DeepSetNet: Predicting Sets with Deep Neural Networks

Seyed Hamid Rezatofighi Vijay Kumar B.G Anton Milan Ehsan Abbasnejad Anthony Dick Ian Reid
School of Computer Science, The University of Adelaide, Australia
hamid.rezatofighi@adelaide.edu.au

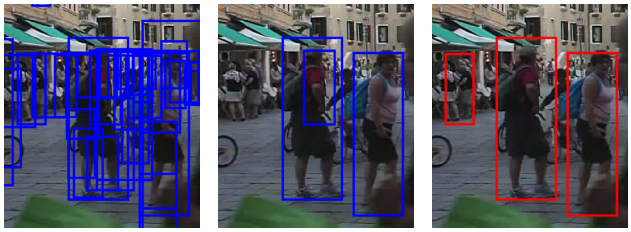
Abstract

This paper addresses the task of set *prediction using deep learning. This is important because the output of many computer vision tasks, including image tagging and object detection, are naturally expressed as sets of entities rather than vectors. As opposed to a vector, the size of a set is not fixed in advance, and it is invariant to the ordering of entities within it. We define a likelihood for a set distribution and learn its parameters using a deep neural network. We also derive a loss for predicting a discrete distribution corresponding to set cardinality. Set prediction is demonstrated on the problems of multi-class image classification and pedestrian detection. Our approach yields state-of-the-art results in both cases on standard datasets.*

1. Introduction

Deep neural networks have state-of-the-art performance for many computer vision problems, including semantic segmentation [25], visual tracking [24], image captioning [16], scene classification [17], and object detection [20]. However, traditional convolutional architectures require a problem to be formulated in a certain way: in particular, they are designed to predict a vector (or a matrix, or a tensor in a more general sense), that is either of a fixed length or whose size depends on the input.

For example, consider the task of scene classification where the goal is to predict the label (or category) of a given image. Modern approaches typically address this by a series of convolutional layers, followed by a number of fully connected layers, which are finally mapped to predict a fixed-sized vector [17, 30, 33]. The length of the predicted vector corresponds to the number of candidate categories, *e.g.* 1,000 for the ImageNet challenge [28]. Each element is a score or probability of a particular category, and the final prediction is a probability distribution over all categories. This strategy is perfectly admissible if one expects to find exactly one or at least the same number of categories across all images. However, natural images typically show multiple entities (*e.g.* table, pizza, person, *etc.*), and what is perhaps more important, this number differs from image to image. During evaluation, this property is not taken into account. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) only counts an error if the "true" label is not among the top-5 candidates. Another strategy to account for multiple classes is to fix the number to a certain value for all test instances, and report precision and recall by counting false positive and false negative predictions, as was done in [11, 38]. Arguably, both methods are suboptimal because in a real-world scenario, where the correct labelling is unknown, the prediction should in fact not only rank all labels according to their likelihood of being present, but also to report how many objects (or labels) are actually present in one particular image. Deciding how many objects are actually present in an image is a crucial part of human scene understanding that is missing from our current evaluation of automated image understanding methods.



(a) Proposals (b) MS-CNN [2] (c) Our result

Figure 1: Example pedestrian detection result of our approach. To select relevant detection candidates from an overcomplete set of proposals *(a)*, state-of-the-art methods rely on non-maximum suppression (NMS) with a fixed setting *(b)*. We show that it is crucial to adjust the NMS threshold adaptively, depending on the number of instances in each image (3 in this case) *(c)*.

As a second example, let us turn to object detection, and in particular pedestrian detection. The parallel to scene classification that we motivated above is that, once again, in real scenarios, the number of people in a particular scene is not known beforehand. The most common approach is to assign a confidence score to a number of region candidates [4, 8, 10, 27], which are typically selected heuristically by thresholding and non-maxima suppression. We ar-

gue that it is important not to simply discard the information about the actual number of objects at test time, but to exploit it while selecting the subset of region proposals.

The examples above motivate and underline the importance of *set prediction* in certain applications. It is important to note that, in contrast to vectors, a set is a collection of elements which is invariant under permutation. One naive way of extracting a set from a distribution is to apply thresholding. One single global threshold, however, cannot produce the desired effect because it is highly dependent on each data sample and should therefore be adjusted for each instance. However, an adaptive threshold per training instance cannot be learned in a straightforward manner as it is not directly observable from the training data. Instead, we use a principled definition of a set as the union of cardinality distribution and family of joint distributions over each cardinality value.

In summary, our main contributions include:

- Starting from the mathematical definition of a set distribution, we derive a loss that enables us to employ existing machine learning methodology to learn this distribution from data.
- We integrate our loss into a deep learning framework to exploit the power of a multi-layer architecture.
- We present state-of-the-art results on standard datasets on the tasks of multi-label image classification and pedestrian detection.

2. Related Work

A sudden success in multiple applications including voice recognition [13], machine translation [32] and image classification [17], has sparked the deployment of deep learning methods throughout numerous research areas. Deep convolutional (CNN) and recurrent (RNN) neural networks now outperform traditional approaches in tasks like semantic segmentation [3], image captioning [16] or object detection [20]. Here, we will briefly review some of the recent approaches to image classification and object detections and point out their limitations.

Image or scene classification is a fundamental task of understanding photographs. The goal here is to predict a scene label for a given image. Early datasets, such as Caltech-101 [7], mostly contained one single object and could easily be described by one category. Consequently, a large body of literature focused on single-class prediction [17, 29, 40, 23]. However, real-world photographs typically contain a collection of multiple objects and should therefore be captioned with multiple tags.

