Set2Model Networks: Learning Discriminatively To Learn Generative Models

Alexander Vakhitov
a.vakhitov@skoltech.ru

Andrey Kuzmin
a.kuzmin@skolkovotech.ru

Victor Lempitsky
lempitsky@skoltech.ru

Skolkovo Institute of Science and Technology
Moscow, Russia

Abstract

We present a new "learning-to-learn"-type approach for small-to-medium sized training sets. At the core lies a deep architecture (a Set2Model network) that maps sets of examples to simple generative probabilistic models such as Gaussians or mixtures of Gaussians in the space of high-dimensional descriptors. The parameters of the embedding into the descriptor space are discriminatively trained in the end-to-end fashion. The main technical novelty of our approach is the derivation of the backprop process through the mixture model fitting. A trained Set2Model network facilitates learning in the cases when no negative examples are available, and whenever the concept being learned is polysemous or represented by noisy training sets. Among other experiments, we demonstrate that these properties allow Set2Model networks to pick visual concepts from the raw outputs of Internet image search engines better than a set of strong baselines.

1. Introduction

The ability to learn concepts from small training sets has emerged as an important frontier in AI. It is well known [29] that such ability is a hallmark of human intelligence, as humans demonstrate remarkable ability to learn complex visual concepts from only few representative images. Despite surpassing human intelligence in many narrow application domains, AI systems are not able to match this human ability.

Such ability has important practical applications. For example, to pick up a new visual concept, most modern computer vision systems require a set of images depicting this concept. Such an image set is usually mined from the World Wide Web using an Intenet image search engine or comes from a website containing tagged images. Most notable and influential in this respect is the Image-Net project [5] that has the goal of obtaining a "clean" image set for each of the 80,000+ visual *synsets* corresponding to English nouns. For each such noun, the image set is first obtained by querying the search engine and is then curated through crowdsourced human labour. The second step (screening by humans) represents a significant burden. As a result, while the positive influence of the Image-Net project on the fields of computer vision and artificial intelligence has been enormous, the progress towards its initial goal has stalled at about one quarter (at the time of submission 21,841 synsets out of 80,000 have been indexed).

The use of *uncurated* image sets from Internet search engines can potentially enable computers to learn visual concepts automatically and without humans in the loop. Such capability is highly beneficial for intelligent systems, especially in certain scenarios, such as *open-vocabulary* image retrieval that allows users to formulate queries to image collections using arbitrary natural language queries. The use of uncurated image sets obtained from the web, however, is known to be challenging [8, 15], since despite the ever-improving performance of image search engines, the returned image sets still contain irrelevant images, since many natural language queries are inherently polysemous, and since many visual concepts often correspond to different visual aspects (e.g. outdoor and indoor views of a certain landmark building).

Here, we introduce a new meta-learning approach that is based around a new deep learning architecture called *Set2Model (S2M) network*. An S2M network can be trained to learn new concepts from small-to-medium sized training sets. To map a set of examples to a generative model, a pre-trained S2M network first embeds them into a high-dimensional space, and then fits a generative model in that space to the outputs of the mapping. The training process for the S2M model considers a large set of modeling tasks and corresponds to the tuning of the parameters of the non-linear embedding in order to facilitate easy generative learn-

