Deep Learning from Crowds

Filipe Rodrigues, Francisco C. Pereira

Dept. of Management Engineering, Technical University of Denmark
Bygning 116B, 2800 Kgs. Lyngby, Denmark
rodr@dtu.dk, camara@dtu.dk

Abstract

Over the last few years, deep learning has revolutionized the field of machine learning by dramatically improving the state-of-the-art in various domains. However, as the size of supervised artificial neural networks grows, typically so does the need for larger labeled datasets. Recently, crowdsourcing has established itself as an efficient and cost-effective solution for labeling large sets of data in a scalable manner, but it often requires aggregating labels from multiple noisy contributors with different levels of expertise. In this paper, we address the problem of learning deep neural networks from crowds. We begin by describing an EM algorithm for jointly learning the parameters of the network and the reliabilities of the annotators. Then, a novel general-purpose crowd layer is proposed, which allows us to train deep neural networks end-to-end, directly from the noisy labels of multiple annotators, using only backpropagation. We empirically show that the proposed approach is able to internally capture the reliability and biases of different annotators and achieve new state-of-the-art results for various crowdsourced datasets across different settings, namely classification, regression and sequence labeling.

Introduction

In the last decade, deep learning has made major advances in solving artificial intelligence problems in different domains such as speech recognition, visual object recognition, object detection and machine translation (Schmidhuber 2015). This success is often attributed to its ability to discover intricate structures in high-dimensional data (LeCun, Bengio, and Hinton 2015), thereby making it particularly well suited for tackling complex tasks that are often regarded as characteristic of humans, such as vision, speech and natural language understanding. However, typically, a key requirement for learning deep representations of complex high-dimensional data is large sets of labeled data. Unfortunately, in many situations this data is not readily available, and humans are required to manually label large collections of data.

On the other hand, in recent years, crowdsourcing has established itself as a reliable solution to annotate large collections of data. Indeed, crowdsourcing platforms like Amazon Mechanical Turk[1] and Crowdflower[2] have proven to be an

[1] http://www.mturk.com

[2] http://crowdflower.com

efficient and cost-effective way for obtaining labeled data (Snow et al. 2008; Buhrmester, Kwang, and Gosling 2011), especially for the kind of human-like tasks, such as vision, speech and natural language understanding, for which deep learning methods have been shown to excel. Even in fields like medical imaging, crowdsourcing is being used to collect the large sets of labeled data that modern data-savvy deep learning methods enjoy (Greenspan, van Ginneken, and Summers 2016; Albarqouni et al. 2016; Guan et al. 2017). However, while crowdsourcing is scalable enough to allow labeling datasets that would otherwise be impractical for a single annotator to handle, it is well known that the noise associated with the labels provided by the various annotators can compromise practical applications that make use of such type of data (Sheng, Provost, and Ipeirotis 2008; Donmez and Carbonell 2008). Thus, it is not surprising that a large body of the recent machine learning literature is dedicated to mitigating the effects of the noise and biases inherent to such heterogeneous sources of data (e.g. Yan et al. (2014); Albarqouni et al. (2016); Guan et al. (2017)).

When learning deep neural networks from the labels of multiple annotators, typical approaches rely on some sort of label aggregation mechanisms prior to training. In classification settings, the simplest and most common approach is to use majority voting, which naively assumes that all annotators are equally reliable. More advanced approaches, such as the one proposed in (Dawid and Skene 1979) and other variants (e.g. Ipeirotis, Provost, and Wang (2010); Whitehill et al. (2009)) jointly model the unknown biases of the annotators and their answers as noisy versions of some latent ground truth. Despite their improved ground truth estimates over majority voting, recent works have shown that jointly learning the classifier model and the annotators noise model using EM-style algorithms generally leads to improved results (Raykar et al. 2010; Albarqouni et al. 2016).

In this paper, we begin by describing an EM algorithm for learning deep neural networks from crowds in multi-class classification settings, highlighting its limitations. Then, a novel *crowd layer* is proposed, which allows us to train neural networks end-to-end, directly from the noisy labels of multiple annotators, using only backpropagation. This alternative approach not only allows us to avoid the additional computational overhead of EM, but also leads to a general-purpose framework that generalizes trivially beyond classi-

fication settings. Empirically, the proposed crowd layer is shown to be able to automatically distinguish the good from the unreliable annotators and capture their individual biases, thus achieving new state-of-the-art results in real data from Amazon Mechanical Turk for image classification, text regression and named entity recognition. As our experiments show, when compared to the more complex EM-based approaches and other approaches from the state of the art, the crowd layer is able to achieve comparable or, in many cases, significantly superior results.

Related work

The increasing popularity of crowdsourcing as a way to label large collections of data in an inexpensive and scalable manner has led to much interest of the machine learning community in developing methods to address the noise and trustworthiness issues associated with it. In this direction, one of the key early contributions is the work of Dawid and Skene (1979), who proposed an EM algorithm to obtain point estimates of the error rates of patients given repeated but conflicting responses to medical questions. This work was the basis for many other variants for aggregating labels from multiple annotators with different levels of expertise, such as the one proposed in (Whitehill et al. 2009), which further extends Dawid and Skene's model by also accounting for item difficulty in the context of image classification. Similarly, Ipeirotis et al. (2010) propose using Dawid and Skene's approach to extract a single quality score for each worker that allows to prune low-quality workers. The approach proposed in our paper contrast with this line of work, by allowing neural networks to be trained directly on the noisy labels of multiple annotators, thereby avoiding the need to resort to prior label aggregation schemes.