Surprisingly, there exists rather little work on multi-class image classification that makes use of deep architectures. Gong *et al*. [12] combine deep CNNs with a top-k approximate ranking loss to predict multiple labels. Wei *et al*. [39] propose a Hypotheses-Pooling architecture that is specifically designed to handle multi-label output. While both methods open a promising direction, their underlying architectures largely ignore the correlation between multiple labels. To address this limitation, recently, Wang *et al*. [38] proposed a model that combines CNNs and RNNs (convolutional and recurrent networks) to predict a number classes in a sequential manner. RNNs, however, are not suitable for set prediction mainly for two reasons. First, the output represents a sequence and is thus highly dependent on the prediction order, as was shown recently by Vinyals *et al*. [34]. Second, the final prediction may not result in a feasible solution (*e.g*. it may contain the same element multiple times), such that post-processing or heuristics such as beam search must be employed [35, 38]. Here we show that our approach not only guarantees to always predict a valid set, but also outperforms previous methods.

Pedestrian detection can also be viewed as a classification problem. Traditional approaches follow the sliding-window paradigm [36, 4, 37, 8, 1], where each possible (or rather plausible) image region is scored independently to contain or not to contain a person. More recent methods, such as Fast R-CNN [10] or the single-shot multi-box detector (SSD) [20] learn the relevant image features rather than manually engineering them, but retain the sliding window approach.

All the above approaches require some form of post-processing to suppress spurious detection responses that originate from the same person. This is typically addressed by non-maximum suppression (NMS), a greedy optimization strategy with a fixed overlap threshold. Recently, several alternatives have been proposed to replace the greedy NMS procedure. Russel *et al*. [31] perform end-to-end head detection by predicting the bounding boxes sequentially using an LSTM [14]. Their approach, however, processes only a small image region at a time, thus limiting the applicability on crowded scenarios with tens or hundreds of people. Pham *et al*. [26] and Lee *et al*. [18] formulate NMS as a global optimisation problem while Hosang *et al*. [15] propose to learn the NMS algorithm end-to-end using CNNs. Both methods, however, do not consider the number of objects while selecting the final set of boxes. Contrary to existing pedestrian detection approaches, we incorporate the cardinality into the NMS algorithm itself. This leads to an improvement over the state of the art, validated on two benchmarks.

3. Random Vectors vs. Random Finite Sets

To explain our approach, we first review some mathematical background and introduce the notation used throughout the paper.

In statistics, a continuous random variable y is a variable that can take an infinite number of possible values.

A continuous random vector can be defined by stacking several continuous random variables into a vector, $Y = (y_1, \cdots, y_m)$. The mathematical function describing the possible values of a continuous random vector, and their associated joint probabilities, is known as a probability density function (PDF) $p(Y)$ such that $\int p(Y)dY = 1$.

A random finite set (RFS) $\mathcal{Y}$ is a finite-set valued random variable $\mathcal{Y} = \{y_1, \cdots, y_m\}$. The main difference between an RFS and a random vector is that for the former, the number of constituent variables is random and the variables themselves are random and unordered.

A statistical function describing a finite-set variable $\mathcal{Y}$ is a combinatorial probability density function $p(\mathcal{Y})$, which consists of a discrete probability distribution, the so-called cardinality distribution, and a family of joint probability densities on both the number and the value of the constituent variables.

Similar to the definition of a PDF for a random variable, the PDF of an RFS must sum to unity over all possible cardinality values and all possible element values and their permutations, *i.e.*

$$\int p(\mathcal{Y})\delta\mathcal{Y} \triangleq p(\emptyset) + \sum_{m=1}^{\infty} \frac{1}{m!} \int p(\{y_1, \cdots, y_m\}) dy_1 \cdots dy_m = 1, \tag{1}$$

where $f(\emptyset)$ is the probability of the empty set. The PDF of an m-dimensional random vector can be defined in terms of an RFS as:

$$p(y_1, \cdots, y_m) \triangleq \frac{1}{m!} p(\{y_1, \cdots, y_m\}). \tag{2}$$

The factor $m! = \prod_{k=1}^{m} k$ appears because the probability density for a set $\{y_1, \cdots, y_m\}$ must be equally distributed among all the $m!$ possible permutations of the vector [21].

Conventional machine learning approaches, such as Bayesian learning and convolutional neural networks, have been proposed to learn the optimal parameters θ^* of the distribution $p(Y|\theta^*, \mathbf{x})$ which maps the input vector $\mathbf{x}$ to the *output vector* Y. In this paper, we instead propose an approach that can learn a set of parameters $(\theta^*, \mathbf{w}^*)$ for a set distribution that allow one to map the input vector $\mathbf{x}$ into the *output set* $\mathcal{Y}$, i.e. $p(\mathcal{Y}|\theta^*, \mathbf{w}^*, \mathbf{x})$. The additional parameters $\mathbf{w}$ define a PDF over the set cardinality, as we explain in the next section.

