The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Joint Learning of Set Cardinality and State Distribution

S. Hamid Rezatofighi,[1] Anton Milan,[2*] Qinfeng Shi,[1] Anthony Dick,[1] Ian Reid[1]

[1]School of Computer Science, The University of Adelaide, Australia
[2]Amazon Development Center, Germany
firstname.lastname@adelaide.edu.au

Abstract

We present a novel approach for learning to predict sets using deep learning. In recent years, deep neural networks have shown remarkable results in computer vision, natural language processing and other related problems. Despite their success, traditional architectures suffer from a serious limitation in that they are built to deal with structured input and output data, *i.e.* vectors or matrices. Many real-world problems, however, are naturally described as sets, rather than vectors. Existing techniques that allow for sequential data, such as recurrent neural networks, typically heavily depend on the input and output order and do not guarantee a valid solution. Here, we derive in a principled way, a mathematical formulation for set prediction where the output is permutation invariant. In particular, our approach jointly learns both the cardinality and the state distribution of the target set. We demonstrate the validity of our method on the task of multi-label image classification and achieve a new state of the art on the PASCAL VOC and MS COCO datasets.

Introduction

Recently, deep structured networks such as deep convolutional (CNN) and recurrent (RNN) neural networks have become increasingly popular in artificial intelligence, showing remarkable performance on many real-world problems, including scene classification (Krizhevsky, Sutskever, and Hinton 2012), speech recognition (Hinton et al. 2012), gaming (Mnih et al. 2013; 2015), semantic segmentation (Papandreou et al. 2015), and image captioning (Johnson, Karpathy, and Fei-Fei 2016). However, like most machine learning techniques, current deep learning approaches are based on conventional statistics and require the problem to be formulated in a structured way. In particular, they are designed to learn a model for a distribution (or a function) that maps a structured input, typically a vector, a matrix, or a tensor, to a structured output.

Consider the task of image classification as an example. The goal here is to predict a label (or a category) of a given image. The most successful approaches address this task with CNNs, *i.e.* by applying a series of convolutional layers followed by a number of fully connected layers (Krizhevsky, Sutskever, and Hinton 2012; Simonyan and Zisserman 2014; Szegedy et al. 2014; He et al. 2016). The final output layer is a fixed-sized vector with the length corresponding to the number of categories in the dataset (*e.g.* 1000 in the case of the ILSVR Challenge (Russakovsky et al. 2015)). Each element in this vector is a score or probability for one particular category such that the final prediction corresponds to a probability distribution over all classes. The difficulty arises when the number of classes is unknown in advance and in particular varies for each example. Then, the final output is generated heuristically by a discretization process such as choosing the k highest scores (Gong et al. 2013a; Wang et al. 2016), which is not part of the learning process. This shortcoming concerns not only image tagging but also other problems like detection or graph optimization, where connectivity and graph size can be arbitrary.

We argue that such problems can be naturally expressed with sets 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. Therefore, learning approaches built on conventional statistics cannot be the right choice for these problems. In this paper, we propose a learning approach based on point processes and finite set statistics to deal with sets in a principled manner. More specifically, in the presented model, we assume that the input (the observation) is still structured, but the output is modelled as a set. Our approach is inspired by a recent work on set learning using deep neural networks (Rezatofighi et al. 2017). The main limitation of that work, however, is that the approach employs two sets of independent weights (two independent networks) to generate the cardinality and state distributions of the output set. In addition, to generate the final output as a set, sequential inference has to be applied instead of joint inference. In this paper, we derive a principled formulation for performing both learning and inference steps jointly. The main contribution of the paper is summarised as follows:

1. We present a novel way to learn both cardinality and state distributions jointly within a single deep network. Our model is learned end-to-end to generate the output set.
2. We perform the inference step both jointly and optimally. We show how we can generate the most likely (the optimal) set using MAP inference for our given model.
3. Our approach outperforms existing solutions and achieves

*The work was carried out at the University of Adelaide.

state-of-the-art results on the task of multi-label image classification on two standard datasets.

Related Work

Handling unstructured input and output data, such as sets or point patterns, for both learning and inference is an emerging field of study that has generated substantial interest in recent years. Approaches such as mixture models (Blei, Ng, and Jordan 2003; Hannah, Blei, and Powell 2011; Tran et al. 2016), learning distributions from a set of samples (Muandet et al. 2012; Oliva, Póczos, and Schneider 2013), model-based multiple instance learning (Vo et al. 2017) and novelty detection from point pattern data (Vo et al. 2016), can be counted as few out many examples that use point patterns or sets as input or output and directly or indirectly model the set distributions. However, existing approaches often rely on parametric models, *e.g.* the elements in output sets needs to be derived from an independent and identically distributed (*i.i.d.*) Poisson point process distribution (Adams, Murray, and MacKay 2009; Vo et al. 2016). Recently, deep learning has enabled us to use less parametric models to capture highly complex mapping distributions between structured inputs and outputs. Somewhat surprisingly, there are only few works on learning sets using deep neural networks. One interesting exception in this direction is the recent work of Vinyals *et al.* (2015), which uses an RNN to read and predict sets. However, the output is still assumed to have an ordered structure, which contradicts the orderless (or permutation invariant) property of sets. Moreover, the framework can be used in combination with RNNs only and cannot be trivially extended to any arbitrary learning framework such as feed-forward architectures. Another recent work proposed by Zaheer *et al.* (2017) is a deep learning framework which can deal with sets as input with different sizes and permutations. However, the outputs are either assumed to be structured, *e.g.* a scalar as a regressing score, or a set with the same entities of the input set, which prevents this approach to be used for the problems that require output sets with arbitrary entities. Perhaps the most related work to our problem is a deep set network recently proposed by Rezatofighi *et al.* (2017) which seamlessly integrates a deep learning framework into set learning in order to learn to predict sets in two challenging computer vision applications, image tagging and pedestrian detection. However, the approach requires to train two independent networks to model a set, one for cardinality and one for state distribution. Our approach is largely inspired by this latter work but overcomes its limitation on independent learning and inference.