Despite the generality of label aggregation approaches described above, which can be used in combination with any type of machine learning algorithm, they are sub-optimal when compared to approaches that also jointly learn the classifier itself. One of the most prominent works in this direction is the one of Raykar et al. (2010), who proposed an EM algorithm for jointly learning the levels of expertise of different annotators and the parameters of a logistic regression classifier, by modeling the ground truth labels as latent variables. This idea was later extended to other types of models such as Gaussian process classifiers (Rodrigues, Pereira, and Ribeiro 2014), supervised latent Dirichlet allocation (Rodrigues et al. 2017) and, recently, to convolutional neural networks with softmax outputs (Albarqouni et al. 2016). In this paper, we begin by describing a generalization of the approach in (Albarqouni et al. 2016) to multiclass settings, highlighting some of the technical difficulties associated with it. Then, a novel type of neural network layer is proposed, which allows the training of deep neural networks directly from the noisy labels of multiple annotators using pure backpropagation. This contrasts with most of works in the literature, which rely on more complex iterative procedures based on EM. Furthermore, the simplicity of the proposed approach allows for straightforward extensions to regression and structured prediction problems.

Recently, Guan et al. (2017) also proposed an approach for training deep neural networks that exploits information about the annotators. The idea is to model the multiple experts individually in the neural network and then, while keeping their predictions fixed, independently learning averaging weights for combining them using backpropagation. Like our proposed approach, this two-stage procedure does not require an EM algorithm to estimate the annotators weights. However, while our approach has the ability to capture the biases of the different annotators (e.g. confusing class 2 with class 4) and correct them, the approach in (Guan et al. 2017) only learns how to combine the predicted answers of multiple annotators by weighting them differently. Moreover, its two-stage learning procedure increases the computation complexity of training, whereas in our proposed approach is kept the same. Lastly, while the work in (Guan et al. 2017) focuses only on classification, we consider regression and structured prediction problems as well.

Regarding applications areas for multiple-annotator learning, some of the most popular ones are: image classification (Smyth et al. 1995; Welinder et al. 2010), computer-aided diagnosis/radiology (Raykar et al. 2010; Greenspan, van Ginneken, and Summers 2016), object detection (Su, Deng, and Fei-Fei 2012), text classification (Rodrigues et al. 2017), natural language processing (Snow et al. 2008) and speech-related tasks (Parent and Eskenazi 2011). In this paper, we will use data from some of these areas to evaluate different approaches. Given that these are precisely some of the areas that have seen the most dramatic improvements due to recent contributions in deep learning (LeCun, Bengio, and Hinton 2015; Schmidhuber 2015), developing novel efficient algorithms for learning deep neural networks from crowds is of great importance to the field.

EM algorithm for deep learning from crowds

Let $\mathcal{D} = \{\mathbf{x}_n, \mathbf{y}_n\}_{n=1}^N$ be a dataset of size N, where for each input vector $\mathbf{x}_n \in \mathbb{R}^D$ we are given a vector of crowd-sourced labels $\mathbf{y}_n = \{y_n^r\}_{r=1}^R$, with y_n^r representing the label provided by the rth annotator in a set of R annotators. Following the ideas in (Raykar et al. 2010; Yan et al. 2014), we shall assume the existence of a latent true class z_n whose value is, in this particular case, determined by a softmax output layer of a deep neural network parameterized by $\mathbf{\Theta}$, and that each annotator then provides a noisy version of z_n according to $p(y_n^r|z_n, \mathbf{\Pi}^r) = Multinomial(y_n^r|\boldsymbol{\pi}_{z_n}^r)$. This formulation corresponds to keeping a per-annotator confusion matrix $\mathbf{\Pi}^r = (\boldsymbol{\pi}_1^r, \dots, \boldsymbol{\pi}_C^r)$ to model their expertise, where C denotes the number of classes. Further assuming that annotators provide labels independently of each other, we can write the complete-data likelihood as

$$p(\mathcal{D}, \mathbf{z}|\mathbf{\Theta}, \{\mathbf{\Pi}^r\}_{r=1}^R) = \prod_{n=1}^N p(z_n|\mathbf{x}_n, \mathbf{\Theta}) \prod_{r=1}^R p(y_n^r|z_n, \mathbf{\Pi}^r).$$