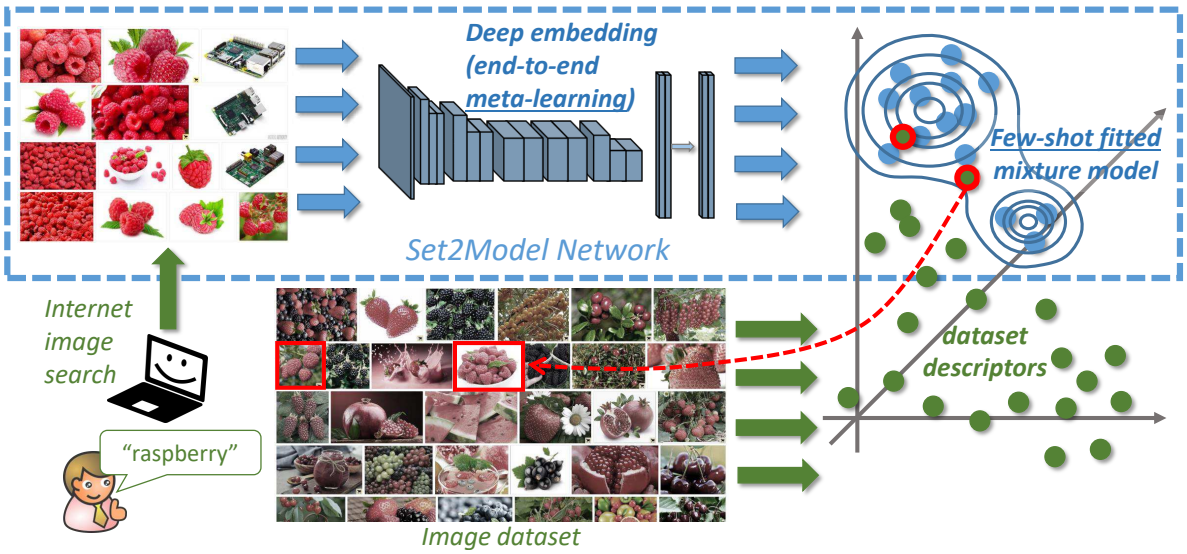

Figure 1. **Top (blue)**: the Set2Model (S2M) network, which takes the set of data points (e.g. images), maps them through a non-linear architecture (e.g. a deep ConvNet) to a high-dimensional descriptor space, and then fits a generative model (e.g. Gaussian mixture) to the resulting set of descriptors. The parameters of the deep embedding are optimized in the end-to-end meta-learning stage, while the generative model is fitted in the few-shot learning stage. **Bottom (green)**: our motivating application (Internet-based learning and retrieval). Given a visual concept "raspberry", the user obtains a noisy image set depicting raspberries from an Internet image search engine. A pre-meta-learned Set2Model network then maps the set to a mixture model in the descriptor space. Given an unannotated dataset of images, the user can search for images with raspberries by mapping every image to the descriptor space (using the same deep embedding from the S2M network) and evaluating the likelihood w.r.t. the obtained model.

ing for the embedded data.

Unlike analogous meta-learning approaches [28, 31, 32] that learn to learn *discriminative* classifiers, a trained Set2Model outputs *generative* models (in this work, we investigate Gaussians and Gaussian mixtures with diagonal covariance matrices). Consequently, a trained Set2Model can learn concepts from only positive examples, which is a more natural setup in many scenarios where negative/background class can be much more diverse than the positive class. Through the use of Gaussian mixtures, S2M networks can also efficiently handle polysemous concepts as well as outliers in the provided training sets. This makes Set2Model networks suitable for picking up concepts from uncurated Internet search engine outputs.

Below, we briefly discuss relevant prior works in Section 2, detail our approach in Section 3, present results of experimental comparisons in Section 4 and conclude with a short discussion in Section 5.

2. Related work

Meta-learning and Few-shot Learning Meta-learning (aka "learning-to-learn") [6, 26] has been a popular approach to handle multi-shot learning scenarios. Interplays between discriminative training and generative probabilistic models in the small training size regime has been investigated in [23, 9, 18], where some of the parameters of such models were optimized based on discriminative criteria. Minka [21] has pointed out a principled way to derive discriminatively-trainable models.

Learning concepts from small training sets (sometimes referred to as "few-shot learning") has been a subject of intense recent research in meta-learning. The common idea is to learn internal representation, where few-shot learning is simple by observing a large number of few-shot learning tasks. The previous approaches [28, 25, 32, 31] invariably focused on learning multi-class discriminative classifiers for such few-shot learning problems. Our approach however focuses on single-concept learning problems, which are presented in the form of positive one-class samples. We separate discriminative and generative learning across the two layers of our meta-learning system. In particular, once the embedding within a S2M network is trained discriminatively, a generative learning process in the descriptor space (which is fixed after the S2M training) is used to fit a generative model over those embeddings.

The approach [20] learns a metric in the image feature domain in order to improve distance-based image classification and shows that the resulting metric generalizes well to the classes unseen during training. It also proposes a Nearest Class Mean (NCM) classifier as a distance to a mean of image class descriptors which we use as one of the baseline methods.

Historically, Internet image search relied mostly on tex-

tual information surrounding the image using image content to improve ranking [7, 27].

[1] addressed large-scale image retrieval problem using image sets as queries and SIFT-based bag-of-words vectors as image features, considering several variants including binary SVM learned on the query set with randomly sampled negatives, ranking using averaged query feature vectors and averaging of the rankings for each individual query from the query set. The works [4, 3] developed these methods further proposing cascades of classifiers for real-time on-the-fly object category retrieval in large image and video datasets using various features including deep features in [3].

Our approach is also closely related to certain directions pursued in the information retrieval and multimedia search communities including a large body of query reranking approaches, some of which use discriminative learning [10, 33].

3. Set2Model Networks

Set2Model networks imply two levels of learning following the "learning-to-learn" principle (e.g., [25]). It implies that a system has two timescales, and the rapid one is associated with learning to solve a task, while the gradual one aims at acquisition of knowledge across tasks. In case of the Set2Model network, on the rapid timescale the network maps a number of samples to a generative model, while on the gradual timescale the parameters of the network are tuned based on a large number of sample class-modeling problems. We call the training process happening at the gradual timescale the *meta-learning* stage. The application of the network to a particular problem is called the *learning* stage. Below, we discuss the details of these stages, first starting with the learning stage, and then discussing the meta-learning stage.

3.1. Learning models with pre-trained S2M networks

The learning stage (Figure 1,top) considers the set of examples $X = \{x^1, x^2, \dots, x^N\}$, $X \subset \mathcal{X}$, such as a set of images depicting a certain concept. In the learning stage, the S2M network maps X to a probabilistic model that can be used to evaluate probabilities of other elements belonging to the same concept.

Modeling the probability distribution in the original space (e.g. images) might be overly complex. Therefore, the S2M network firstly maps the elements of X to a specially constructed latent space of descriptors and then uses a simple parametric probability density function (pdf) to model it in the new space. We denote this mapping as $f(x; w) : \mathcal{X} \mapsto \mathbb{R}^n$, whereas w denotes the parameters of the mapping. In this work, we focus on deep convolutional networks as such mappings, though our approach is not specific to a particular architecture.