To validate our model, we apply it on the multi-label image classification task. Despite its relevance, there exists rather little work on this problem that makes use of deep architectures. One example is Gong *et al.* (2013b), who combine deep CNNs with a top-k approximate ranking loss to predict multiple labels. Wei *et al.* (2014) 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.* (2016) proposed a model that combines CNNs and RNNs to predict an arbitrary number of classes in a sequential manner. RNNs, however, are not suitable for set prediction mainly for two reasons. First, the output represents a sequence and not a set, and is thus highly dependent on the prediction order, as was shown recently by Vinyals *et al.* (2015). 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 (Vinyals, Fortunato, and Jaitly 2015; Wang et al. 2016). Here we show that our approach not only guarantees to always predict a valid set, but also outperforms previous methods.

Background

To better 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 fixed length 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 $\int p(Y)dY = 1$.

In contrast, 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, m, is random and the variables themselves are random and unordered. Throughout the paper, we use $\mathcal{Y}$ for a set with unknown cardinality, $\mathcal{Y}^m$ for a set with known cardinality m and $Y = (y_1, \cdots, y_m)$ for a vector (or an ordered set) with known dimension m.

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 values of the constituent variables (Mahler 2007; Vo et al. 2017), *i.e.*

$$\begin{aligned} p(\mathcal{Y}) &= p(m)U^m p_m(\{y_1, y_2, \cdots, y_m\}) \\ &= p(m)m!U^m p_m(y_1, y_2, \cdots, y_m), \end{aligned} \tag{1}$$

where $p(m)$ is the cardinality distribution of the set $\mathcal{Y}$ and $p_m(\{y_1, y_2, \cdots, y_m\})$ is a symmetric joint probability density distribution of the set $\mathcal{Y}^m$ given known cardinality m. The normalisation factor $m! = \prod_{k=1}^{m} k$ between $p_m(\mathcal{Y}^m)$ and $p_m(Y)$ appears because the probability density for a set with known cardinality $\mathcal{Y}^m$ must be equally distributed among all the $m!$ possible permutations of the corresponding vector Y (Mahler 2007; Vo et al. 2017). U is the unit of hypervolume in the feature space, which cancels out the unit of the probability density $p_m(\cdot)$ making it unit-less, and thereby avoids the unit mismatch across the different dimensions (cardinalities). Without this normalizing constant, the comparison between probabilities of the sets with different cardinalities is not properly defined because a distribution with the smallest set size will always have the highest probabilities. For

example, $p(y_1) \geq p(y_1, y_2)$ always holds regardless of the particular choice for y_1 and y_2. Please refer to (Vo et al. 2017) for an intuitive discussion.

Finite Set Statistics provides powerful and practical mathematical tools for dealing with random finite sets, based on the notion of integration and density that is consistent with the point process theory (Mahler 2007).[1] For example, 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 as well as their permutations. This type of statistics, which is derived from the point process stochastic process, defines basic mathematical operations on finite sets such as functions, derivatives and integrations as well as other statistical tools such as probability density function of a random finite set and its statistical moments (Mahler 2007; Vo et al. 2017). For further details on point processes, we refer the reader to textbooks such as (Chiu et al. 2013; Daley and Vere-Jones 2007; Moller and Waagepetersen 2003).

Conventional machine learning approaches, such as Bayesian learning and convolutional neural networks, have been proposed to learn the optimal parameters $\mathbf{w}^*$ of the distribution $p(Y|\mathbf{x}, \mathbf{w}^*)$ which maps the input vector $\mathbf{x}$ to the *output vector* Y. In this paper, we instead propose an approach that can learn the parameters $\mathbf{w}^*$ for a set distribution that allows one to map the input vector $\mathbf{x}$ to the *output set* $\mathcal{Y}$, *i.e.* $p(\mathcal{Y}|\mathbf{x}, \mathbf{w}^*)$. For mathematical convenience, we use an *i.i.d.*-cluster point process model. Moreover, we target applications where the order of the outputs during training is irrelevant, *e.g.* multi-label image classification. Modifying the application or the *i.i.d.* assumption to non-*i.i.d.* set elements, may require to deal with the complexity of permutation invariant property of sets during the learning step, which leads to serious mathematical complexities and is left for future work.