Based on this formulation, we can derive an Expectation-Maximization (EM) algorithm for jointly learning the reliabilities of the annotators $\mathbf{\Pi}^r$ and the parameters of the neural network $\mathbf{\Theta}$. The expected-value of the complete-data log-

likelihood under a current estimate of the posterior distribution over latent variables $q(z_n)$ is given by

$$\mathbb{E}\left[\ln p(\mathcal{D}, \mathbf{z} | \boldsymbol{\Theta}, \boldsymbol{\Pi}^1, \ldots, \boldsymbol{\Pi}^R)\right] =$$

$$\sum_{n=1}^{N} \sum_{z_n} q(z_n) \ln \left(p(z_n | \mathbf{x}_n, \boldsymbol{\Theta}) \prod_{r=1}^{R} p(y_n^r | z_n, \boldsymbol{\Pi}^r) \right), \quad (1)$$

where the posterior $q(z_n)$ is obtained by making use of Bayes' theorem given the previous estimate of the model parameters $\{\boldsymbol{\Theta}_{\text{old}}, \boldsymbol{\Pi}_{\text{old}}^1, \ldots, \boldsymbol{\Pi}_{\text{old}}^R\}$, yielding

$$q(z_n = c) \propto p(z_n = c | \mathbf{x}_n, \boldsymbol{\Theta}_{\text{old}}) \prod_{r=1}^{R} p(y_n^r | z_n = c, \boldsymbol{\Pi}_{\text{old}}^r).$$

This corresponds to the E-step of EM. In the M-step, we find the new maximum likelihood for the model parameters. The update for the annotators' reliability parameters is given by

$$\pi_{c,l}^r = \frac{\sum_{n=1}^{N} q(z_n = c) \, \mathbb{I}(y_n^r = l)}{\sum_{n=1}^{N} q(z_n = c)},$$

where $\mathbb{I}(y_n^r = l)$ is an indicator function that takes the value 1 when $y_n^r = l$, and zero otherwise. In practice, since crowd annotators typically only label a small portion of the data, it is particularly important to carefully impose Dirichlet priors on each $\boldsymbol{\pi}_c^r$ and compute MAP estimates instead, in order to avoid numerical issues.

As for estimating the parameters of the deep neural network $\boldsymbol{\Theta}$, we follow the approach in (Albarqouni et al. 2016) and use the noise-adjusted ground-truth estimates $q(z_n)$ to backpropagate the error through the network using standard stochastic optimization techniques such as stochastic gradient descent (SGD) or Adam (Kingma and Ba 2014). Kindly notice how this raises the important question of how to schedule the EM steps. If we perform one EM iteration per mini-batch, we risk not having enough evidence to estimate the annotators reliabilities. On the other hand, if we run SGD or Adam until convergence, then the computational overhead of EM becomes very large. In practice, we found that, typically, one EM iteration per training epoch provides good computational efficiency without compromising accuracy. However, this seems to vary among different datasets, thus making it hard to tune in practice.

One key fundamental aspect for the development of this EM approach was the probabilistic interpretation of the softmax output layer of deep neural networks for classification. Unfortunately, such probabilistic interpretation is typically not available when considering, for example, continuous output variables, thereby making it more difficult to generalize this approach to regression problems. Furthermore, notice that if the target variable is a sequence (or any other structured prediction output), then the marginalization over the latent variables in (1) quickly become intractable, as the number of possible label sequences to sum over grows exponentially with the length of the sequence.

Crowd layer

In this section, we propose the *crowd layer*: a special type of network layer that allows us to train deep neural networks directly from the noisy labels of multiple annotators,

thereby avoiding some of the aforementioned limitations of EM-based approaches for learning from crowds. The intuition is rather simple. The crowd layer takes as input what would normally be the output layer of a deep neural network (e.g. softmax for classification, or linear for regression), and learns an annotator-specific mapping from the output layer to the labels of the different annotators in the crowd that captures the annotator reliabilities and biases. In this way, the former output layer becomes a bottleneck layer that is shared among the different annotators. Figure 1 illustrates this bottleneck structure in the context of a simple convolutional neural network for classification problems with 4 classes and R annotators.
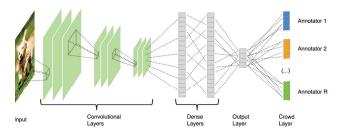

Figure 1: Bottleneck structure for a CNN for classification with 4 classes and R annotators.

The idea is then that when using the labels of a given annotator to propagate errors through the whole neural network, the crowd layer adjusts the gradients coming from the labels of that annotator according to his/her reliability by scaling them and adjusting their bias. In doing so, the bottleneck layer of the network now receives adjusted gradients from the different annotators' labels, which it aggregates and backpropagates further through the rest of the network. As it turns out, through this crowd layer, the network is able to account for unreliable annotators and even correct systematic biases in their labeling. Moreover, all of that can be done naturally within the backpropagation framework.

Formally, let $\boldsymbol{\sigma}$ be the output of a deep neural network with an arbitrary structure. Without loss of generality, we shall assume the vector $\boldsymbol{\sigma}$ to correspond to the output of a softmax layer, such that σ_c corresponds to the probability of the input instance belonging to class c. The activation of the crowd layer for each annotator r is then defined as $\mathbf{a}^r = f_r(\boldsymbol{\sigma})$, where f_r is an annotator-specific function, and the output of the crowd layer simply as the softmax of the activations $o_c^r = e^{a_c^r} / \sum_{l=1}^{C} e^{a_l^r}$.