4. Deep Set Network

Let us begin by defining a training set $\mathcal{D} = \{\mathcal{Y}_i, \mathbf{x}_i\}$, where each training sample $i = 1, \ldots, n$ is a pair consisting of an input feature $\mathbf{x}_i \in \mathbb{R}^l$ and an output (or label) set $\mathcal{Y}_i = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m\}, \mathbf{y}_k \in \mathbb{R}^d$. In the following we will drop the instance index i for better readability. Note that

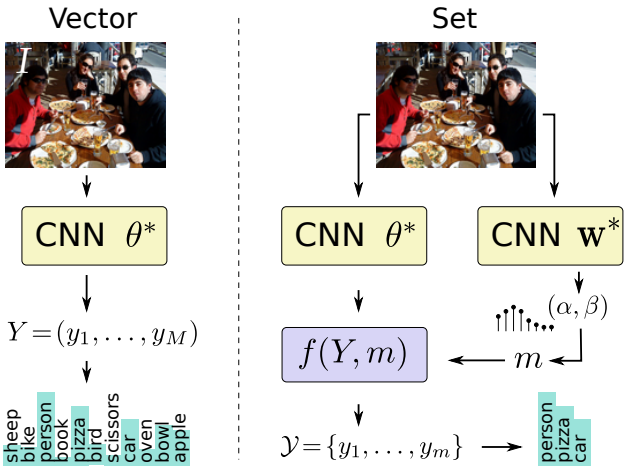


Figure 2: **Left:** Traditional CNNs learn a parameter set $\boldsymbol{\theta}^*$ to predict a fixed *vector* Y. **Right** In contrast, we propose to train a separate CNN to learn a parameter vector $\mathbf{w}$, which is used to predict the set cardinality of a particular output.

$m := |\mathcal{Y}|$ denotes the cardinality of set $\mathcal{Y}$. The probability of a set $\mathcal{Y}$ is defined as:

$$p(\mathcal{Y}|\boldsymbol{\theta}, \mathbf{w}, \mathbf{x}) = p(m|\mathbf{w}, \mathbf{x}) \times m! \times p(\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_m|\boldsymbol{\theta}, \mathbf{x}), \tag{3}$$

where $\boldsymbol{\theta}$ denotes the parameters of $\mathbf{y}_k$ that estimates the distribution of set element values for a fixed cardinality[1], while $\mathbf{w}$ represents the collection of parameters which estimate the cardinality distribution of the set elements. For example, if the outputs (or labels) in the set are independent and identically distributed (i.i.d) and their cardinality follows a Poisson distribution, we can write the likelihood as

$$p(\mathcal{Y}|\boldsymbol{\theta}, \mathbf{w}, \mathbf{x}) = \int p(m|\lambda)p(\lambda|\mathbf{x}, \mathbf{w})d\lambda \times m! \times \left(\prod_{k=1}^{m} p(\mathbf{y}_k|\boldsymbol{\theta}, \mathbf{x}) \right). \tag{4}$$

4.1. Posterior distribution

To learn the parameters $\boldsymbol{\theta}$ and $\mathbf{w}$, we first define the posterior distribution over them as

$$p(\boldsymbol{\theta}, \mathbf{w}|\mathcal{D}) \propto p(\mathcal{D}|\boldsymbol{\theta}, \mathbf{w})p(\boldsymbol{\theta})p(\mathbf{w}) \propto \prod_{i=1}^{n} \left[\int p(m_i|\lambda)p(\lambda|\mathbf{x}_i, \mathbf{w})d\lambda \times m_i! \times \left(\prod_{k=1}^{m_i} p(\mathbf{y}_k|\boldsymbol{\theta}, \mathbf{x}_i) \right) \right] p(\mathbf{x}_i)p(\boldsymbol{\theta})p(\mathbf{w}). \tag{5}$$

[1] This is also known as the spatial distribution of points in point process statistics.

A closed form solution for the integral in Eq. (5) can be obtained by using conjugate priors:

$$\begin{aligned} m &\sim \mathcal{P}(m;\lambda) \\ \lambda &\sim \mathcal{G}(\lambda;\alpha(\mathbf{x},\mathbf{w}),\beta(\mathbf{x},\mathbf{w})) \\ &\quad \alpha(\mathbf{x},\mathbf{w}),\beta(\mathbf{x},\mathbf{w})>0 \quad \forall \mathbf{x},\mathbf{w} \\ \boldsymbol{\theta} &\sim \mathcal{N}(\boldsymbol{\theta};0,\sigma_1^2\mathbf{I}) \\ \mathbf{w} &\sim \mathcal{N}(\mathbf{w};0,\sigma_2^2\mathbf{I}), \end{aligned}$$

where $\mathcal{P}(\cdot,\lambda)$, $\mathcal{G}(\cdot;\alpha,\beta)$, and $\mathcal{N}(\cdot;0,\sigma^2\mathbf{I})$ represent respectively a Poisson distribution with parameters λ, a Gamma distribution with parameters (α,β) and a zero mean normal distribution with covariance equal to $\sigma^2\mathbf{I}$.