Joint Deep Set Network

We follow the convention introduced in (Rezatofighi et al. 2017) and define a training set $\mathcal{D} = \{(\mathbf{x}_i, \mathcal{Y}_i)\}$, where each training sample $i = 1, \ldots, n$ is a pair consisting of an input feature (*e.g.* image), $\mathbf{x}_i \in \mathbb{R}^l$ and an output (or label) set $\mathcal{Y}_i = \{y_1, y_2, \ldots, y_{m_i}\}, y_k \in \mathbb{R}^d, m_i \in \mathbb{N}^0$. In the following we will drop the instance index i for better readability. Note that $m := |\mathcal{Y}|$ denotes the cardinality of set $\mathcal{Y}$. Following the definition in Eq.(1), the probability density of a set $\mathcal{Y}$ with an unknown cardinality is defined as

$$\begin{aligned} p(\mathcal{Y}|\mathbf{x}, \mathbf{w}) =& p(m|\mathbf{x}, \mathbf{w}) \times U^m \\ & \times p_m(\{y_1, y_2, \cdots, y_m\}|\mathbf{x}, \mathbf{w}), \end{aligned} \tag{2}$$

where $\mathbf{w}$ denotes the collection of parameters which model both the *cardinality* distribution of the set elements $p(m|\cdot)$ as well as the parameters of y_k that model the joint distribution of set element *values* for a fixed cardinality $p_m(\{y_1, y_2, \cdots, y_m\}|\cdot)$. Note that in contrast to previous works (Rezatofighi et al. 2017; Vo et al. 2016; 2017) that assume that two sets of independent parameters (two independent networks) are required to represent the set distribution $p(\mathcal{Y}|\cdot)$, we will show that one set of parameters $\mathbf{w}$ is sufficient to learn this distribution and as it turns out also yields better performance.

The above formulation represents the probability density of a set which is very general and completely independent of the choices of both cardinality and state distributions. It is thus straightforward to transfer it to many applications that require the output to be a set. However, to make the problem amenable to mathematical derivation and implementation, we adopt two assumptions: *i)* the outputs (or labels) in the set are derived from an independent and identically distributed (*i.i.d.*)-cluster point process model, and *ii)* their cardinality follows a categorical distribution parameterised by event probabilities $\boldsymbol{\rho}$. Thus, we can write the distribution as

$$\begin{aligned} p(\mathcal{Y}|\mathbf{w}, \mathbf{x}) = & \int p(m|\boldsymbol{\rho}) p(\boldsymbol{\rho}|\mathbf{x}, \mathbf{w}) d\boldsymbol{\rho} \times U^m \\ & \times \left(\prod_{y \in \mathcal{Y}^m} p(\{y\}|\mathbf{x}, \mathbf{w}) \right), \end{aligned} \tag{3}$$

where $p(\{y\}|\cdot, \cdot)$ denotes the probability of taking on the state y in a singleton set $\{y\}$, and $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_M)$ is the vector of event probabilities, *i.e.* $\sum_{i=1}^{M} \rho_i = 1$ and $\rho_i > 0, \forall i \in \{1, \ldots, M\}$.

Posterior distribution

To learn the parameters $\mathbf{w}$, we assume that the training samples are independent from each other and that the distribution $p(\mathbf{x})$ from which the input data is sampled is independent from both the output and the parameters. Then, the posterior distribution over the parameters can be derived as

$$\begin{aligned} p(\mathbf{w}|\mathcal{D}) &\propto p(\mathcal{D}|\mathbf{w}) p(\mathbf{w}) \\ &= \prod_{i=1}^{n} \left[\int p(m_i|\boldsymbol{\rho}) p(\boldsymbol{\rho}|\mathbf{x}_i, \mathbf{w}) d\boldsymbol{\rho} \times U^{m_i} \right. \\ & \qquad \left. \times \left(\prod_{y \in \mathcal{Y}_i^{m_i}} p(\{y\}|\mathbf{x}_i, \mathbf{w}) \right) \right] p(\mathbf{w}). \end{aligned} \tag{4}$$

A closed-form solution for the integral in Eq. (4) can be obtained by using conjugate priors:

$$\begin{aligned} m &\sim \text{Cat}(m; \boldsymbol{\rho}) \\ \boldsymbol{\rho} &\sim \text{Dir}(\boldsymbol{\rho}; \boldsymbol{\alpha}(\mathbf{x}, \mathbf{w})) \\ & \quad \boldsymbol{\alpha}(\mathbf{x}, \mathbf{w}) > 0 \quad \forall \mathbf{x}, \mathbf{w}, \end{aligned}$$

where $\text{Cat}(\cdot, \boldsymbol{\rho})$ and $\text{Dir}(\cdot; \boldsymbol{\alpha})$ represent respectively a categorical distribution with the event probabilities $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_M)$ and a Dirichlet distribution with parameters $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_M)$. Moreover, $p(\mathbf{w})$ can be assumed a zero-mean normal distribution with covariance equal to $\sigma^2\mathbf{I}$, *i.e.* $p(\mathbf{w}) = \mathcal{N}(\cdot; 0, \sigma^2\mathbf{I})$. The key difference between our method and (Rezatofighi et al. 2017) is that we only need to use one network as opposed to two networks used in the previous work. It is important to note that our method *jointly*

[1] A random finite set can be viewed as a simple finite point process (Baddeley, Bárány, and Schneider 2007).

predicts both cardinality and the set elements as opposed to sequentially predicting the cardinality first and then the set elements as previously done in (Rezatofighi et al. 2017). We have provided a comparison between the graphical models of both methods in terms of plate notation in Fig. 1 to further illustrate their differences.