The question is then how to define the function mapping $f_r(\boldsymbol{\sigma})$. In the experiments section, we study different alternatives. For classification problems a reasonable assumption is to consider a matrix transformation, such that $f_r(\boldsymbol{\sigma}) = \mathbf{W}^r \boldsymbol{\sigma}$, where $\mathbf{W}^r$ is an annotator-specific matrix. Given a cost function $E(\mathbf{o}^r, y^r)$ between the expected output of the rth annotator and its actual label y^r, we can compute the gradients $\partial E / \partial \mathbf{a}^r$ at the activation $\mathbf{a}^r$ for each annotator

and backpropagate them to the bottleneck layer, leading to

$$\frac{\partial E}{\partial \boldsymbol{\sigma}} = \sum_{r=1}^{R} \mathbf{W}^r \frac{\partial E}{\partial \mathbf{a}^r}.$$

The gradient vector at the bottleneck layer then naturally becomes a weighted sum of gradients according to the labels of the different annotators. Moreover, if the annotator is likely to mislabel class c as class l (annotation bias), then the matrix $\mathbf{W}^r$ can actually adjust the gradients accordingly. The problem of missing labels from some of the annotators can be easily addressed by setting their gradient contributions to zero. As for estimating the annotator weights $\{\mathbf{W}^r\}_{r=1}^{R}$, since they parameterize the mapping from the output of the bottleneck layer $\boldsymbol{\sigma}$ to the annotators labels $\{\mathbf{o}^r\}_{r=1}^{R}$, they can be estimated using standard stochastic optimization techniques such as SGD or Adam (Kingma and Ba 2014). Once the network is trained, the crowd layer can be removed, thus exposing the output of bottleneck layer $\boldsymbol{\sigma}$, which can readily be used to make predictions for unseen instances.

An obvious concern with the approach described above is identifiability. Therefore, it is important to not over-parametrize $f_r(\boldsymbol{\sigma})$, since adding parameters beyond necessary can make the output of the bottleneck layer $\boldsymbol{\sigma}$ lose its interpretability as a shared estimated ground truth. Another important aspect is parameter initialization. In our experiments, we found that the best practice is to initialize the crowd layer with identities, i.e. zeros for additive parameters, ones for scalar parameters, identity matrix for multiplicative matrices, etc. An alternative solution is to use regularization to force the parameters of the crowd layer to be close to identities. However, in some cases this might be an undesirable property. For example, if we consider a very biased annotator, then we do not wish to force the matrix $\mathbf{W}^r$ to be close to the identity matrix. Based on our experiments, the initialization alternative provides the best results. Lastly, it should be noted that, as with EM-based approaches, there is an implicit assumption that random or adversarial annotators do not constitute a vast majority (which generally holds in practice), in which case the crowd layer would not perform better than a random predictor.

A particularly important aspect to note, is that the framework described above is quite general. For example, it can be straightforwardly applied to sequence labeling problems without further changes, or be adapted to regression problems by considering univariate scalar and bias parameters per annotator in the crowd layer.

Experiments

The proposed crowd layer (CL) was implemented as a new type of layer in Keras (Chollet 2015), so that using it in practice requires only a single line of code. Source code, datasets and demos for all experiments are provided at: http://www.fprodrigues.com/.

Image classification

We begin by evaluating the proposed crowd layer in a more controlled setting, by using simulated annotators with different levels of expertise on a large image classification dataset consisting of 25000 images of dogs and cats from (Kaggle 2013), where the goal is to distinguish between the two species. Let the dog and cat classes be represented by 1 and 0, respectively. Since this is a binary classification task, we can easily simulate annotators with different levels of expertise by assigning them individual sensitivities α^r and specificities β^r, and sampling their answers from a Bernoulli distribution with parameter α^r if the true label is 1, and from a Bernoulli distribution with parameter β^r otherwise. Using this procedure, we simulated a challenging scenario with 5 annotators with the following values of $\alpha^r = [0.6, 0.9, 0.5, 0.9, 0.9]$ and $\beta^r = [0.7, 0.8, 0.5, 0.2, 0.9]$.

For this particular problem we used a fairly standard CNN architecture with 4 convolutional layers with 3x3 patches, 2x2 max pooling and ReLU activations. The output of the convolutional layers is then fed to a fully-connected (FC) layer with 128 ReLU units and finally goes to an output layer with a softmax activation. We use batch normalization (Ioffe and Szegedy 2015) and apply 50% dropout between the FC and output layers. The proposed approach further adds a crowd layer on top of the softmax output layer during training. The base architecture was selected from a set of possible configurations using the true labels by optimizing the accuracy on a validation set (consisting of 20% of the train set) through random search. It is important to note that it is supposed to be representative of a set of typical approaches for image classification rather than being the single best possible architecture in the literature for this particular dataset. Furthermore, our main interest in this paper is the contribution of the crowd layer to the training of the neural network.

