FVE of the part features, whereas the prediction on the global image is made based on standard CNN features. Both predictions are then weighted equally and summed up to the final prediction. The GMM parameters are either estimated via gradient descent or by our proposed online EM algorithm. After investigating the effect of both the number and dimension of the mixture components (see Sect. S3.2 in supplementary material), we use one component with a dimension of 2048. We select these hyperparameters since (1) this setup introduces the least number of additional

[3] https://github.com/cvjena/fve_experiments.

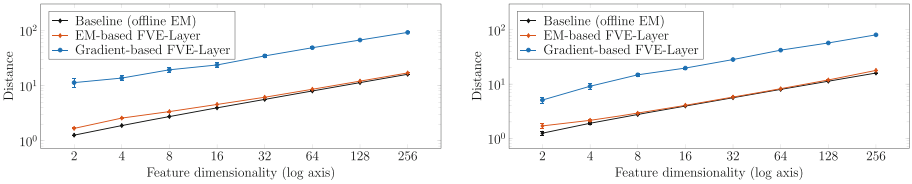
**Table 2.** Results on the Stanford Dogs and EU-Moths datasets. For Stanford Dogs, we only compare to methods that do not use ImageNet pre-training. Similar to our work, they utilize a pre-training on the iNaturalist 2017 dataset. This kind of pre-training results in a more fair comparison, since the training set of ImageNet contains the test set of Stanford Dogs.

| Method | Stanford Dogs | EU-Moths |
|---|---|---|
| Cui *et al.* [7] | 78.5 | – |
| DATL [17] (with [16]) | 79.1 | – |
| No Parts (baseline) | 77.5 ± 0.5 | 90.5 ± 0.5 |
| GAP (parts of [24]) | 77.8 ± 0.4 | 91.0 ± 0.5 |
| Gradient-based FVE (parts of [24]) | **79.1** ± 0.3 | **93.0** ± 1.2 |
| EM-based FVE (parts of [24]) | **79.2** ± 0.1 | **92.0** ± 0.9 |

parameters and (2) favors both GMM parameter estimation methods equally. We also compare the FVE method with results based on concatenated part features and GAP and with a baseline using only the prediction obtained from the global image (no parts). In our preliminary experiments (see Sect. S3.1 in supplementary material), we also showed the superiority of the joint training of the GMM parameters and the CNN weights against the conventional pipeline that is for example used in [6,36].

For bird species classification, Table 1 contains our results as well as accuracies reported in previous work. Our proposed FVE for part features performs best on all three datasets. The results for NA-Birds stand out, since our approach is the only one reaching an accuracy greater than 90 % on this challenging dataset Accuracies for dogs and moth species are shown in Table 2. Again, our FVE approach performs best, showing its suitability beyond the bird species domain.

Nevertheless, it is worth mentioning that the GMM parameter estimation method has a minor effect on classification accuracy. We observed in our preliminary experiments that the GMM parameters have little effect on the expressiveness of the final encoded feature. Even if the GMM parameters are initialized randomly and are not further adapted to the data, the classification performance remains equally high compared to the reported values in Table 1. This also explains why the Gradient-based estimation of the parameters performs on a par with the EM-based estimation, even though the GMM parameters do not change much from their initialization values (see Sect. S5 in supplementary material). Finally, we think the observation of why the randomly initialized GMM parameters perform as well as the trained parameters is out of the scope of this paper, and we would investigate this in the future.




(a) One component (b) Five components

**Fig. 3.** Comparison of the weighted normalized Euclidean distance (see Eq. (5)) on a generated dataset. The dataset consist of five classes and the experiment was performed with one and five GMM components.

### 4.4 Quality of the Estimated GMM Parameters

We now investigate the quality of the estimated GMM parameters with respect to the two proposed approaches (gradient-based vs. EM-based). For this purpose, we first compare them on a generated dataset consisting of feature vectors for five classes sampled from different normal distributions. The corresponding mean vectors (class centroids) are arranged on the unit sphere, and we set the variance of the class distributions to $\frac{1}{5}$ such that the features do not overlap. We trained a simple neural network consisting of the FVE-Layer and a linear layer as a classifier. The data dimension has been varied, using powers of two in the range 2 to 256, which also defines the dimensions of the mixture components. Each setup was repeated five times. As a baseline, we also estimated the GMM parameters with a standard EM algorithm independent of neural network training.