We assume that the cardinality follows a categorical distribution whose event probabilities vector $\boldsymbol{\rho}$ is estimated from a Dirichlet distribution with parameters $\boldsymbol{\alpha}$, which can be directly estimated from the input data $\mathbf{x}$. Note that the cardinality distribution in Eq. (3) can be replaced by any other discrete distribution, *e.g.* Poisson, binomial or negative binomial (*cf.* (Rezatofighi et al. 2017)). Here, we use the categorical distribution as the cardinality model, which better suits the task at hand. The rationale here is that Poisson and negative binomial are long-tailed distributions and their variance increases with their mean. Therefore, the final model will have more uncertainty (and possibly a higher error) in estimating the cardinality of the sets with high values. In contrast, the categorical distribution does not have the drawback of correlating its mean and variance. Moreover, in the image tagging application, the maximum cardinality is often known and there is no need to use long-tailed distributions, which are more suitable for the applications where the maximum cardinality is unknown.

Consequently, the integral in Eq. (4) is simplified and forms a Dirichlet-Categorical distribution

$$DC\left(m;\boldsymbol{\alpha}\right)=\frac{\alpha_m+C_m}{\sum_{\acute{m}}\alpha_{\acute{m}}+C}, \tag{5}$$

where C_m is the number of samples in the training set with cardinality m, and C is the total number of training samples. Finally, the full posterior distribution can be written as

$$p(\mathbf{w}|\mathcal{D})\propto\prod_{i=1}^{n}\Bigg[DC\left(m_i;\boldsymbol{\alpha}(\mathbf{x}_i,\mathbf{w})\right)\times U^{m_i} \times\left(\prod_{y\in\mathcal{Y}_i^{m_i}}p(\{y\}|\mathbf{x}_i,\mathbf{w})\right)\Bigg]p(\mathbf{w}). \tag{6}$$

Learning

For simplicity, we use a point estimate for the posterior $p(\mathbf{w}|\mathcal{D})$, *i.e.* $p(\mathbf{w}|\mathcal{D})=\delta(\mathbf{w}=\mathbf{w}^*|\mathcal{D})$, where $\mathbf{w}^*$ is computed using the MAP estimator, *i.e.* $\mathbf{w}^*=\arg\max_{\mathbf{w}}\ \log\left(p\left(\mathbf{w}|\mathcal{D}\right)\right)$. Therefore, we have

$$\mathbf{w}^*=\arg\max_{\mathbf{w}}\ \sum_{i=1}^{n}\Bigg[\sum_{y\in\mathcal{Y}_i^{m_i}}\Big(\log\big(p(\{y\}|\mathbf{x}_i,\mathbf{w})\big)\Big) +m_i\log U+\log\left(DC\left(m_i;\boldsymbol{\alpha}(\mathbf{x}_i,\mathbf{w})\right)\right)\Bigg]-\gamma\|\mathbf{w}\|_2^2. \tag{7}$$

$p(\{y\}|\mathbf{x}_i,\mathbf{w})$ describes a neural network with coefficients $\mathbf{w}$ learned to map the input $\mathbf{x}_i$ to the output (label) y. This function represents the *state distribution* of each set element over the state space. γ is the regularisation parameter, proportional to the predefined covariance parameter σ. This parameter is also known as the weight decay parameter and is commonly used in training neural networks.

For example, in the application to multi-label image classification, $y\in\mathcal{Y}_i^{m_i}\subseteq\{\ell_1,\ell_2,\cdots,\ell_M\}$ represents the existence of a specific label ℓ_j in the input image instance $\mathbf{x}_i$ from all pre-defined M labels. In this application, we can rewrite an equivalent binary formulation for the above MAP problem as

$$\begin{aligned}\mathbf{w}^*&=\arg\max_{\mathbf{w}}\ \sum_{i=1}^{n}\Bigg[\sum_{\ell=1}^{M}z_i^\ell\log p(z_i^\ell|\mathbf{x}_i,\mathbf{w})+\sum_{\ell=1}^{M}z_i^\ell\log U\\&\quad+\log DC\left(\sum_{\ell=1}^{M}z_i^\ell;\boldsymbol{\alpha}(\mathbf{x}_i,\mathbf{w})\right)\Bigg]-\gamma\|\mathbf{w}\|_2^2\\&=\arg\max_{\mathbf{w}}\ \sum_{i=1}^{n}\Bigg[\sum_{\ell=1}^{M}z_i^\ell\log p(z_i^\ell|\mathbf{x}_i,\mathbf{w})+\\&\quad\log DC\left(\sum_{\ell=1}^{M}z_i^\ell;\boldsymbol{\alpha}(\mathbf{x}_i,\mathbf{w})\right)\Bigg]-\gamma\|\mathbf{w}\|_2^2,\end{aligned} \tag{8}$$

where $z_i^\ell\in\{0,1\}$ represents the existence or non-existence of any specific label in the image $\mathbf{x}_i$. $p(z_i^\ell=1|\mathbf{x}_i,\mathbf{w})$ can be defined as a binary logistic regression function

$$p(z_i^\ell=1|\mathbf{x}_i,\mathbf{w})=\frac{\exp O^\ell(x_i,\mathbf{w})}{1+\exp O^\ell(x_i,\mathbf{w})},$$

where $O^\ell(x_i,\mathbf{w})$ is the network's predicted output corresponding to the ℓ^{th} label.