The proposed CNN with a crowd layer (referred to as "DL-CL") is compared with: the multi-annotator approach from (Rodrigues et al. 2017) based on supervised latent Dirichlet allocation - "MA-sLDA"; a CNN trained on the result of (hard) majority voting - "DL-MV"; a CNN trained on the output of the label aggregation approach proposed by Dawid and Skene (1979) - "DL-DS"; a CNN using the EM approach described earlier - "DL-EM"; a CNN using the "Doctor Net" approach from (Guan et al. 2017) - "DL-DN", which consists on training a CNN to predict the labels of the multiple annotators and then combining their predictions using majority voting; and, lastly, a CNN using the "Weighted Doctor Net" approach from (Guan et al. 2017) - "DL-WDN", which is the best performing variant according to the original paper. This approach is similar to "DL-DN" but additionally learns how to weight the predictions of the different annotators. Kindly see (Guan et al. 2017) for further details. As a reference point, we also compare with a CNN trained on true labels - "DL-TRUE". We consider 3 variants of the proposed crowd layer (CL) with different annotator-specific functions f_r with increasing number of parameters: a vector of per-class weights $\mathbf{w}^r$, such that $f_r(\boldsymbol{\sigma}) = \mathbf{w}^r \odot \boldsymbol{\sigma}$ (referred to as "VW"); a vector of per-class biases $\mathbf{b}^r$, such that $f_r(\boldsymbol{\sigma}) = \boldsymbol{\sigma} + \mathbf{b}^r$ ("VB"); and a version with a matrix of weights $\mathbf{W}^r$, such that $f_r(\boldsymbol{\sigma}) = \mathbf{W}^r \boldsymbol{\sigma}$ ("MW"). In our experiments, we found that for approaches with more parameters than MW, such as $f_r(\boldsymbol{\sigma}) = \mathbf{W}^r \boldsymbol{\sigma} + \mathbf{b}^r$, identifiability issues start to occur.

Table 1: Accuracy results for classification datasets: Dogs vs. Cats and LabelMe.

Method	Dogs vs. Cats	LabelMe (MTurk)
MA-sLDAc (Rodrigues et al. 2017)	-	78.120 ($\pm$ 0.397)
DL-MV	71.377 ($\pm$ 1.123)	76.744 ($\pm$ 1.208)
DL-DS (Dawid and Skene 1979)	76.750 ($\pm$ 1.282)	80.792 ($\pm$ 1.066)
DL-EM (Albarqouni et al. 2016)	80.184 ($\pm$ 1.454)	82.677 ($\pm$ 0.981)
DL-DN (Guan et al. 2017)	79.005 ($\pm$ 1.347)	81.888 ($\pm$ 1.114)
DL-WDN (Guan et al. 2017)	76.822 ($\pm$ 2.838)	82.410 ($\pm$ 0.783)
DL-CL (VW)	79.534 ($\pm$ 1.064)	81.051 ($\pm$ 0.899)
DL-CL (VW+B)	79.688 ($\pm$ 1.406)	81.886 ($\pm$ 0.893)
DL-CL (MW)	80.265 ($\pm$ 1.230)	83.151 ($\pm$ 0.877)
DL-TRUE	84.912 ($\pm$ 1.248)	90.038 ($\pm$ 0.652)

Table 2: Results for MovieReviews (MTurk) dataset.

Method	MAE	RMSE	R^2
MA-sLDAr (Rodrigues et al. 2017)	-	-	35.553 ($\pm$ 1.282)
DL-MEAN	1.215 ($\pm$ 0.048)	1.498 ($\pm$ 0.050)	31.496 ($\pm$ 4.690)
DL-EM	1.201 ($\pm$ 0.046)	1.482 ($\pm$ 0.048)	32.974 ($\pm$ 4.457)
DL-DN (Guan et al. 2017)	1.270 ($\pm$ 0.021)	1.549 ($\pm$ 0.022)	26.775 ($\pm$ 2.102)
DL-WDN (Guan et al. 2017)	1.261 ($\pm$ 0.016)	1.541 ($\pm$ 0.018)	27.597 ($\pm$ 1.763)
DL-CL (S)	1.228 ($\pm$ 0.041)	1.508 ($\pm$ 0.044)	30.560 ($\pm$ 4.101)
DL-CL (S+B)	1.163 ($\pm$ 0.031)	1.440 ($\pm$ 0.033)	37.086 ($\pm$ 2.407)
DL-CL (B)	1.130 ($\pm$ 0.025)	1.411 ($\pm$ 0.028)	39.276 ($\pm$ 2.374)
DL-TRUE	1.050 ($\pm$ 0.029)	1.330 ($\pm$ 0.036)	45.983 ($\pm$ 2.895)

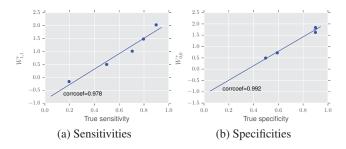(a) Sensitivities   (b) Specificities

Figure 2: Comparison between the true sensitivities and specificities of the annotators and the diagonal elements of their weight matrices $\mathbf{W}^r$ for the Dogs vs. Cats dataset.