In Fig. 3, we visualize the normalized Euclidean distance of the estimated parameters averaged over the entire dataset. The normalized Euclidean distance for a single feature vector can be computed via:

$$D(\vec{x}_{n,t}|\theta) = \sum_k w_{n,t,k} \cdot \sqrt{\sum_d \frac{(x_{n,t,d} - \mu_{k,d})^2}{(\sigma_{k,d})^2}} \quad . \tag{5}$$

It is the sum of distances to the mean vectors of the mixture components, normalized by the corresponding variances and weighted by the soft-assignments $w_{n,t,k}$. Furthermore, we estimate the same distances for the CUB-200-2011 dataset, shown in Table 3. Both evaluations on synthetic and real data show that GMMs estimated by our proposed online EM algorithm fit the data much better, resulting in more precise clusters due to lower normalized Euclidean distances. Moreover, we show in the supplementary material that the gradient-based method changes the GMM parameters only slightly for two-dimensional data, whereas the online EM algorithm estimates parameters as good as the offline EM algorithm such that the mixtures match the data distributions well.

**Table 3.** Comparison of weighted normalized Euclidean distances (see Eq. (5)) for the three birds datasets, evaluated for two CNN architectures.

| | GMM Estimation for ResNet50 | | GMM Estimation for InceptionV3 | |
|---|---|---|---|---|
| | EM-based | Gradient-based | EM-based | Gradient-based |
| CUB-200-2011 | 21.49 (±0.68) | 46.85 (±0.18) | 33.21 (±0.13) | 37.15 (±0.18) |
| NA-Birds | 18.36 (±1.00) | 45.35 (±0.18) | 36.62 (±0.38) | 35.50 (±0.15) |
| Birdsnap | 11.08 (±2.01) | 46.75 (±0.23) | 32.75 (±0.31) | 35.62 (±0.30) |

**Table 4.** Ablation study on the CUB-200-2011 dataset with two different CNNs.

| Method | ResNet50 | InceptionV3 |
|---|---|---|
| Baseline CNN | 84.4 ± 0.3 | 89.5 ± 0.2 |
| Parts [24] + GAP | 79.7 ± 0.3 | 88.8 ± 0.2 |
| Parts [24] + FVE | 82.6 ± 0.4 | 89.1 ± 0.2 |
| Baseline CNN + Parts [24] + GAP | 85.9 ± 0.2 | 90.9 ± 0.1 |
| Baseline CNN + Parts [24] + FVE | 86.4 ± 0.2 | 91.1 ± 0.2 |

### 4.5 Ablation Study

In an ablation study, we investigate the impact of the FVE in the proposed approach. As seen in Fig. 2, our method consists of two branches: classification of the global image and classification based on estimated parts. In Table 4, we show accuracies achieved by the individual branches as well as by the combined classification for two CNN architectures. We see that part features with FVE result in better classification accuracies compared to GAP. This effect propagates to the final accuracy, resulting in improved classification performance.

## 5 Conclusions

In this paper, we have proposed a new FVE-Layer for aggregating part features of a CNN in the context of fine-grained categorization, which uses an online EM algorithm for estimating the underlying GMM jointly with other network parameters. With this layer, we are able to compute a unified fixed-length representation for a varying number of local part features, which allows a deep neural network to cope with missing parts as well as with an arbitrary order of part features, e.g., given by an unsupervised part detector. In our experiments, we have achieved state-of-the-art recognition accuracies on three fine-grained datasets for bird species classification: CUB-200-2011 (91.1 %), NA-Birds (90.3 %), and Birdsnap (84.9 %). Furthermore, we have shown that compared to existing deep FVE implementations, our online EM-based approach results in more accurate estimates of the mixture model.