Note that $\mathbf{w}$ can generally be learned using a number of existing machine learning techniques. In this paper we rely on deep CNNs to perform this task. More formally, to estimate $\mathbf{w}^*$, we compute the partial derivatives of the objective function in Eq. (8) and use standard backpropagation to learn the parameters of the deep neural network.

Inference

Having learned the network parameters $\mathbf{w}^*$, for a test image $\mathbf{x}^+$, we use a MAP estimate to generate a set output as

$$\mathcal{Y}^*=\arg\max_{\mathcal{Y}}\ p(\mathcal{Y}|\mathcal{D},\mathbf{x}^+,\mathbf{w}^*), \tag{9}$$

where $p(\mathcal{Y}|\mathcal{D},\mathbf{x}^+,\mathbf{w}^*)\propto\int p(\mathcal{Y}|\mathbf{w},\mathbf{x}^+)p(\mathbf{w}|\mathcal{D})d\mathbf{w}$, and $p(\mathbf{w}|\mathcal{D})=\delta(\mathbf{w}=\mathbf{w}^*|\mathcal{D})$ as above. Therefore, the MAP estimate can be written as follows,

$$\begin{aligned}\mathcal{Y}^*&=\arg\max_{\mathcal{Y}}\ p(\mathcal{Y}|\mathcal{D},\mathbf{x}^+,\mathbf{w}^*)\\&=\arg\max_{\mathcal{Y}}\ \log\big(p(\mathcal{Y}|\mathcal{D},\mathbf{x}^+,\mathbf{w}^*)\big)\\&=\arg\max_{m,\mathcal{Y}^m}\ \log DC\left(m;\boldsymbol{\alpha}(\mathbf{x}^+,\mathbf{w}^*)\right)+m\log U\\&\qquad+\sum_{y\in\mathcal{Y}^m}\log\left(p(\{y\}|\mathbf{x}^+,\mathbf{w}^*)\right).\end{aligned} \tag{10}$$

Since the unit of hyper-volume U in this application is unknown, we assume it as a constant hyper-parameter, estimated from the validation set of the data.

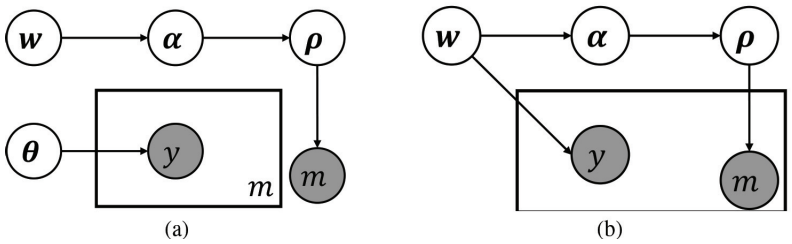

Figure 1: Comparison of the graphical models: a) The set learning approach introduced in (Rezatofighi et al. 2017) by replacing Dirichlet-Categorical as its cardinality distribution; (b) our proposed joint set learning. The work in (Rezatofighi et al. 2017) first predicts the cardinality m, and then the labels ys given m. There is a separation between parameters $\mathbf{w}$ and $\boldsymbol{\theta}$. Consequently, an incorrect m predicted via $\mathbf{w}$ can not be fixed by $\boldsymbol{\theta}$. Our method only uses one joint parameter $\mathbf{w}$ which aims to learn and predict the best m and y's *jointly*. y and m are shaded as they are observed in the training data. Note that m is a variable in our model (b) which determines the repetition of the plates (*i.e.* the number of y's). Removing the top-right chain in (a) recovers the traditional vector based (non-set based) method.

To solve the above inference problem, we define the binary variable $z^\ell \in \{0,1\}$ for existence of each label similar to the learning process. Therefore, an equivalent formulation for Eq. (10) is

$$Z^* = \arg\max_Z \ \log DC\left(\sum_{\ell=1}^{M} z^\ell; \boldsymbol{\alpha}(\mathbf{x}^+, \mathbf{w}^*)\right) + \sum_{\ell=1}^{M} z^\ell \log U + \sum_{\ell=1}^{M} z^\ell \log\left(\frac{\exp O^\ell(\mathbf{x}^+, \mathbf{w}^*)}{1+\exp O^\ell(\mathbf{x}^+, \mathbf{w}^*)}\right), \tag{11}$$

where $Z = (z^1, \cdots, z^M) \in \{0,1\}^M$. The above problem can be seen as a combination of a higher-order term,

$$f(\mathbf{1}^T Z, \boldsymbol{\alpha}(\cdot)) = \log DC\left(\sum_{\ell=1}^{M} z^\ell; \boldsymbol{\alpha}(\cdot)\right), \tag{12}$$

which accounts for the total number of selected variables, and a linear binary program, $C^T Z$, where $C = (c^1, \cdots, c^M)$ and

$$c^\ell = \log U + \log\left(\frac{\exp O^\ell(\mathbf{x}^+, \mathbf{w}^*)}{1+\exp O^\ell(\mathbf{x}^+, \mathbf{w}^*)}\right). \tag{13}$$