The mapping f thus transforms the original set X into a descriptor set $D = \{d^1, d^2, \dots, d^N\}$, where $d^i = f(x^i; w)$. The last stage of the learning process fits a parametric generative model with the pdf $p_{GM}(d; \theta)$ to the set D, where θ are the model parameters. In this work, we consider Gaussian and mixture of Gaussian models with diagonal covariance matrices. We chose a generative approach to descriptor set modelling because it led to better precision while using the pre-learned deep features in our experiments (see Table 1, 'Gauss-PL' vs 'SVM-PL').

The fitting of the model to the set D is performed using maximum likelihood (ML). Thus the parameters θ^* that maximize the likelihood function $l(\theta|D)$ are sought:

$$\theta^* = \arg\max_\theta l(\theta|D), \;\; l(\theta|D) = \sum_{i=1}^{N} \log p_{GM}(d^i, \theta). \tag{1}$$

Overall, the learning performed by an S2M network can be regarded as the mapping:

$$F : X \to \theta^*. \tag{2}$$

And the relevance of a new data point (e.g. an image) z to the concept represented by the set X can be estimated using the obtained density function:

$$p(z|X; w) = p_{GM}(f(z; w), F(X; w)). \tag{3}$$

The resulting probabilistic relevance measure can be used e.g. to perform retrieval from an untagged set (Figure 1,bottom).

3.2. Meta-Learning S2M Networks

The goal of the meta-learning is to find the parameters w of the mapping $f(x, w)$ such that the learning process discussed above works well for different concepts.

The meta-learning is performed using the set of tuples $\{\mathcal{T}_i = (X_i, Z_{+,i}, Z_{-,i})\}$, where each tuple $\mathcal{T}_i$ includes the *concept-describing set* $X_i = \{x_i^1, x_i^2, \dots, x_i^{N_i}\}$, the *relevant examples set* $Z_{+,i} = \{z_{+,i}^1, z_{+,i}^2, \dots, z_{+,i}^{M_{+,i}}\}$ and the *irrelevant examples set* $Z_{-,i} = \{z_{-,i}^1, z_{-,i}^2, \dots, z_{-,i}^{M_{-,i}}\}$. For example, X_1 can be some set of images of apples from the Internet, the set of $Z_{+,1}$ can be another set of different images also containing apples, and all the images from $Z_{-,1}$ will not contain apples. The second training tuple can then include the sets X_2 and $Z_{+,2}$ of pear images and the set of $Z_{-,2}$ of non-pear images, and so on.

Generally, the meta-learning stage seeks the parameters w such that across all training tuples, the probabilistic relevances estimated using (3) are higher for the members of the relevant sets than for the members of the irrelevant sets, i.e.:

$$p(z_{+,i}^k|X_i; w) > p(z_{-,i}^l|X_i; w), \tag{4}$$

for various i, k, l.

There are several ways to design loss functions that seek to enforce (4). E.g. a loss that draws random elements of relevant and irrelevant sets and computes a monotonic function of the differences in their relevance values. Here, we use a tuple-level loss that directly estimates the probability of the (4) to be violated. Given a tuple $(X_i, Z_{+,i}, Z_{-,i})$, let θ_i^* be $F(X_i; w)$. Using (3), we compute the relevance scores for the elements of the relevant and irrelevant sets.

Let $R_{+,i}$ be the set of relevances for the relevant set $Z_{i,+}$ (i.e. $R_{+,i} = \{p(z^1_{+,i}|X_i; w), p(z^2_{+,i}|X_i; w) \dots p(z^{M_{+,i}}_{+,i}|X_i; w)\}$), and let $R_{-,i}$ be the set of relevances for the irrelevant set $Z_{i,-}$. Then the loss based on the probability of the violation of (4) can be computed as:

$$L(w) = \sum_i \frac{1}{M_{+,i} M_{-,i}} \sum_{k=1}^{M_{+,i}} \sum_{l=1}^{M_{-,i}} \chi \left[R^k_{+,i} < R^l_{-,i} \right] , \quad (5)$$

where $\chi[\cdot]$ returns one if the argument is true and zero otherwise. Here, each term in the outer summation corresponds to the empirical estimate of the probability of violation of (4).

While the loss (5) is not piecewise-differentiable, we consider a histogram trick recently suggested in [30] for metric learning. The idea is to accumulate the relevances for the relevant and irrelevant sets into histograms, and then estimate the required probability (5) using these histograms. Here, to compute the histograms, we fix the triangular kernel density estimator $K(s, \omega)$ that for the argument s and the width parameter ω is defined as:

$$K(s, \omega) = \max\{1 - \frac{2|s|}{\omega}, 0\}. \quad (6)$$

We choose $l_{\min,i}$ and $l_{\max,i}$ to be the lower and the upper bounds of the numbers in the union of $R_{+,i}$ and $R_{-,i}$, and further accumulate the two normalized histograms $h_{+,i}$ and $h_{-,i}$ spanning the range from $l_{\min,i}$ to $l_{\max,i}$ having B bins each and corresponding to the sets $R_{+,i}$ and $R_{-,i}$ respectively. As discussed in [30], the entries of the histograms $h_{+,i}$ and $h_{-,i}$ depend in a piecewise-differentiable manner on the entries of $R_{+,i}$ and $R_{-,i}$.