Of the 25000 images in the Dogs vs Cats dataset, 50% were used for training and the remaining for testing the different approaches. In order to account for the effect of the random initialization that is used for most of the parameters in the network, we performed 30 executions of all approaches and report their average accuracies in Table 1. We can immediately verify that both the EM-based and the crowd layer (CL) approaches significantly outperform the majority voting (DL-MV) and Dawid & Skene (DL-DS) baselines, thus demonstrating the gain of learning from the answers of multiple annotators directly rather than relying on aggregation schemes prior to training. As for the DL-

DN and DL-WDN approaches from (Guan et al. 2017), we can observe that, although they also outperform the DL-MV and DL-DS baselines, their accuracy is inferior to that of the proposed DL-CL, which can be explained by the fact that DL-DN and DL-WDN are unable to correct the annotators' biases (e.g. confusing class 2 with class 4). Furthermore, it important to recall that due to two-stage procedure of DL-WDN, its computational time can be significantly higher than DL-CL. Regarding the different variants of the proposed crowd layer, we can verify that the MW approach is the one that gives the best average accuracy. In order to better understand what the MW approach is doing, we inspected the weight matrices $\mathbf{W}^r$ of each annotator r. Figure 2 shows the relationship between the diagonal elements of $\mathbf{W}^r$ and the true sensitivities and specificities of the corresponding annotators, highlighting the strong linear correlation between the two. This evidences that the proposed crowd layer is able to internally represent the reliabilities of the annotators.

Having verified that the crowd layer was performing well for simulated annotators, we then moved on to evaluating it in real data from Amazon Mechanical Turk (AMT). For this purpose, we used the image classification dataset from (Rodrigues et al. 2017) adapted from part of the LabelMe data (Russell et al. 2008), whose goal is to classify images according to 8 classes: "highway", "inside city", "tall building", "street", "forest", "coast", "mountain" or "open coun-
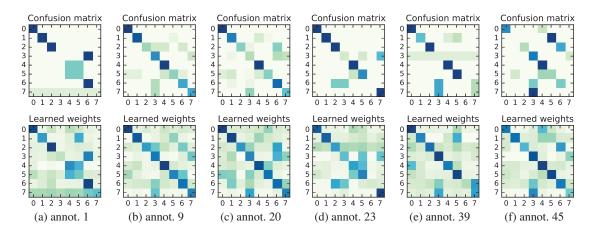
Figure 3: Comparison between the learned weight matrices $\mathbf{W}^r$ and the corresponding true confusion matrices.

try". It consists of a total of 2688 images, where 1000 of them were used to obtain labels from multiple annotators from Amazon Mechanical Turk. Each image was labeled by an average of 2.547 workers, with a mean accuracy of 69.2%. The remaining 1688 images were using for evaluating the different approaches.

Since the training set is rather small, we use the pretrained CNN layers of the VGG-16 deep neural network (Simonyan and Zisserman 2014) and apply only one FC layer (with 128 units and ReLU activations) and one output layer on top, using 50% dropout. The last column of Table 1 shows the obtained results. We can once more verify that DL-EM, DL-WDN and DL-CL approaches outperform the majority voting and Dawid & Skene baselines, and also the probabilistic approaches proposed in (Rodrigues et al. 2017) based on supervised latent Dirichlet allocation (sLDA), being the proposed crowd layer (DL-CL) the approach that again gives the best results. However, unlike for the Dogs vs. Cats dataset, the differences between the different function mappings f_r for the crowd layer (CL) become more evident. This can be justified by the ability of the MW version to be able to model the biases of the annotators. Indeed, if we compare the learned weight matrices $\mathbf{W}^r$ with the respective true confusion matrices of the annotators, we can notice how they resemble each other. Figure 3 shows this comparison for 6 annotators, where the color intensity of the cells increases with the relative magnitude of the value, thus demonstrating that the crowd layer is able to learn the labeling patterns of the annotators.

Text regression

As previously mentioned, one of the key advantages of the proposed crowd layer is its straightforward extension to other types of target variables. In this section, we consider a regression problem based on the dataset also introduced in (Rodrigues et al. 2017). This dataset consists of 5006 movie reviews, where the goal is to predict the rating given to the movie (on a scale of 1 to 10) based on the text of the review. Using AMT, the authors collected an average of 4.96 answers from a pool of 137 workers

for 1500 movie reviews. The remaining 3506 reviews were used for testing. Letting the (continuous) output of the bottleneck layer be denoted μ, we considered 3 variants of the proposed crowd layer with different annotator-specific functions f_r: a per-annotator scale parameter s^r, such that $f_r(\mu) = s^r\mu$ (referred to as "S"); a per-annotator bias parameter b^r, such that $f_r(\mu) = \mu + b^r$ ("B"); and a version with both: $f_r(\mu) = s^r\mu + b^r$ ("S+B"). The base neural network architecture used for this problem consists of a 3x3 convolutional layer with 128 features and 5x5 max pooling, a 5x5 convolutional layer with 128 features and 5x5 max pooling, and a FC layer with 32 hidden units. All layers, except for the output one, use ReLU activations.