We assume that the cardinality follows a Poisson distribution whose mean, λ, follows a Gamma distribution, with parameters which can be estimated from the input data $\mathbf{x}$. Consequently, the integrals in $p(\boldsymbol{\theta},\mathbf{w}|\mathcal{D})$ are simplified and form a negative binomial distribution,

$$\text{NB}\,(m;a,b) = \frac{\Gamma(m+a)}{\Gamma(m+1)\Gamma(a)}.(1-b)^a b^m, \qquad (6)$$

where Γ is the Gamma function. Finally, the full posterior distribution can be written as

$$p(\boldsymbol{\theta},\mathbf{w}|\mathcal{D}) \propto \prod_{i=1}^{n}\left[\text{NB}\left(m_i;\alpha(\mathbf{x}_i,\mathbf{w}),\frac{1}{1+\beta(\mathbf{x}_i,\mathbf{w})}\right) \times m_i! \times \left(\prod_{k=1}^{m_i} p(\mathbf{y}_k|\boldsymbol{\theta},\mathbf{x}_i)\right)\right] p(\boldsymbol{\theta})p(\mathbf{w}). \qquad (7)$$

4.2. Learning

For simplicity, we use a point estimate for the posterior $p(\boldsymbol{\theta},\mathbf{w}|\mathcal{D})$, *i.e.* $p(\boldsymbol{\theta},\mathbf{w}|\mathcal{D}) = \delta(\boldsymbol{\theta}=\boldsymbol{\theta}^*,\mathbf{w}=\mathbf{w}^*|\mathcal{D})$, where $(\boldsymbol{\theta}^*,\mathbf{w}^*)$ are computed using the following MAP estimator:

$$(\boldsymbol{\theta}^*,\mathbf{w}^*) = \arg\max_{\boldsymbol{\theta},\mathbf{w}} \quad \log\left(p\left(\boldsymbol{\theta},\mathbf{w}|\mathcal{D}\right)\right). \qquad (8)$$

The optimisation problem in Eq. (8) can be decomposed w.r.t. the parameters $\boldsymbol{\theta}$ and $\mathbf{w}$. Therefore, we can learn them independently as

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}} \quad \gamma_1\|\boldsymbol{\theta}\| + \sum_{i=1}^{n}\sum_{k=1}^{m_i}\log\left(p(\mathbf{y}_k|\boldsymbol{\theta},\mathbf{x}_i)\right) \qquad (9)$$

and

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} \quad \sum_{i=1}^{n}\left[\log\left(\frac{\Gamma(m_i+\alpha(\mathbf{x}_i,\mathbf{w}))}{\Gamma(m_i+1)\Gamma(\alpha(\mathbf{x}_i,\mathbf{w}))}\right) + \log\left(\frac{\beta(\mathbf{x}_i,\mathbf{w})^{\alpha(\mathbf{x}_i,\mathbf{w})}}{(1+\beta(\mathbf{x}_i,\mathbf{w})^{\alpha(\mathbf{x}_i,\mathbf{w})+m_i)}}\right)\right] - \gamma_2\|\mathbf{w}\|, \qquad (10)$$

where γ_1 and γ_2 are the regularisation parameters, proportional to the predefined covariance parameters σ_1 and σ_2. These parameters are also known as weight decay parameters and commonly used in training neural networks.

The learned parameters $\boldsymbol{\theta}^*$ in Eq. (9) are used to map an input feature vector $\mathbf{x}$ into an output vector Y. For example, in image classification, $\boldsymbol{\theta}^*$ is used to predict the distribution Y over all categories, given the input image $\mathbf{x}$. Note that $\boldsymbol{\theta}^*$ can generally be learned using a number of existing machine learning techniques. In this paper we rely on deep CNNs to perform this task.

To learn the highly complex function between the input feature $\mathbf{x}$ and the parameters (α,β), which are used for estimating the output cardinality distribution, we train a second deep neural network. Using neural networks to predict a discrete value may seem surprising, because these methods at their core rely on the backpropagation algorithm, which assumes a differentiable loss. Note that we achieve this by describing the discrete distribution by continuous parameters α,β (Negative binomial $\text{NB}(\cdot,\alpha,\frac{1}{1+\beta})$), and can then easily draw discrete samples from that distribution. More formally, to estimate $\mathbf{w}^*$, we compute the partial derivatives of the objective function in Eq. (10) w.r.t. $\alpha(\cdot,\cdot)$ and $\beta(\cdot,\cdot)$ and use standard backpropagation to learn the parameters of the deep neural network.

We refer the reader to the supplementary material for the complete derivation of the partial derivatives, a more detailed derivation of the posterior in Eqs. (5)-(7) and the proof for decomposition of the MAP estimation in Eq. (8).

4.3. Inference

Having the learned parameters of the network $(\mathbf{w}^*,\boldsymbol{\theta}^*)$, for a test feature $\mathbf{x}^*$, we use a MAP estimate to generate a set output as

$$\mathcal{Y}^* = \arg\max_{\mathcal{Y}} \quad p(\mathcal{Y}^*|\mathcal{D},\mathbf{x}^*), \qquad (11)$$

where

$$p(\mathcal{Y}^*|\mathcal{D},\mathbf{x}^*) \quad = \quad \int p(\mathcal{Y}^*|\boldsymbol{\theta},\mathbf{w},\mathbf{x}^*)p(\boldsymbol{\theta},\mathbf{w}|\mathcal{D})d\boldsymbol{\theta}d\mathbf{w}$$

and $p(\boldsymbol{\theta},\mathbf{w}|\mathcal{D}) = \delta(\boldsymbol{\theta}=\boldsymbol{\theta}^*,\mathbf{w}=\mathbf{w}^*|\mathcal{D})$. To calculate the mode of the set distribution $p(\mathcal{Y}^*|\mathcal{D},\mathbf{x}^*)$, we first need to calculate the mode m^* of the cardinality distribution

$$m^* = \arg\max_{m} \quad p(m|\mathbf{w}^*,\mathbf{x}^*), \qquad (12)$$

where $p(m|\mathbf{w}^*,\mathbf{x}^*) = \text{NB}\left(m;\alpha(\mathbf{w}^*,\mathbf{x}^*),\frac{1}{1+\beta(\mathbf{w}^*,\mathbf{x}^*)}\right)$. Then, we calculate the mode of the joint distribution for the given cardinality m^* as

$$\mathcal{Y}^* = \arg\max_{\mathcal{Y}_{m^*}} \quad p(\{y_1,\cdots,y_{m^*}\}|\boldsymbol{\theta}^*,\mathbf{x}^*). \qquad (13)$$