Given the two histograms, the loss for the tuple $(X_i, Z_{+,i}, Z_{-,i})$ is defined as:

$$L(w) = \sum_i \sum_{k=1}^{B} h^k_{-,i} \sum_{l=1}^{B} h^l_{+,i} , \quad (7)$$

where $h^k_{-,i}$ and $h^l_{+,j}$ denote the entries of the histograms. Note that the new loss (7) can be regarded as piecewise-differentiable approximation to the non-differentiable loss (5).

Given the loss (7) (or any other piecewise-differentiable loss enforcing (4)), the meta-learning process follows the standard stochastic optimization procedure. The training tuples are sampled randomly, the stochastic approximations of the loss (7) based on single tuples are computed by forward propagation. During forward propagation, the maximum likelihood fitting (1) is done by conventional means (e.g. closed form for Gaussian distribution, EM-algorithm for Gaussian mixture model). The estimated loss is then backpropagated through the S2M network. Any of the SGD-based optimization algorithms such as ADAM [12] can be used to update the mapping weights w. Backpropagation through the S2M network $F(X, w)$ however relies on the ability to backprop through the maximum-likelihood model fitting (1). This backprop step is discussed below.

3.3. Backpropagation through model fitting

We now detail the backpropagation through the ML model fitting (1), i.e. the computation of the partial derivatives $\frac{\partial \theta^*}{\partial d^i_{(j)}}$, where $\cdot_{(j)}$ denotes the j-th component of a vector. We start with the Gaussian model, for which this computation is based on a simple closed-form expression, and then proceed to the case of Gaussian mixtures.

In the first case, we consider the Gaussian pdf $p_G(d, \theta) = \mathcal{N}(d, \mu, \Sigma)$:

$$\mathcal{N}(d, \mu, \Sigma) = (2\pi)^{-n/2} |\Sigma|^{-1/2} e^{-1/2 (d-\mu)^T \Sigma^{-1} (d-\mu)}, \quad (8)$$

where μ is a mean and Σ is a covariance matrix which we take to be diagonal, and $\theta = \left(\mu^T, \phi^T\right)^T$, denoting $\Sigma = \operatorname{diag} \phi$. The optimal (in the ML sense) parameters θ^* can then be found as:

$$\mu^* = \frac{1}{N} d^i, \quad \phi^*_{(i)} = \frac{1}{N} \sum_j \left(d^j_{(i)} - \mu^*_{(i)} \right)^2 , \quad (9)$$

Differentiation of these formulas w.r.t. the descriptor vectors d^i leads to the following:

$$\nabla_{d^i} \mu^* = \frac{1}{N} \mathbf{1}^n, \quad \frac{\partial \phi^*_{(i)}}{\partial d^j_{(k)}} = \delta_{ik} \frac{2}{N} (d^j_{(k)} - \mu^*_{(i)}), \quad (10)$$

where $\mathbf{1}^n$ is a vector of ones of dimension n, δ_{ik} is a Kronecker symbol, which is zero when $i \neq k$ and one for $i = k$.

In the case of Gaussian mixtures (GMM), we consider the following pdf:

$$p_{GMM}(d, \theta) = \sum_{i=1}^{k} v_i \mathcal{N}(d, \mu_i, \Sigma_i), \quad (11)$$

where k is a number of GMM components, $\mathcal{N}$ denotes Gaussian pdfs, $\{\mu_i\}_{i=1}^k$ are the means, $\Sigma_i = \operatorname{diag}(\phi_i)$ are the diagonal covariance matrices, $\{v_i\}_{i=1}^k$ are weights of

the corresponding components and θ consists of the means, covariance diagonals and weights concatenated. The following constraint on the weights should be satisfied:

$$c(\theta) = \sum_{i=1}^{k} v_i - 1 = 0. \tag{12}$$

As θ^* delivers maximum to an optimization problem with equality constraints, the method of Lagrange multipliers can be used to derive conditions on the partial derivatives. We consider the scalar multiplier λ^* corresponding to the constraint (12) that maximizes the Lagrangian $\mathcal{L}(\theta, \lambda)$:

$$\mathcal{L}(\theta, \lambda) = l(\theta|D) + \lambda c(\theta), \quad (\theta^*, \lambda^*) = \arg\max \mathcal{L}(\theta, \lambda), \tag{13}$$

The following conditions of optimality then holds for θ^* and λ^*:

$$\begin{cases} \nabla_\theta \mathcal{L}(\theta^*, \lambda^*) &= 0, \\ \nabla_\lambda \mathcal{L}(\theta^*, \lambda^*) &= 0. \end{cases} \tag{14}$$