The proposed DL-CL is compared with: a neural network trained on the mean answer of the annotators (DL-MEAN) and the approach from (Rodrigues et al. 2017) based on supervised LDA. In order to make the baselines even more competitive, we further propose a new variant of the EM algorithm described earlier that follows the same approach as the extension proposed in (Raykar et al. 2010) for regression problems. This approach assumes the following model for the annotators answers given the ground truth: $p(y_n^r|z_n) = \mathcal{N}(y_n^r|z_n, 1/\lambda^r)$. Although the formulation in (Raykar et al. 2010) relies on the probabilistic interpretation of the linear regression model to develop an EM algorithm for learning, we can nevertheless adapt the resultant EM algorithm by replacing the linear regression model with a deep neural network. The final iterative procedure then alternates between computing the adjusted ground truth (E-step) and re-estimating the neural network and the annotators' parameters (M-step). Finally, although Guan et al. (2017) do not discuss extensions to regression, we also developed variants of DL-DN and DL-WDN for continuous output variables. For the DL-WDN approach, we considered different weighting functions for combining the answers of the multiple annotators, namely: a single weight per annotator, a single bias, or both. We experimented with the different alternatives and found that using a per-annotator bias for combining the answers of the multiple annotators gives the best results.

Table 2 shows the obtained results for 30 runs of the dif-

Table 3: Results for CoNLL-2003 NER (MTurk) dataset.

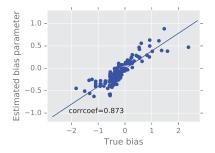| Method	Precision	Recall	F1
CRF-MA (Rodrigues, Pereira, and Ribeiro 2013)	0.494	0.856	0.626
DL-MV	0.664 ($\pm$ 0.017)	0.464 ($\pm$ 0.021)	0.546 ($\pm$ 0.014)
DL-EM	0.679 ($\pm$ 0.012)	0.499 ($\pm$ 0.010)	0.575 ($\pm$ 0.008)
DL-DN (Guan et al. 2017)	0.723 ($\pm$ 0.009)	0.459 ($\pm$ 0.014)	0.562 ($\pm$ 0.012)
DL-WDN (Guan et al. 2017)	0.611 ($\pm$ 0.063)	0.480 ($\pm$ 0.058)	0.534 ($\pm$ 0.042)
DL-CL (VW)	0.709 ($\pm$ 0.013)	0.472 ($\pm$ 0.020)	0.566 ($\pm$ 0.016)
DL-CL (VW+B)	0.603 ($\pm$ 0.013)	0.609 ($\pm$ 0.012)	0.606 ($\pm$ 0.007)
DL-CL (MW)	0.660 ($\pm$ 0.018)	0.593 ($\pm$ 0.013)	0.624 ($\pm$ 0.010)
DL-TRUE	0.711 ($\pm$ 0.013)	0.740 ($\pm$ 0.009)	0.725 ($\pm$ 0.008)

Figure 4: Relationship between the learned b^r parameters and the true biases of the annotators.

ferent approaches, where we verify that the proposed crowd layer, particularly the "B" variant, significantly outperforms all the other methods. In order to better understand what the crowd layer in the "B" variant is doing, we plotted learned b^r values in comparison with the true biases of the annotators, computed as the average difference between their answers and the ground truth. Figure 4 shows this comparison, in which we can verify that the learned values of b^r are highly correlated with the true biases of the annotators, thus showing that crowd layer is able to account for annotator bias when learning from the noisy labels of multiple annotators.

Named entity recognition

Lastly, we evaluated the proposed crowd layer on a named entity recognition (NER) task. For this purpose, we used the AMT dataset introduced in (Rodrigues, Pereira, and Ribeiro 2013) which is based on the 2003 CoNLL shared NER task (Sang and Meulder 2003), where the goal is to identify the named entities in the sentence and classify them as persons, locations, organizations or miscellaneous. The dataset consists of 5985 labeled sentences using a pool of 47 workers. The remaining 3250 sentences of the original dataset were used for testing. The neural network architecture used for this problem consists of a layer of 300-dimensional word embeddings initialized with the pre-trained weights of GloVe (Pennington, Socher, and Manning 2014), followed by a 5x5 convolutional layer with 512 features, whose output is then fed to a GRU cell with a 50-dimensional hidden state. The individual hidden states of the GRU are then

passed to a FC layer with a softmax activation. The crowd layer uses the same annotator function mappings f_r used for image classification.

The proposed crowd layer is compared with same baselines considered for the classification problems. As previously explained, the EM approach is hard to generalize to sequence labelling problems due to marginalization over the latent ground truth sequences in Eq. (1). In order to make this marginalization tractable, we assume a fully factorized distribution of the posterior approximation $q(\mathbf{z}_n)$, such that $q(\mathbf{z}_n) = \prod_{t=1}^{T} q(z_{nt})$, where T denotes the length of the sequence.[3] Although the focus of this paper is on deep learning approaches, for the sake of completeness, we also compare with the results of the multi-annotator approach from (Rodrigues, Pereira, and Ribeiro 2013) based on conditional random fields (CRF-MA). Table 3 shows the obtained average results, which clearly demonstrate that the proposed approach significantly outperforms all the other methods, and provides similar results to those of CRF-MA, while reducing the training time by at least one order of magnitude when compared to the latter (minutes instead of several hours on a Core i7 with 32GB of RAM and a NVIDIA GTX 1070).