It can be seen from Eq. (2) that the mode of $p(\{y_1, \cdots, y_{m^*}\}|\boldsymbol{\theta}^*, \mathbf{x}^*)$ and $p(y_1, \cdots, y_{m^*}|\boldsymbol{\theta}^*, \mathbf{x}^*)$ needs to be identical w.r.t. a fixed permutation. Since the output of the first CNN with the parameters $\boldsymbol{\theta}^*$ is already the mode of $p(y_1, \cdots, y_M|\boldsymbol{\theta}^*, \mathbf{x}^*)$ and the samples are i.i.d., we use the m^* highest values from the output of the first CNN to generate the final set output $\mathcal{Y}^*$.

5. Experimental Results

To evaluate the performance of our proposed deep set network, we perform experiments on two separate and relevant applications: multi-label image classification and pedestrian detection.

5.1. Multi-label Image Classification

In this section, we evaluate our approach on the task of multi-label image classification. As opposed to the more common and more studied problem of (single-label) image classification, the task here is rather to label a photograph with an arbitrary, a-priori unknown number of tags. We perform experiments on two standard benchmarks, the PASCAL VOC 2007 dataset [6] and the Microsoft Common Objects in Context (MS COCO) dataset [19].

Implementation details. In this experiment, similar to [38], we build on the 16-layers VGG network [30], pre-trained on the 2012 ImageNet dataset. We adapt VGG for our purpose by modifying the last fully connected prediction layer to predict 20 classes for PASCAL VOC, and 80 classes for MS COCO. We then fine-tune the entire network for each of these datasets using two commonly used losses for multi-label classification, *softmax* and *binary cross-entropy (BCE)*[2] [11, 38]. To learn both classifiers, we set the weight decay to $5 \cdot 10^{-4}$, with a momentum of 0.9 and a dropout rate of 0.5. The learning rate is adjusted to gradually decrease after each epoch, starting from 0.01 for *softmax* and from 0.001 for *binary cross-entropy*. The learned parameters of these classifiers correspond to θ^* for our proposed deep set network (*cf*. Eq. (9) and Fig. 2).

To learn the cardinality distribution, we use the same VGG-16 network as above and modify the final fully connected layer to predict 2 values for α and β. It is important to note, that the predicted values must be positive to describe a valid Gamma distribution. We therefore also append two weighted sigmoid transfer functions with weights α_M, β_M to ensure that the values predicted for α and β are in a valid range. Our model is not sensitive to these parameters and we set their values to be large enough ($\alpha_M = 160$ and $\beta_M = 20$) to guarantee that the mode of the distribution can accommodate the largest cardinality existing in the dataset. We then fine-tune the network on cardinality distribution using the objective loss defined in Eq. (10). To train the cardinality CNN, we set a constant learning rate 0.001, weight decay $5 \cdot 10^{-12}$, momentum rate 0.9 and dropout 0.5.

Evaluation protocol. To evaluate the performance of the classifiers and our deep set network, we employ the commonly used evaluation metrics for multi-label image classification [11, 38]: *precision* and *recall* of the generated labels per-class (C-P and C-R) and overall (O-P and O-R). Precision is defined as the ratio of correctly predicted labels and total predicted labels, while recall is the ratio of correctly predicted labels and ground-truth labels. In case no predictions (or ground truth) labels exist, *i.e*. the denominator becomes zero, precision (or recall) is defined as $\%100$. To generate the predicted labels for a particular image , we perform a forward pass of the CNN and choose top-k labels according to their scores similar to [11, 38]. Since the classifier always predicts a fixed-sized prediction for all categories, we sweep k from 0 to the maximum number of classes to generate a precision/recall curve. However, for our proposed DeepSet Network, the number of labels per instance is predicted from the cardinality network. Therefore, prediction/recall is not dependent on value k and one single precision/recall value can be computed.

To calculate the per class and overall precision and recall, their average values over all classes and all examples are respectively computed. In addition, we also report the F1 score (the harmonic mean of precision and recall) averaged over all classes (C-F1) and all instances and classes (O-F1).

PASCAL VOC 2007. The Pascal Visual Object Classes (VOC) [6] benchmark is one of the most widely used datasets for detection and classification. It consists of 9963 images with a 50/50 split for training and test, where objects from 20 pre-defined categories have been annotated by bounding boxes. Each image may contain between 1 and 7 unique objects.

We compare our results with a state-of-the-art classifier as described above. The resulting precision/recall plots are shown in Fig. 3(a) together with our proposed approach using the estimated cardinality. Note that by enforcing the correct cardinality for each image, we are able to clearly outperform the baseline w.r.t. both measures. Note also that our prediction (+) can nearly replicate the oracle ($*$), where the ground truth cardinality is known. The mean absolute cardinality error of our prediction on PASCAL VOC is 0.32 ± 0.52.