Applying differentiation over d^i brings the following system of equations:

$$\begin{cases} \frac{\partial^2}{\partial \theta^2}\mathcal{L}\nabla^T\theta^* + \frac{\partial^2}{\partial \lambda \partial \theta}\mathcal{L}\nabla^T\lambda^* + \frac{\partial^2}{\partial d^i \partial \theta}\mathcal{L} &= 0, \\ \frac{\partial^2}{\partial \theta \partial \lambda}\mathcal{L}\nabla^T\theta^* + \frac{\partial^2}{\partial \lambda^2}\mathcal{L}\nabla^T\lambda^* + \frac{\partial^2}{\partial d^i \partial \lambda}\mathcal{L} &= 0, \end{cases} \tag{15}$$

and these are linear equations w.r.t. unknown matrix $\nabla\theta^*$ and vector $\nabla\lambda^*$ of partial derivatives w.r.t. $d^i_{(j)}$ (in particular, $\nabla\theta^*$ is composed of the values $\frac{\partial\theta^*}{\partial d^i_{(j)}}$ that are sought in this derivation). If m is the dimensionality of θ, then (14) contains $m+1$ equations. As each of the equation is differentiated by nN variables corresponding to the descriptors, the system (15) contains $(m+1)nN$ equations on the same number of entries in $\nabla\theta^*$ and $\nabla\lambda^*$. Solving the system (15) then yields the values of the partial derivatives $\frac{\partial\theta^*}{\partial d^i_{(j)}}$. The linear system solution is performed as the part of the backpropagation process. The system very often becomes sparse allowing for the significant solution process speedup, see appendix A1.

Finally, we note that a similar derivation can be conducted for other generative probabilistic models. Furthermore, one could address discriminative model fitting (e.g. logistic regression) in the same setting as in [28, 25, 32, 31], where the set X is augmented with class labels.

4. Experiments

Below we provide the experimental evaluation of Set2Model networks (both using single Gaussians and Gaussian mixtures as generative models). We investigate the importance of learning the underlying features. We show that Set2Model networks can be used as generative set models and compare the performance of the Set2Model networks to a number of baselines. Also we apply the Set2Model networks in a few-shot learning problem mainly for the sake of comparison to other meta-learning approaches.

4.1. Protocols

We evaluate the S2M networks in three different sets of experiments. The bulk of the experiments investigates image retrieval with concept-describing image sets generated using Internet image search engines (web-initialized retrieval). Then, we evaluate the possibility of handwritten character retrieval using the Omniglot dataset. Finally, we show that S2M networks can be used for classification of the characters from Omniglot. In particular, to solve the 5-way or 20-way classification problem we build a generative model for each class and compare the ranks w.r.t. these models during test time. The latter experiment shows that the S2M networks can achieve few-shot learning accuracy comparable to the state of the art discriminative approaches although they are not specifically tailored for this task. The results of retrieval experiments are in the Table 1, and the character classification results are in the Table 2.

In each case, we split classes into training, validation, and testing. We form the training set out of the training classes, and we train the methods (including ours) on such classes. The methods are compared on the test set. During test, we use the mean average precision (mAP) as the accuracy metric for the retrieval. The meta parameters for all methods are tuned on the validation set.

We perform web-initialized experiments with three different datasets. Google image search is used to obtain the concept-describing sets at all stages (the API for the search engine typically returns 90-100 images). We use class names provided with the datasets to define text queries to the engine. No textual augmentations or search modifiers have been used. Since some of the considered datasets have overlaps with the search engine output, before running the experiments, we identified potential near-duplicates between the Google search results and the datasets using deep descriptors from non-finetuned convolutional network (AlexNet). We then manually checked the potential pairs and removed the true near duplicates from consideration.

4.2. Implementation details

We use Caffe [11] framework to work with convolutional networks. Web-initialized retrieval experiments are based on the AlexNet architecture [14] and character retrieval ones are based on the LeNet architecture [19] (Caffe versions are used in all cases). We note that more modern deep convolutional architectures could be used in place of AlexNet or LeNet, however such substitution is likely to benefit all methods equally. For the character classification, we use our Caffe implementation of the network described

Figure 2. Retrieval comparison for the ImageNet dataset (query = 'crucian carp,Carassius carassius,Carassius vulgaris,') - left and the Oxford Flowers dataset (query = 'silverbush') right. Top row: part of the query image set obtained from Google Search API using the query. Other rows: the top ranked images provided by various methods, namely AVG-FT (average), NN-FT (nearest neighbor), S2M-Gauss (one Gaussian), S2M-GMM2 (mixture of two Gaussians), S2M-GMM3 (mixture of three Gaussians) and S2M-GMM4 (mixture of four Gaussians) using the fine-tuned descriptors, SVM-PL using the pre-learned 'fc8' features. Color bars encode that image belongs to a certain mixture component. End-to-end-based methods perform better on the given examples. Mixture models successfully filter noisy search engine outputs and capture multiple visual aspects of relevant images.

in [31]. It consists of a stack of modules, and each module is a sequence of 3×3 convolution with 64 filters, batch normalization, Relu non-linearity and 2×2 max-pooling. It takes 28×28 images as input and produces 64-dimensional features.

When performing baseline experiments with pre-learned features, we use 1000-dimensional features that are produced by AlexNet (for the tasks we consider, these features performed optimally or close to optimally compared to other layers). When performing end-to-end learning in case of web-initialized or character retrieval experiments, we replace the last fully connected layer with a smaller one of the same type, of size 128 or 100 respectively.

We perform l_2-normalization of the descriptors at the end of the network. In the web-initialized experiments, we start learning from the network weights of the AlexNet provided with Caffe, while the last fully-connected layer for the end-to-end trained architectures is initialized randomly.