Conclusion

This paper proposed the *crowd layer* - a novel neural network layer that enables us to train deep neural networks end-to-end, directly from the labels of multiple annotators and crowds, using backpropagation. Despite its simplicity, the crowd layer is able to capture the reliabilities and biases of the different annotators and adjust the error gradients that are backpropagated during training accordingly. As our empirical evaluation shows, the proposed approach outperforms other approaches that rely on the aggregation of the annotators' answers prior to training, as well as other methods from the state-of-the-art which often rely on more complex, harder to setup and more computationally demanding EM-based approaches. Furthermore, unlike the latter, the crowd layer is trivial to generalize beyond classification problems, which we empirically demonstrate using real

---

[3]Please note that, while this makes EM tractable, the computational complexity of the E-step is now increased to $\mathcal{O}(NTR)$.

data from Amazon Mechanical Turk for text regression and named entity recognition tasks.

References

Albarqouni, S.; Baur, C.; Achilles, F.; Belagiannis, V.; Demirci, S.; and Navab, N. 2016. Aggnet: deep learning from crowds for mitosis detection in breast cancer histology images. *IEEE transactions on medical imaging* 35(5):1313–1321.

Buhrmester, M.; Kwang, T.; and Gosling, S. D. 2011. Amazon's mechanical turk a new source of inexpensive, yet high-quality, data? *Perspectives on psychological science* 6(1):3–5.

Chollet, F. 2015. Keras. https://github.com/fchollet/keras.

Dawid, A. P., and Skene, A. M. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *J. of the Royal Statistical Society. Series C* 28(1):20–28.

Donmez, P., and Carbonell, J. 2008. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *Proc. of the 17th ACM Conf. on Information and Knowledge Management*, 619–628.

Greenspan, H.; van Ginneken, B.; and Summers, R. M. 2016. Guest editorial deep learning in medical imaging: Overview and future promise of an exciting new technique. *IEEE Transactions on Medical Imaging* 35(5):1153–1159.

Guan, M. Y.; Gulshan, V.; Dai, A. M.; and Hinton, G. E. 2017. Who said what: Modeling individual labelers improves classification. *arXiv preprint arXiv:1703.08774*.

Ioffe, S., and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*.

Ipeirotis, P. G.; Provost, F.; and Wang, J. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, 64–67. ACM.

Kaggle. 2013. Dogs vs. cats competition. https://www.kaggle.com/c/dogs-vs-cats.

Kingma, D., and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

LeCun, Y.; Bengio, Y.; and Hinton, G. 2015. Deep learning. *Nature* 521(7553):436–444.

Parent, G., and Eskenazi, M. 2011. Speaking to the crowd: Looking at past achievements in using crowdsourcing for speech and predicting future challenges. In *INTERSPEECH*, 3037–3040. Citeseer.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, 1532–1543.

Raykar, V.; Yu, S.; Zhao, L.; Valadez, G.; Florin, C.; Bogoni, L.; and Moy, L. 2010. Learning from Crowds. *J. Mach. Learn. Res* 1297–1322.

Rodrigues, F.; Lourenço, M.; Ribeiro, B.; and Pereira, F. 2017. Learning supervised topic models for classification and regression from crowds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Rodrigues, F.; Pereira, F.; and Ribeiro, B. 2013. Sequence labeling with multiple annotators. *Machine Learning* 1–17.

Rodrigues, F.; Pereira, F.; and Ribeiro, B. 2014. Gaussian process classification and active learning with multiple annotators. In *Proc. of the 31st Int. Conf. on Machine Learning*, 433–441.

Russell, B.; Torralba, A.; Murphy, K.; and Freeman, W. 2008. Labelme: a database and web-based tool for image annotation. *Int. J. of Computer Vision* 77(1-3):157–173.

Sang, E., and Meulder, F. D. 2003. Introduction to the conll-2003 shared task: language-independent named entity recognition. In *Proceedings of the 7th Conference on Natural Language Learning at HLT-NAACL*, volume 4, 142–147.

Schmidhuber, J. 2015. Deep learning in neural networks: An overview. *Neural networks* 61:85–117.

Sheng, V.; Provost, F.; and Ipeirotis, P. 2008. Get another label? Improving data quality and data mining using multiple, noisy labelers. In *Proc. of the Int. Conf. on Knowledge Discovery and Data Mining*, 614–622.

Simonyan, K., and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Smyth, P.; Fayyad, U.; Burl, M.; Perona, P.; and Baldi, P. 1995. Inferring ground truth from subjective labelling of venus images. In *Advances in Neural Information Processing Systems*, 1085–1092.

Snow, R.; O'Connor, B.; Jurafsky, D.; and Ng, A. 2008. Cheap and fast - but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proc. of the Conf. on Empirical Methods in Natural Language Processing*, 254–263.

Su, H.; Deng, J.; and Fei-Fei, L. 2012. Crowdsourcing annotations for visual object detection. In *Workshops at the Twenty-Sixth AAAI Conference on Artificial Intelligence*, volume 1.

Welinder, P.; Branson, S.; Perona, P.; and Belongie, S. 2010. The multidimensional wisdom of crowds. In *Advances in neural information processing systems*, 2424–2432.

Whitehill, J.; Wu, T.-f.; Bergsma, J.; Movellan, J. R.; and Ruvolo, P. L. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in neural information processing systems*, 2035–2043.

Yan, Y.; Rosales, R.; Fung, G.; Subramanian, R.; and Dy, J. 2014. Learning from multiple annotators with varying expertise. *Mach. Learn.* 95(3):291–327.