Therefore, we can re-write it as

$$Z^* = \arg\max_Z \ f(\mathbf{1}^T Z, \boldsymbol{\alpha}(\cdot)) + C^T Z. \tag{14}$$

Since for each specific cardinality $m = \mathbf{1}^T Z$, the most likely set corresponds to the m highest values of C, the optimal solution for m can be found efficiently when the sorted values of C, here denoted by $C_{st} = (c^1_{st}, \cdots, c^M_{st})$, and $f(\cdot)$ is maximised *w.r.t.* m:

$$m^* = \arg\max_m \ f(m, \boldsymbol{\alpha}(\cdot)) + \sum_{\ell=1}^{m} c^\ell_{st}. \tag{15}$$

Then, the optimal Z^* can be obtained by solving a simple linear program:

$$Z^* = \arg\max_Z \quad C^T Z, \qquad \text{s. t.} \quad \mathbf{1}^T Z = m^*. \tag{16}$$

Note that the optimal solution to the problem in Eq. (16) are exactly those variables that correspond to the m^* highest values of C.

Experimental Results

To validate our proposed joint set learning approach, we perform experiments on the task of multi-label image classification. This is an appropriate application for our model as its output is expected to be in the form of a set (a set of labels in this particular case) with an unknown cardinality while the order of its elements (labels) in the output list does not have any meaning. Moreover, we assume that the labels are derived from an *i.i.d.*-cluster process model. To make our work directly comparable to (Rezatofighi et al. 2017), we use the same two standard and popular benchmarks, the PASCAL VOC 2007 dataset (Everingham et al. 2007) and the Microsoft Common Objects in Context (MS COCO) dataset (Lin et al. 2014).

Implementation details. We follow the same experimental setup used in (Rezatofighi et al. 2017; Wang et al. 2016). Our model is built on the 16-layers VGG network (Simonyan and Zisserman 2014), pretrained on the 2012 ImageNet dataset. We adapt VGG for our purpose by modifying the last fully connected prediction layer to predict both cardinality and classification distributions according to the loss proposed in Eq. (8), *i.e.* DC for cardinality and *binary cross-entropy* for classification. We then fine-tune the entire network using the training set of these datasets with the same train/test split as in existing literature (Rezatofighi et al. 2017; Gong et al. 2013a; Wang et al. 2016).

To train our network, which we call JDS in the following, we use stochastic gradient descent and set the weight decay to $\gamma = 5 \cdot 10^{-4}$, 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 0.001. The network is trained for 60 epochs for both datasets and the epoch with the lowest validation objective value is chosen for evaluation on the test set. The hyper-parameter U is set to be 2.36, adjusted on the validation set.

To demonstrate that joint learning is helpful to learn a better classifier (state distribution) as well as a better cardinality distribution, we perform an additional baseline experiment where we replace the negative binomial (NB) distribution used in (Rezatofighi et al. 2017) with the Dirichlet-

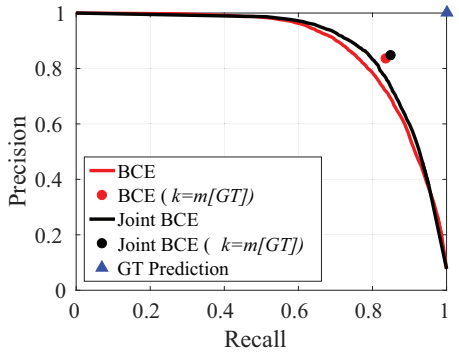

Figure 2: Precision/recall curves for the classification scores when the classifier is trained independently (red solid line) and when it is trained jointly with the cardinality term using our proposed joint approach (black solid line) on PASCAL VOC dataset. The circles represent the upper bound when ground truth cardinality is used for the evaluation of the corresponding classifiers. The ground truth prediction is shown by a blue triangle.

Categorical (DC) distribution from Eq. (5). Then, an independent cardinality distribution network is trained using the same network structure as the one used in (Rezatofighi et al. 2017) while modifying the final fully connected layer to predict the cardinality using the DC distribution. We finetune the network on cardinality distribution, initialised with the network weights learned for the classification task of each of the reported datasets, *i.e.* PASCAL VOC and MS COCO. To train the cardinality CNN, we use the exact same hyper-parameters and training strategy as described above.

Evaluation protocol. We employ the common evaluation metrics for multi-label image classification also used in (Gong et al. 2013a; Wang et al. 2016; Rezatofighi et al. 2017). These include the average *precision*, *recall* and *F1-score*[2] of the generated labels, calculated per-class (C-P, C-R and C-F1) and overall (O-P, O-R and O-F1). Since C-P, C-R and C-F1 can be biased to the performance of the most frequent classes, we also report the average *precision*, *recall* and *F1-score* of the generated labels per image/instance (I-P, I-R and I-F1).

We rely on F1-score to rank approaches on the task of label prediction. A method with better performance has a precision/recall value that has a closer proximity to the perfect point shown by the blue triangle in Fig. 2. To this end, for the classifiers such as BCE and Softmax, we find the optimal evaluation parameter $k = k^*$ that maximises the F1-score. For the deep set network (DS) (Rezatofighi et al. 2017) and our proposed joint set network (JDS), prediction/recall is not dependent on the value of k. Rather, one single value for precision, recall and F1-score is computed.