Microsoft COCO. Another popular benchmark for image captioning, recognition, and segmentation is the recent Microsoft Common Objects in Context (MS-COCO) [19]. The dataset consists of 123 thousand images, each labelled

[2] *Weighted Approximate Ranking* (WARP) objective is another commonly used loss for multi-label classification. However, it does not perform as well as *softmax* and *binary cross-entropy* for the used datasets [38].

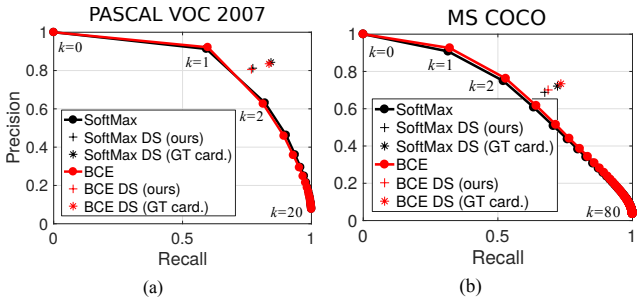


Figure 3: Experimental results on multi-label image classification on two datasets. The baselines (solid curves) represent state-of-the-art classifiers, fine-tuned for each dataset, using two different loss functions. The methods are evaluated by choosing the top-k predictions across the entire dataset, for different k. Our approach predicts k and is thus evaluated only on one single point (+). It outperforms both classifiers significantly in terms of precision and recall and comes very close to the performance when the true cardinality is known ($*$).

with per instance segmentation masks of 80 classes. The number of unique objects for each image can vary between 0 and 18. Around 700 images in the training dataset do not contain any of the 80 classes and there are only a handful of images that have more than 10 tags. The majority of the images contain between one and three labels. We use 82783 images as training and validation split (%90-10%), and the remaining 40504 images as test data. We predict the cardinality of objects in the scene with a mean absolute error of 0.74 and a standard deviation of 0.86.

Fig. 3(b) shows a significant improvement of precision and recall and consequently the F1 score using our deep set network compared to the softmax and binary cross-entropy classifiers for all ranking values k. We also outperform the state-of-the art multi-label classifier CNN-RNN [38], for the reported value of $k = 3$ in Table 1. Our results show around 10 percentage points improvement for the F1 score on top of the baseline classifiers and about 3 percentage points improvement compared to the state of the art on this dataset. Examples of perfect label prediction using our proposed approach are shown in Fig. 4. The deep set network can properly recognise images with no labels at all, as well as images with many tags. We also investigated failure cases where either the cardinality CNN or the classifier fails to make a correct prediction. We showcase some of these cases in Fig 5. We argue here that some of the failure cases are simply due to a missed ground truth annotation, such as the left-most example, but some are actually semantically correct w.r.t. the cardinality prediction, but are penalized during evaluation because a particular object category is not available in the dataset. This is best illustrated in the second example in Fig. 5. Here, our network correctly predicts the number of objects in the scene, which is two, however, the can does not belong to any of the 80 categories in the dataset and is thus not annotated. Similar situations also appear in other images further to the right.

5.2. Pedestrian Detection

To demonstrate the generality of our approach, we test it on an entirely different setting of pedestrian detection. We perform experiments on two widely used datasets, Caltech Pedestrians [5] and MOT16, the 2016 edition of the MOTChallenge benchmark [22]. To push the state-of-the art performance on pedestrian detection, we chose the leading detector on Caltech Pedestrians, which is the recent the multi-scale deep CNN approach (MS-CNN) [2]. Note that we do not retrain the detector, but are still able to improve its performance by predicting the number of pedestrians in each frame.

To learn the cardinality distribution, we use a gray-scale image as the network input, constructed by superimposing all region proposals and their scores generated by MS-CNN detector (before non-maximum suppression approach). We found, that this input provides a stronger signal than the raw RGB images, yielding better results. We build on top of the well-known AlexNet [17] architecture, and replace the first convolutional layer with a single channel filter and last fully connected layer with 2 layers output followed by two weighted sigmoid activation function, similar to the case above (*cf*. Sec. 5.1). The weights are initialised randomly and the network is trained for 100 epochs using the objective loss proposed in Eq. (10). We set a constant learning rate 10^{-3}, weight decay $5 \cdot 10^{-4}$, momentum rate 0.9 and dropout rate 0.5. The checkpoint with the lowest objective on the validation set is chosen for prediction.

Non-maximum suppression. To generate the final detection outputs, most detectors often rely on non-maximum suppression, which greedily picks the boxes with highest scores and suppresses any boxes that overlap more than a pre-defined threshold T_O. In fact, choosing the right threshold is crucial and can have a large impact on the overall performance, as shown *e.g.* in [31]. However, there may not exist a single value T_O, rather it highly depends on the setting. We argue that it should therefore be adjusted for each frame separately. To that end, we use the prediction on the number of people (m) in the scene to choose an adaptive NMS threshold for each image. In particular, we start from the default value of T_O, and adjust it step-wise until the number of boxes reaches m. In the case if the number of final boxes is larger than m, we pick m boxes with the highest scores. For a fair comparison, we also find the best (global) value for T_O for our baseline, the MS-CNN detector.

Table 1: Quantitative results for multi-label image classification on the MS COCO dataset.