## References


1. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 5297–5307 (2016)
2. Awwad Shiekh Hasan, B., Gan, J.Q.: Sequential EM for unsupervised adaptive gaussian mixture model based classifier. In: Perner, P. (ed.) MLDM 2009. LNCS (LNAI), vol. 5632, pp. 96–106. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-03070-3_8
3. Berg, T., Liu, J., Woo Lee, S., Alexander, M.L., Jacobs, D.W., Belhumeur, P.N.: Birdsnap: large-scale fine-grained visual categorization of birds. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2011–2018 (2014)
4. Cappe, O., Moulines, E.: On-line expectation-maximization algorithm for latent data models. J. R. Stat. Soc. Ser. B (Stat. Methodol.) **71**(3), 593–613 (2009)
5. Chen, J., Zhu, J., Teh, Y.W., Zhang, T.: Stochastic expectation maximization with variance reduction. In: Advances in Neural Information Processing Systems, pp. 7967–7977 (2018)
6. Cimpoi, M., Maji, S., Vedaldi, A.: Deep filter banks for texture recognition and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3828–3836 (2015)
7. Cui, Y., Song, Y., Sun, C., Howard, A., Belongie, S.: Large scale fine-grained categorization and domain-specific transfer learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2018)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255. IEEE (2009)
9. Fu, J., Zheng, H., Mei, T.: Look closer to see better: recurrent attention convolutional neural network for fine-grained image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2017)
10. Ge, W., Lin, X., Yu, Y.: Weakly supervised complementary parts models for fine-grained image classification from the bottom up. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3034–3043 (2019)
11. Gong, Y., Wang, L., Guo, R., Lazebnik, S.: Multi-scale orderless pooling of deep convolutional activation features. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8695, pp. 392–407. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10584-0_26
12. Guo, P., Farrell, R.: Aligned to the object, not to the image: a unified pose-aligned representation for fine-grained recognition. In: IEEE Winter Conference on Applications of Computer Vision, pp. 1876–1885. IEEE (2019)
13. Hasan, B.A.S., Gan, J.Q.: Unsupervised adaptive GMM for bci. In: 2009 4th International IEEE/EMBS Conference on Neural Engineering, pp. 295–298 (2009)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 770–778 (2016)
15. He, X., Peng, Y., Zhao, J.: Which and how many regions to gaze: focus discriminative regions for fine-grained visual categorization. Int. J. Comput. Vis. **127**, 1–21 (2019). https://doi.org/10.1007/s11263-019-01176-2
16. Hu, T., Qi, H., Huang, Q., Lu, Y.: See better before looking closer: weakly supervised data augmentation network for fine-grained visual classification. arXiv preprint arXiv:1901.09891 (2019)

17. Imran, A., Athitsos, V.: Domain adaptive transfer learning on visual attention aware data augmentation for fine-grained visual categorization. In: Bebis, G., et al. (eds.) ISVC 2020. LNCS, vol. 12510, pp. 53–65. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64559-5_5
18. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167 (2015)
19. Jaakkola, T., Haussler, D.: Exploiting generative models in discriminative classifiers. In: Advances in Neural Information Processing Systems, pp. 487–493 (1999)
20. Jégou, H., Douze, M., Schmid, C., Pérez, P.: Aggregating local descriptors into a compact image representation. In: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 3304–3311. IEEE (2010)
21. Karimi, B., Wai, H.T., Moulines, É., Lavielle, M.: On the global convergence of (fast) incremental expectation maximization methods. In: Advances in Neural Information Processing Systems, pp. 2833–2843 (2019)
22. Khosla, A., Jayadevaprakash, N., Yao, B., Li, F.F.: Novel dataset for fine-grained image categorization: Stanford dogs. In: Proceedings of CVPR Workshop on Fine-Grained Visual Categorization (FGVC), vol. 2. Citeseer (2011)
23. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
24. Korsch, D., Bodesheim, P., Denzler, J.: Classification-specific parts for improving fine-grained visual categorization. In: Fink, G.A., Frintrop, S., Jiang, X. (eds.) DAGM GCPR 2019. LNCS, vol. 11824, pp. 62–75. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-33676-9_5
25. Krause, J., et al.: The unreasonable effectiveness of noisy data for fine-grained recognition. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9907, pp. 301–320. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46487-9_19
26. Liang, P., Klein, D.: Online EM for unsupervised models. In: Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, pp. 611–619 (2009)
27. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear CNN models for fine-grained visual recognition. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 1449–1457 (2015)
28. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101 (2017)
29. Neal, R.M., Hinton, G.E.: A view of the EM algorithm that justifies incremental, sparse, and other variants. In: Jordan, M.I. (ed.) Learning in Graphical Models. ASID, vol. 89, pp. 355–368. Springer, Heidelberg (1998). https://doi.org/10.1007/978-94-011-5014-9_12
30. Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–8. IEEE (2007)
31. Perronnin, F., Sánchez, J., Mensink, T.: Improving the fisher kernel for large-scale image classification. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 143–156. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_11
32. Russakovsky, O., et al.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vision **115**(3), 211–252 (2015). https://doi.org/10.1007/s11263-015-0816-y