The set modelling layer and the loss layer have been implemented using Theano [2], which was used mainly for symbolic differentiation. For back-propagation and learning we use the Caffe implementation of the ADAM algorithm [12] with momentum 0.9. We choose the learning rate and the termination moment using validation sets. To solve the linear system (15), we utilize its sparsity, since in most cases the coefficient matrices consist of diagonal blocks.

Training of the S2M networks includes an EM algorithm for fitting the GMMs. When a class was encountered for the first time during training, GMM fitting was started with random initialization, and the resulting GMM parameters were memorized. Next time when samples from this class appeared, the saved parameters served as an initial point for the EM algorithm.

4.3. Baselines

Below we describe a set of baseline algorithms. For each of these baselines, we use a certain (not necessarily probabilistic) relevance score $r(z|X;w)$ in the same way as we use $p(z|X;w)$ in the S2M network. The following baselines are considered:

- The mean-based system (**AVG**), which ranks the images in the test set based on the scalar product between their descriptors and the mean of the descriptors $f(x^i, w)$ of the concept-describing set:

$$r_{AVG}(z|X;w) = \left(\frac{1}{N}\sum f(x^i, w)\right)^T z. \qquad (16)$$

- The nearest neighbor (**NN**) ranker that ranks images in the test set based on the maximum of their scalar product with the query set descriptors:

$$r_{NN}(z|X;w) = \max_i \left(f(x^i, w)\right)^T z. \qquad (17)$$

Model	ImageNet	RGBD Object	Oxford Flowers	Omniglot
NN-PL	0.070	0.316	0.076	-
NN-FT	0.073	0.318	0.528	0.145
SVM-PL	0.080	0.481	0.145	-
AVG-PL	0.183	0.556	0.212	-
AVG-FT	0.236	0.669	0.439	0.661
Gauss-PL	0.186	0.529	0.240	-
S2M-Gauss	0.254	0.706	0.467	**0.740**
Gauss-AVG-FT	0.246	0.676	0.465	0.695
GMM2-PL	0.180	0.483	0.301	-
S2M-GMM2	**0.265**	**0.711**	0.560	0.689
GMM3-PL	0.174	0.473	0.319	-
S2M-GMM3	0.258	0.658	**0.581**	-
GMM4-PL	0.178	0.455	0.327	-
S2M-GMM4	0.250	0.690	0.577	-

Table 1. Mean average precisions for the experiments (see text for discussion). Top row: dataset. The baselines either use pre-learned ('-PL') or fine-tuned ('-FT') deep features. Gauss-Avg-FT baseline uses features fine-tuned using an AVG baseline (same as AVG-FT), but a Gaussian model on top. Methods with the 'S2M' prefix are the proposed ones. We do not perform any fine-tuning for the SVM, and we do not use GMM with 3 or 4 components with Omniglot due to small size of the concept-describing sets that contain only 10 images. The best achieved results are bolded. The Set2Model networks (S2M-) outperform baselines. End-to-end finetuning improves the results considerably for all methods ('-PL' vs '-FT'). Also, using correct end-to-end learning for a single Gaussian (S2M-Gauss) performs better than using end-to-end learning for mean-based retrieval, while using Gaussian models fitted to resulting features during retrieval (Gauss-AVG-FT).

Model	5-way	20-way
Matching Nets [31]	**0.989**	**0.985**
MANN (No Conv) [25]	0.949	-
Convolutional Siamese Net [13]	0.984	0.965
S2M-Gauss	0.985	0.956

Table 2. Results for the 5-shot 5-way or 20-way classification on the Omniglot dataset. We performed meta-learning of S2M-Gauss based on the same underlying deep network as described in [31] using the protocol described in Section 3. During classification, we choose the class label corresponding to the maximal rank produced by the S2M network for 5 (or 20) considered classes. The testing protocol follows [31].

- The support vector machine (**SVM**) 1-vs-all classifier as in [1], where SVM is learned using the query images as positive class and $2|X|$ randomly sampled images from other queries as negative class. If we denote the weight vector of the SVM as $u(\{f(x^i, w)\}_{i=1}^N)$, then the ranking function can be defined as:

$$r_{SVM}(z|X; w) = z^T u(\{f(x^i, w)\}_{i=1}^N). \quad (18)$$

We evaluate these baseline methods as well as single Gaussian and GMM models for the pre-learned descriptors('**-PL**' in Table 1). We also consider fine-tuning of the convolutional network for the mean (AVG) and the nearest neighbor (NN) ranking ('**-FT**' in Table 1). In this case, we use exactly the same learning architecture as explained in the previous section, but plug the corresponding relevance measure $r_{NN}(Z|X; w)$ or $r_{AVG}(z|X; w)$ instead of (3) into the computation of the histogram loss.

Finally, our strongest baseline (**Gauss-AVG-FT** in Table 1) is an ablated S2M network that is fine-tuned for the mean-based retrieval, but uses Gaussian model fitting during retrieval in the same way as our model based on single Gaussian does. Alongside the baselines, we report the results of our system (S2M network) for different number of Gaussians in the mixtures while the same number of mixture components is used during retrieval and during meta-learning ('**S2M-Gauss**', '**S2M-GMM**m', $m = 2, 3, 4$).