PASCAL VOC 2007. We first test our approach on the Pascal Visual Object Classes benchmark (Everingham et al. 2007), which is one of the most widely used datasets for detection and classification. This dataset includes 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 contains between 1 and 7 unique objects.

We first investigate if the joint learning improves the performance of cardinality and classifier. Fig. 2 shows the precision/recall curves for the classification scores when the classifier is trained solely using binary cross-entropy (BCE) loss (red solid line) and when it is trained using the same loss jointly with the cardinality term (Joint BCE). We also evaluate the precision/recall values when the ground truth cardinality $m[GT]$ is provided. The results confirm our claim that the joint learning indeed improves the classification performance. We also calculate the mean absolute error of the cardinality estimation when the cardinality term using the DC loss is learned jointly and independently as proposed in (Rezatofighi et al. 2017). The mean absolute cardinality error of our prediction on PASCAL VOC is 0.31 ± 0.54, while this error is 0.33 ± 0.53 when the cardinality is learned independently. We compare the performance of our proposed joint deep set network, *i.e.* JDS (BCE-DC), with softmax and BCE classifiers with the best k value as well as the deep set network (Rezatofighi et al. 2017) when the classifier is binary cross entropy and the cardinality loss is negative binomial, *i.e.* DS (BCE-NB). In addition, Table 1 reports the results for the deep set network when the cardinality loss is replaced by our proposed Dirichlet-Categorical loss, *i.e.* (BCE-DC). The results show that we outperform the other approaches *w.r.t.* all three types of F1-scores. In addition, our joint formulation allows for a single training step to obtain the final model, while the deep set network learns two VGG networks to generate the output sets.

Microsoft COCO.

The MS-COCO (Lin et al. 2014) benchmark is another popular benchmark for image captioning, recognition, and segmentation. The dataset includes 123K images, each labelled with per instance segmentation masks of 80 classes. The number of unique objects for each image varies between 0 and 18. Around 700 images in the training set do not contain any of the 80 classes and there are only a handful of images that have more than 10 tags. Most images contain between one and three labels. We use 82783 images with identical training and validation split as (Rezatofighi et al. 2017), and the remaining 40504 images as test data.

The classification results on this dataset are reported in Table 2. The results once again show that our approach consistently outperforms our baselines and the other methods measured by F1-score. Due to this improvement, we achieve state-of-the-art results on this dataset as well. Some examples of label prediction using our joint deep set network and comparison with other deep set networks are shown in Fig. 3. The results show that our joint learning can simultaneously reduce the cardinality and classification errors in these examples.

Conclusion

We proposed a framework to jointly learn and predict a set's cardinality and state distributions by modelling both distribu-

[2]F1-score is calculated as the harmonic mean of precision and recall.

Table 1: Quantitative results for multi-label image classification on the PASCAL VOC dataset.

Classifier	Eval.	C-P	C-R	C-F1	O-P	O-R	O-F1	I-P	I-R	I-F1
Softmax	k=$k^*(1)$	88.2	65.4	75.1	91.3	59.2	71.8	91.3	69.8	79.1
BCE	k=$k^*(1)$	88.7	58.6	70.5	92.2	59.8	72.5	92.2	70.1	79.6
DS (BCE-NB) (Rezatofighi et al. 2017)	k=m^*	76.8	74.8	75.8	80.6	76.7	78.6	83.4	81.9	82.6
DS (BCE-DC)	k=m^*	77.1	75.2	76.2	81.0	77.1	79.0	83.9	82.1	83.0
JDS (BCE-DC)	k=m^*	83.5	74.4	**78.7**	85.5	77.9	**81.5**	87.6	82.8	**85.1**

c

GT:	person, skateboard	person, potted plant, remote	person, chair, handbag, umbrella	person, chair, dining table, bottle, cup, bowl, knife, fork, spoon, pizza
JDS: (Ours)	person, skateboard	person, potted plant, remote	person, chair, handbag, umbrella	person, chair, dining table, bottle, cup, bowl, knife, fork, spoon, wine glass
DS (DC):	person, baseball glove	person, chair, remote	person, chair, dining table, umbrella	person, chair, dining table, cup, bowl, book, knife, fork, wine glass
DS (NB):	person, baseball glove, car	person, potted plant, chair, vase, remote	person, chair, dining table, bench, potted plant, umbrella	person, chair, dining table, cup, bowl, book, knife, fork

Figure 3: Qualitative comparison between our proposed joint deep set network (JDS) and the deep set networks with Dirichlet-Categorical (DS (DC)) and Negative Binomial (DS (NB)) as the cardinality loss. For each image, the ground truth tags and the predictions for our JDS and the two baselines are denoted below. False positives are highlighted in red. Our JDS approach reduces both cardinality and classification error.

Table 2: Quantitative results for multi-label image classification on the MS COCO dataset.