33. Simon, M., Gao, Y., Darrell, T., Denzler, J., Rodner, E.: Generalized orderless pooling performs implicit salient matching. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4970–4979 (2017)
34. Simon, M., Rodner, E., Darell, T., Denzler, J.: The whole is more than its parts? From explicit to implicit pose normalization. IEEE Trans. Pattern Anal. Mach. Intell. **42**, 1–13 (2018)
35. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep fisher networks for large-scale image classification. In: Advances in Neural Information Processing Systems, pp. 163–171 (2013)
36. Song, Y., Zhang, F., Li, Q., Huang, H., O'Donnell, L.J., Cai, W.: Locally-transferred fisher vectors for texture classification. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 4912–4920 (2017)
37. Sydorov, V., Sakurada, M., Lampert, C.H.: Deep fisher kernels-end to end learning of the fisher kernel GMM parameters. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1402–1409 (2014)
38. Szegedy, C., et al.: Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1–9 (2015)
39. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2016)
40. Tang, P., Wang, X., Shi, B., Bai, X., Liu, W., Tu, Z.: Deep FisherNet for image classification. IEEE Trans. Neural Netw. Learn. Syst. **30**(7), 2244–2250 (2018)
41. Touvron, H., Vedaldi, A., Douze, M., Jégou, H.: Fixing the train-test resolution discrepancy. In: Advances in Neural Information Processing Systems, pp. 8250–8260 (2019)
42. Van Horn, G., et al.: Building a bird recognition app and large scale dataset with citizen scientists: the fine print in fine-grained dataset collection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 595–604 (2015)
43. Van Horn, G., et al.: The iNaturalist species classification and detection dataset. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 8769–8778 (2018)
44. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The CALTECH-UCSD birds-200-2011 dataset. Technical report CNS-TR-2011-001, California Institute of Technology (2011)
45. Wieschollek, P., Groh, F., Lensch, H.: Backpropagation training for fisher vectors within neural networks. arXiv preprint arXiv:1702.02549 (2017)
46. Yang, S., Liu, S., Yang, C., Wang, C.: Re-rank coarse classification with local region enhanced features for fine-grained image recognition. arXiv preprint arXiv:2102.09875 (2021)
47. Yang, Z., Luo, T., Wang, D., Hu, Z., Gao, J., Wang, L.: Learning to navigate for fine-grained classification. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y. (eds.) Computer Vision – ECCV 2018. LNCS, vol. 11218, pp. 438–454. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01264-9_26
48. Zhang, J., Zhang, R., Huang, Y., Zou, Q.: Unsupervised part mining for fine-grained image classification. arXiv preprint arXiv:1902.09941 (2019)
49. Zhang, L., Huang, S., Liu, W., Tao, D.: Learning a mixture of granularity-specific experts for fine-grained categorization. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 8331–8340 (2019)

50. Zhang, X., Xiong, H., Zhou, W., Lin, W., Tian, Q.: Picking deep filter responses for fine-grained image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1134–1142 (2016)
51. Zheng, H., Fu, J., Mei, T., Luo, J.: Learning multi-attention convolutional neural network for fine-grained image recognition. In: Proceedings of the IEEE International Conference on Computer Vision (2017)
52. Zheng, H., Fu, J., Zha, Z.J., Luo, J.: Learning deep bilinear transformation for fine-grained image representation. In: Advances in Neural Information Processing Systems, pp. 4279–4288 (2019)
53. Zhuang, P., Wang, Y., Qiao, Y.: Learning attentive pairwise interaction for fine-grained classification. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 13130–13137 (2020)