4.4. Results

The quantitative results for all datasets are summarized in Tables 1, 2. We also illustrate retrieval performance of some of the compared methods at the Figure 2. We indicate whether the image truly belongs to the query class by showing a corresponding symbol in the bottom-right corner of the image, output images also have a colored bar encoding a particular mixture component 'responsible' for this image.

ImageNet Our first experiment uses classes from the ImageNet dataset [5]. Since we use networks pretrained on the ILSVRC classes [24], we made sure that 1000 classes included into the ILSVRC set are excluded from our experiments.

To perform the experiments, we selected 509 random synsets for training, 99 synsets for validation and 91 synset for testing. The results (Table 1) demonstrate that end-to-end learning is able to improve the mAP of the baseline methods by 1-4 percent and of the proposed methods based on distribution fitting by 5-7 percent. Importantly, the gap between the model that fits a single Gaussian and the model that uses the mean vector is almost 2 percent. Using mixture models with two or three components improves the performance further.

We also observe, that uncurated Google image search outputs for some of the ImageNet synsets are very noisy, S2M networks often group all relevant query images into a single Gaussian component (Figure 2-right). This is often a desirable performance, since being able to absorb irrelevant aspects of the query into a separate component may allow to learn a better model for the relevant aspect. At the same time, when multiple aspects are relevant, multiple mixture components are often able to retrieve them as well (Figure 2-left and Figure 3).

RGBD This experiment uses the RGBD-Object [16] dataset. RGBD-Object contains multiple view images of

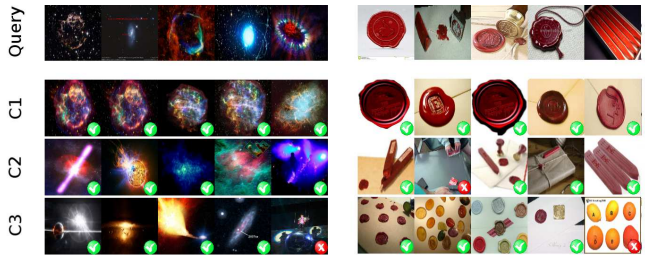

Figure 3. Set2Model networks can handle polysemous queries in a natural way. Top: part of the query image set obtained from Google Search API (queries='supernova' and 'seal, sealing wax'), bottom: relevant images retrieved by the three components of the GMM3 model returned by the corresponding S2M network. In these cases, each mixture component captures a certain aspect of a visual concept.

200 tabletop objects of 51 category .

The amount of relevant images in the Internet search results in this case differs greatly from category to category. For the polysemous categories such as 'apple', the proposed generative models can give significant benefit The results in Table 1 demonstrate even greater improvements from the end-to-end learning (perhaps due to a bigger domain gap between the ImageNet and this dataset). Such training improves the mAP for the mean classifier by nine percent and for the proposed methods by 11-14 percent. The methods based on distribution fitting again perform better.

Oxford Flowers We use the Oxford Flowers-102 dataset [22], consisting of images of 102 different UK flowers. The dataset was split into 80 categories for training and validation, and 22 for testing. The results in the Table 1 show that end-to-end learning procedure improves the mAP on 14 percent for the mean classifier, 16-19 percent for the proposed models. Figure 2 shows an example of the polysemous and noisy query ('silverbush'), where the ability of the Gaussian mixture models to capture multi-modal distributions provides our approach a big advantage.

Omniglot Finally, we used the Omniglot dataset [17] (which has become the standard testbed for meta-learning methods) to test the ability of the S2M network to use small concept-describing sets for generative model construction. The Omniglot dataset consists of 20 hand-drawn images for each of the 1623 characters from different alphabets. In the retrieval experiment, for learning and testing we use concept-describing, relevant and irrelevant sets of ten images, composing one batch of five training tuples $(X_i, Z_{+,i}, Z_{-,i})$. We randomly split the dataset into 1200 classes for validation and training, and 423 for testing. We rotate images by randomly generated multiples of 90 degrees during testing and training, following [31]. Due to the small size of the concept-describing sets, S2M networks based on mixtures of two Gaussians are less accurate, and mixtures of more components were not evaluated. The results in Table 1 show that single Gaussian model performs best .

Furthermore, to compare against the recent few-shot learning methods we performed a classification experiment on this dataset. During meta-learning, we used a batch consisting of three training tuples with $|X_i| = 5$, $|Z_{+,i}| = 15$, $|Z_{-,i}| = 20$. During test time, to do c-way classification we build c models using the S2M network and choose a class label corresponding to the model producing a maximal rank for the test example. We compare the results to the state of the art methods at the Table 2. Although the S2M network is not trained to discriminate between classes, still it can be used this way during test time and provide competitive results: better than [25], similar to [13] and not much worse than [31].

5. Summary

In this work we have proposed Set2Model networks as a new architecture for meta-learning that is particularly suitable for retrieval applications, where queries are given as sets of positive samples. The Set2Model networks are able to map such queries into probabilistic models in specially-designed descriptor spaces. The parameters of such descriptor embeddings are optimized end-to-end, while taking the model fitting into account. We have shown experimentally that such proper end-to-end training is beneficial for the retrieval quality.