Classifier	Eval.	C-P	C-R	C-F1	O-P	O-R	O-F1	I-P	I-R	I-F1
Softmax	k=$k^*(3)$	58.6	57.6	58.1	60.7	63.3	62.0	60.7	74.7	67.0
BCE	k=$k^*(3)$	56.2	60.1	58.1	61.6	64.2	62.9	61.6	75.3	67.8
CNN-RNN (Wang et al. 2016)	k=$k^*(3)$	66.0	55.6	60.4	69.2	66.4	67.8	–	–	–
DS (Softmax-NB) (Rezatofighi et al. 2017)	k=m^*	68.2	59.9	63.8	68.8	67.4	68.1	74.3	72.6	73.5
DS (BCE-NB) (Rezatofighi et al. 2017)	k=m^*	66.5	62.9	64.6	70.1	68.7	69.4	75.2	73.6	74.4
DS (BCE-DC)	k=m^*	68.0	61.7	64.7	72.4	67.1	69.6	76.0	73.3	74.6
JDS (BCE-DC)	k=m^*	70.2	61.5	**65.5**	74.0	67.6	**70.7**	77.9	73.4	**75.6**

tions using the same set of weights. This approach not only significantly reduces the number of learnable parameters, but also helps to model both distributions more accurately. We have demonstrated the effectiveness of this approach on multi-class image classification, outperforming previous state of the art on standard datasets. The main limitation of our framework is that we do not include the complexity of permutation invariance of sets in the learning step. Therefore, our method is only applicable to set problems that do not rely on permutation invariance during training, such as image tagging. In future, we plan to overcome this limitation by incorporating permutation variables during training procedure. Another potential avenue could be to exploit the Bayesian nature of the model to include uncertainty as opposed to relying on the MAP estimation alone.

Acknowledgments. This research was supported by the Australian Research Council through the Centre of Excellence in Robotic Vision, CE140100016, and through Laureate Fellowship FL130100102 to IDR.

References

Adams, R. P.; Murray, I.; and MacKay, D. J. 2009. Tractable nonparametric Bayesian inference in Poisson processes with Gaussian process intensities. In *ICML*, 9–16.

Baddeley, A.; Bárány, I.; and Schneider, R. 2007. Spatial point processes and their applications. *Stochastic Geometry* 1–75.

Blei, D. M.; Ng, A. Y.; and Jordan, M. I. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022.

Chiu, S. N.; Stoyan, D.; Kendall, W. S.; and Mecke, J. 2013. *Stochastic geometry and its applications*. John Wiley & Sons.

Daley, D. J., and Vere-Jones, D. 2007. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media.

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

Gong, Y.; Jia, Y.; Leung, T.; Toshev, A.; and Ioffe, S. 2013a. Deep convolutional ranking for multilabel image annotation. *arXiv preprint arXiv:1312.4894*.

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

Hannah, L. A.; Blei, D. M.; and Powell, W. B. 2011. Dirichlet process mixtures of generalized linear models. *Journal of Machine Learning Research* 12(Jun):1923–1953.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Hinton, G.; Deng, L.; Yu, D.; Dahl, G. E.; Mohamed, A.-r.; Jaitly, N.; Senior, A.; Vanhoucke, V.; Nguyen, P.; Sainath, T. N.; et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine* 29(6):82–97.

Johnson, J.; Karpathy, A.; and Fei-Fei, L. 2016. DenseCap: Fully convolutional localization networks for dense captioning. In *CVPR*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*, 1097–1105.

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

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

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Moller, J., and Waagepetersen, R. P. 2003. *Statistical inference and simulation for spatial point processes*. CRC Press.

Muandet, K.; Fukumizu, K.; Dinuzzo, F.; and Schölkopf, B. 2012. Learning from distributions via support measure machines. In *NIPS*.

Oliva, J.; Póczos, B.; and Schneider, J. 2013. Distribution to distribution regression. In *ICML*, 1049–1057.

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

Rezatofighi, S. H.; Kumar BG, V.; Milan, A.; Abbasnejad, E.; Dick, A.; Reid, I.; et al. 2017. DeepSetNet: Predicting sets with deep neural networks. In *ICCV*.

Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; Berg, A. C.; and Fei-Fei, L. 2015. Imagenet large scale visual recognition challenge. *IJCV* 115(3):211–252.

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

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

Tran, N.-Q.; Vo, B.-N.; Phung, D.; and Vo, B.-T. 2016. Clustering for point pattern data. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, 3174–3179. IEEE.

Vinyals, O.; Bengio, S.; and Kudlur, M. 2015. Order matters: Sequence to sequence for sets. *ICLR*.

Vinyals, O.; Fortunato, M.; and Jaitly, N. 2015. Pointer networks. In *NIPS*, 2692–2700.

Vo, B.-N.; Tran, N.-Q.; Phung, D.; and Vo, B.-T. 2016. Model-based classification and novelty detection for point pattern data. In *ICPR*.

Vo, B.-N.; Phung, D.; Tran, Q. N.; and Vo, B.-T. 2017. Model-based multiple instance learning. *arXiv preprint arXiv:1703.02155*.

Wang, J.; Yang, Y.; Mao, J.; Huang, Z.; Huang, C.; and Xu, W. 2016. CNN-RNN: A unified framework for multi-label image classification. In *CVPR*.

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

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Póczos, B.; Salakhutdinov, R.; and Smola, A. J. 2017. Deep sets. In *NIPS*.