Classifier	/Metric Evaluation	C-P	C-R	C-F1	O-P	O-R	O-F1
Softmax	k=3	58.6	57.6	58.1	60.7	63.3	62.0
BCE	k=3	56.2	60.1	58.1	61.6	64.2	62.9
CNN-RNN [38]	k=3	66.0	55.6	60.4	69.2	66.4	67.8
Ours (Softmax)	Est. Card.	**68.2**	59.9	63.8	68.8	67.4	68.1
Ours (BCE)	Est. Card.	66.5	**62.9**	**64.6**	**70.1**	**68.7**	**69.4**

Figure 4: Qualitative results of our multi-class image labelling approach. For each image, the ground truth tags and our predictions are denoted below. Note that we show the exact output of our network, without any heuristics or post-processing.

Table 2: Mean absolute error and standard deviation for cardinality estimation on test sets.

Error	Multi-label classification PASCAL VOC	Multi-label classification MS COCO	Pedestrian detection Caltech	Pedestrian detection MOT16
Mean	0.32	0.74	0.54	1.94
Std	0.52	0.86	0.79	1.96

Evaluation metrics. To quantify the detection performance, we adapt the same evaluation metrics and follow the protocols used on the Caltech detection benchmark [5]. The evaluation metrics used here are log-average miss rate (MR) over false positive per image. Additionally, we compute the F1 score (the harmonic mean of precision recall). The F1 score is computed from *all* detections predicted from our DeepSet network and is compared with the *highest* F1 score along the MS-CNN precision-recall curve. To calculate MR, we concatenate all boxes resulted from our adaptive NMS approach and change the threshold over all scores from our predicted sets.

Caltech Pedestrians [5] is a de-facto standard benchmark for pedestrian detection. The dataset contains sequences captured from a vehicle driving through regular traffic in an urban environment and provides bounding box annotations of nearly $350,000$ pedestrians. The annotations also includes detailed occlusion labels. The number of pedestrians per image varies between 0 and 14. However, more than 55% of the images contain no people at all and around 30% of the data includes one or two persons. We use the MS-CNN [2] network model and its parameters learned on the Caltech training set as θ^* in Eq. (9). To learn the cardinality, we use 4250 images provided as a training set, splitting it into training and validation ($80\% - 20\%$), reaching a mean absolute error of 0.54 (*cf*. Tab. 2). Quantitative detection results are shown in Tab. 3. We achieve state-of-the art performance on this benchmark, improving on the currently best detector by applying our DeepSet network to estimate the number of persons in each image. The improvement, however, is not as significant as in the case of multi-class image classification. We believe that this is mainly due to very limited variation of the number of pedestrians in the scene. For more than 85% of the images containing 0 to 2 pedestrians. A single global threshold for a state-of-the art detection output appears to work almost as well as introducing the knowledge about the number of persons in the scene. Another reason for the relatively low improvement maybe the already remarkable performance of modern detectors on this dataset (*cf*. [5]).

MOTCallenge 2016. Additionally, we evaluate the MS-CNN detector and DeepSet in significantly more crowded and more challenging sequences from MOT16 dataset [22]. This benchmark targeted at multi-object tracking and is not yet commonly used for evaluating the pedestrian detection. However the variation in the number of pedestrians across

GT:	chair, cup, book, keyboard, mouse	banana	person, toothbrush	teddy-bear	oven
Prediction:	chair, cup, book, keyboard, mouse, tv	banana, bottle	person, toothbrush, cell-phone	teddy-bear, bird, car, person	oven, toaster, fridge, bowl, sink, microvawe

Figure 5: Interesting failure cases of our method. The "spurious" TV class predicted on the left is an artifact in annotation because in many examples, computer monitors are actually labelled as TV. In other cases, our network can correctly reason about the number of objects or concepts in the scene, but is constrained by a fixed list of categories defined in the dataset.

Figure 6: Example results of pedestrian detection on selected frames of the MOT16 dataset (cropped for better visibility). Note that we display *all* m boxes, and do not threshold on a certain confidence score. The rightmost image shows a failure example, where a wrong cardinality prediction (6 in this case) forces the detector to output many false positives.

the frames is relatively large (between 0 and 32) and is also distributed more uniformly, which makes correct cardinality estimation more important. Since the labels for the test set are not available, we use the provided training set of this benchmark consisting of 5316 images from 7 different sequences, and divide it into training, validation and test set with split ratios 60%, 15% and 25%, respectively. We only learn the cardinality network $\mathbf{w}^*$ on training set and we use the MS-CNN network model and its parameters learned on the KITTI dataset [9] as θ^* in Eq. (9). The results are summarised in Tab. 3 and shows a more significant improvement over MS-CNN on this dataset compared to Caltech. We believe that these results can be improved further by formulating a more principled way of choosing the m predicted targets in the scene, as opposed to relying on the greedy NMS heuristic.

6. Conclusion

We proposed a deep learning approach for predicting sets. To achieve this goal, we derived a loss for predicting a discrete distribution over the set cardinality. This allowed us to use standard backpropagation for training a deep network for set prediction. We have demonstrated the effectiveness of this approach on multi-class image classification and pedestrian detection, achieving state-of-the-art results in both applications. As our network is trained independently, it can be trivially applied to any existing classifier or detector, to further improve performance.

In future, we plan to extend our model to multi-class cardinality estimation, extending its application to general object detectors. Another potential avenue could be to exploit the Bayesian nature of the model to include uncertainty as opposed to only relying on the MAP estimation. Finally, we have only considered set outputs. We believe a promising direction for a follow up work is to extend our idea to also handle set inputs, which can then be applied to problems like graph matching.

Table 3: Quantitative evaluation of our method on pedestrian detections measured by F1 score (higher is better) and log-average miss rate (lower is better).