In order to gain the ability to handle mixture model fitting within our approach, we have derived a way for back-propagation through the maximum likelihood model fitting. We have presented a number of experiments for image retrieval based on noisy image sets obtained from the Internet image search engines as well as for the hand-drawn character retrieval that show the ability of the S2M networks to generalize across classes and handle the challenges of visual concept modeling from small and medium-sized training sets better than baseline models. We have also shown that generatively trained S2M networks can achieve similar accuracy to the state of the art in few shot learning problems.

References

[1] R. Arandjelovic and A. Zisserman. Multiple queries for large scale specific object retrieval. In *BMVC*, pages 1–11, 2012. 3, 7

[2] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio. Theano: A CPU and GPU math compiler in python. In *Proc. 9th Python in Science Conf*, pages 1–7, 2010. 6

[3] K. Chatfield, R. Arandjelović, O. Parkhi, and A. Zisserman. On-the-fly learning for visual search of large-scale image and video datasets. *International Journal of Multimedia Information Retrieval*, 4(2):75–93, 2015. 3

[4] K. Chatfield and A. Zisserman. Visor: Towards on-the-fly large-scale object category retrieval. In *Asian Conference on Computer Vision*, pages 432–446. Springer, 2012. 3

[5] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, IEEE Conference on*, pages 248–255. IEEE, 2009. 1, 7

[6] H. C. Ellis. *The transfer of learning*. Macmillan, 1965. 2

[7] R. Fergus, P. Perona, and A. Zisserman. A visual category filter for Google images. In *European Conference on Computer Vision*, pages 242–256. Springer, 2004. 3

[8] S. Guadarrama, E. Rodner, K. Saenko, N. Zhang, R. Farrell, J. Donahue, and T. Darrell. Open-vocabulary object retrieval. In *Robotics: science and systems*, volume 2, page 6, 2014. 1

[9] A. Holub and P. Perona. A discriminative framework for modelling object classes. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 664–671, 2005. 2

[10] V. Jain and M. Varma. Learning to re-rank: query-dependent image re-ranking using click data. In *Proceedings of the 20th international conference on World wide web*, pages 277–286. ACM, 2011. 3

[11] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM International Conference on Multimedia*, pages 675–678. ACM, 2014. 5

[12] D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4, 6

[13] G. Koch, R. Zemel, and R. Salakhutdinov. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015. 7, 8

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 5

[15] N. Kumar and S. Seitz. Photo recall: Using the Internet to label your photos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 771–778, 2014. 1

[16] K. Lai, L. Bo, X. Ren, and D. Fox. A large-scale hierarchical multi-view RGB-D object dataset. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 1817–1824. IEEE, 2011. 7

[17] B. M. Lake, R. Salakhutdinov, J. Gross, and J. B. Tenenbaum. One shot learning of simple visual concepts. In *CogSci*, volume 172, page 2, 2011. 8

[18] J. A. Lasserre, C. M. Bishop, and T. P. Minka. Principled hybrids of generative and discriminative models. In *Computer Vision and Pattern Recognition, IEEE Conference on*, volume 1, pages 87–94. IEEE, 2006. 2

[19] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 5

[20] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Distance-based image classification: Generalizing to new classes at near-zero cost. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(11):2624–2637, 2013. 2

[21] T. Minka. Discriminative models, not discriminative training. Technical report, Technical Report MSR-TR-2005-144, Microsoft Research, 2005. 2

[22] M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Computer Vision, Graphics & Image Processing, Sixth Indian Conference on*, pages 722–729. IEEE, 2008. 8

[23] R. Raina, Y. Shen, A. Y. Ng, and A. McCallum. Classification with hybrid generative/discriminative models. In *NIPS*, volume 16, pages 545–552, 2004. 2

[24] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. 7

[25] A. Santoro, S. Bartunov, M. Botvinick, D. Wierstra, and T. Lillicrap. Meta-learning with memory-augmented neural networks. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 1842–1850, 2016. 2, 3, 5, 7, 8

[26] J. Schmidhuber, J. Zhao, and M. Wiering. Simple principles of metalearning. Technical report, Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale, 1996. 2

[27] F. Schroff, A. Criminisi, and A. Zisserman. Harvesting image databases from the web. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(4):754–766, 2011. 3

[28] K. D. Tang, M. F. Tappen, R. Sukthankar, and C. H. Lampert. Optimizing one-shot recognition with micro-set learning. In *Computer Vision and Pattern Recognition (CVPR), IEEE Conference on*, pages 3027–3034. IEEE, 2010. 2, 5

[29] J. B. Tenenbaum, C. Kemp, T. L. Griffiths, and N. D. Goodman. How to grow a mind: Statistics, structure, and abstraction. *Science*, 331(6022):1279–1285, 2011. 1

[30] E. Ustinova and V. Lempitsky. Learning deep embeddings with histogram loss. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4170–4178, 2016. 4

[31] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. In *Advances in Neural Information Processing Systems*, pages 3630–3638, 2016. 2, 5, 6, 7, 8

[32] Y.-X. Wang and M. Hebert. Learning to learn: Model regression networks for easy small sample learning. In *European Conference on Computer Vision*, pages 616–634. Springer, 2016. 2, 5

[33] L. Yang and A. Hanjalic. Supervised reranking for web image search. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 183–192. ACM, 2010. 3