	F1-score ↑		log-avg miss ↓	
Method / Dataset	Caltech	MOT16	Calt.	MOT16
MS-CNN [2]	51.61	59.04	60.9	82.8
MS-CNN-DS (ours)	52.15	61.86	60.4	81.7
MS-CNN-DS (GT card.)	52.28	62.42	60.3	81.5

References

[1] R. Benenson, M. Mathias, R. Timofte, and L. V. Gool. Pedestrian detection at 100 frames per second. In *CVPR 2012*. 2

[2] Z. Cai, Q. Fan, R. Feris, and N. Vasconcelos. A unified multi-scale deep convolutional neural network for fast object detection. In *ECCV 2016*. 1, 6, 7, 8

[3] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *ICLR*, 2015. 2

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR 2005*, pages 886–893. 1, 2

[5] P. Dollár, C. Wojek, B. Schiele, and P. Perona. Pedestrian detection: An evaluation of the state of the art. *IEEE T. Pattern Anal. Mach. Intell.*, 34, 2012. 6, 7

[6] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. *The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results*. 5

[7] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE T. Pattern Anal. Mach. Intell.*, 28(4):594–611, Apr. 2006. 2

[8] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *IEEE T. Pattern Anal. Mach. Intell.*, 32(9):1627–1645, 2010. 1, 2

[9] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? The KITTI Vision Benchmark Suite. In *CVPR 2012*. 8

[10] R. Girshick. Fast R-CNN. In *ICCV 2015*. 1, 2

[11] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*, 2013. 1, 5

[12] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe. Deep convolutional ranking for multilabel image annotation. *CoRR*, abs/1312.4894, 2013. 2

[13] A. Graves, A.-r. Mohamed, and G. E. Hinton. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649, 2013. 2

[14] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):17351780, Nov. 1997. 2

[15] J. Hosang, R. Benenson, and B. Schiele. A convnet for non-maximum suppression. In B. Rosenhahn and B. Andres, editors, *gcpr-2016*, volume 9796 of *Lecture Notes in Computer Science*, pages 192–204, Hannover, Germany, 2016. Springer. 2

[16] J. Johnson, A. Karpathy, and L. Fei-Fei. Densecap: Fully convolutional localization networks for dense captioning. In *CVPR 2016*. 1, 2

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*2012*, pages 1097–1105. 1, 2, 6

[18] D. Lee, G. Cha, M.-H. Yang, and S. Oh. Individualness and determinantal point processes for pedestrian detection. In *ECCV 2016*, pages 330–346. DOI: 10.1007/978-3-319-46466-4_20. 2

[19] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár. Microsoft COCO: Common objects in context. *arXiv:1405.0312 [cs]*, May 2014. arXiv: 1405.0312. 5

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV 2016*, Lecture Notes in Computer Science, pages 21–37, 2016. DOI: 10.1007/978-3-319-46448-0_2. 1, 2

[21] R. P. Mahler. *Statistical multisource-multitarget information fusion*, volume 685. Artech House Boston, 2007. 3

[22] A. Milan, L. Leal-Taixé, I. Reid, S. Roth, and K. Schindler. Mot16: A benchmark for multi-object tracking. *arXiv:1603.00831 [cs]*, Mar. 2016. arXiv: 1603.00831. 6, 7

[23] V. N. Murthy, V. Singh, T. Chen, R. Manmatha, and D. Comaniciu. Deep decision network for multi-class image classification. In *CVPR 2016*, June 2016. 2

[24] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR 2016*. 1

[25] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille. Weakly- and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *ICCV 2015*, Dec. 2015. 1

[26] T. T. Pham, S. Hamid Rezatofighi, I. Reid, and T.-J. Chin. Efficient point process inference for large-scale object detection. In *CVPR 2016*. 2

[27] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS*2015*. 1

[28] 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. *Int. J. Comput. Vision*, 115(3):211–252, 2015. 1

[29] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. OverFeat: Integrated recognition, localization and detection using convolutional networks. *CoRR*, abs/1312.6229, 2013. 2

[30] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1, 5

[31] R. Stewart, M. Andriluka, and A. Y. Ng. End-to-end people detection in crowded scenes. In *CVPR 2016*. 2, 6

[32] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*2014*, pages 3104–3112. 2

[33] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 1

[34] O. Vinyals, S. Bengio, and M. Kudlur. Order matters: Sequence to sequence for sets. *arXiv:1511.06391 [cs, stat]*, Nov. 2015. arXiv: 1511.06391. 2

[35] O. Vinyals, M. Fortunato, and N. Jaitly. Pointer networks. In *NIPS*2015*, pages 2692–2700. 2

[36] P. Viola and M. J. Jones. Robust real-time face detection. *Int. J. Comput. Vision*, 57(2):137–154, May 2004. 2

[37] S. Walk, N. Majer, K. Schindler, and B. Schiele. New features and insights for pedestrian detection. In *CVPR 2010*. 2

[38] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu. CNN-RNN: A unified framework for multi-label image classification. In *CVPR*, June 2016. 1, 2, 5, 6, 7

[39] Y. Wei, W. Xia, J. Huang, B. Ni, J. Dong, Y. Zhao, and S. Yan. CNN: Single-label to multi-label. *CoRR*, abs/1406.5726, 2014. 2

[40] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *ECCV 2014*, pages 818–833. DOI: 10.1007/978-3-319-10590-1_53. 2