WHO SAID WHAT: MODELING INDIVIDUAL LABELERS IMPROVES CLASSIFICATION

Melody Y. Guan,Varun Gulshan, Andrew M. Dai & Geoffrey E. Hinton
Google Brain
1600 Amphitheatre Pwky, Mountain View, CA 94043
{melodyguan,varungulshan,adai,geoffhinton}@google.com

ABSTRACT

Data are often labeled by many different experts, with each expert labeling a small fraction of the data and each example receiving multiple labels. When experts disagree, the standard approaches are to treat the majority opinion as the truth or to model the truth as a distribution, but these do not make any use of potentially valuable information about which expert produced which label. We propose modeling the experts individually and then learning averaging weights for combining them, possibly in example-specific ways. This allows us to give more weight to more reliable experts and take advantage of the unique strengths of individual experts at classifying certain types of data. We show that our approach performs better than three competing methods in computer-aided diagnosis of diabetic retinopathy.

1 INTRODUCTION

Deep convolutional neural networks have recently led to rapid improvements in the ability of computers to classify objects in images and they are now comparable with human performance in several domains. As they continue to improve, especially for tasks where it is possible to get a very large number of accurately labeled training examples, we can soon expect neural networks to start serving as alternatives to human experts. However the experts used to provide the training labels are often unreliable as indicated by the poor agreement between different experts (55.4% for the datasets we consider) or even between an expert and the same expert looking at the same image some time later (70.7%). This paper demonstrates that there are significantly better ways to use the opinions of multiple experts rather than using the experts to define a probability distribution over labels. Our approach is applicable to the myriad real-world settings that use experts to define ground truth.

2 MOTIVATION

In this paper we focus on datasets of images used for screening diabetic retinopathy (DR), where neural networks have recently achieved human-level performance (Gulshan et al. (2016)). That and previous work simply take the average opinion of all the doctors who have labeled a particular image and treat this distribution as the correct answer for training and evaluation. Our work explores whether a better model can be trained by predicting the opinions of the individual labelers. This preserves the information contained in the assignment of experts to opinions. We expect that some doctors will be more reliable than others and we would like to upweight their opinions. We also expect that the doctors will have received different training and may have received different distributions of images to rate so that the relative reliability of two doctors may depend on both the class of the image and on properties of the image such as the type of camera it was taken with.

Making better use of noisy labels: The amount of constraint that a training case imposes on the weights of a neural net depends on the amount of information required to specify the desired output. So if we force the network to predict what each particular doctor would say for each training case, by having an output layer per doctor, we should be able to get better generalization to test data provided this does not introduce too many parameters. At test time we can compute and average the predictions of all of the modeled doctors. Additionally we can learn how much to weight each modeled doctor's opinion in the averaging, allowing us to downweight unreliable models.

Diabetic retinopathy classification: DR is the fastest growing cause of blindness worldwide, with nearly 415 million diabetics at risk (IDF (2015)). Early detection and treatment can reduce the risk of blindness by 95% (NEI (2015)). The disease is commonly detected by a specialist examining images of the back of the eye and rating them on a 5-point severity scale (AAO (2002)). Our training and validation datasets consist respectively of 126,522 and 7,805 images sourced from DR screening patients. 30 of a total of 54 ophthalmologists graded at least 1,000 of these images and the rest we lumped into a composite doctor to avoid introducing doctor-specific parameters constrained by too few examples. To obtain single opinions on the 3,547-image test set for model evaluation, we introduce a rigorous adjudicated reference standard where labels are obtained from committee discussions by 3 retinal specialists, who were excluded from training and validation (Appendix C).

3 METHODS

We consider a sequence of models of increasing complexity (see Appendix D for diagrams, explicit architectural descriptions, and loss inputs in tabular form). For a single image, let I be the set of indices of the doctors who graded that image, and let l_i be the label of doctor $i \in I$. The target used to compute cross entropy loss is $\frac{1}{|I|}\sum_{i\in I} l_i$ in training and evaluation for all models.

- *Baseline Net (BN)*: Inception-v3 (Szegedy et al. (2016)) trained on average opinions of doctors with a single 5-way softmax output. This is a reimplementation of Gulshan et al. (2016) (see Appendix E for differences). Let $p_\emptyset$ be the prediction of BN's average doctor model. For both training and evaluation of BN, the prediction input used in loss computation is $p_\emptyset$.
- *Doctor Net (DN)*: BN extended to model the opinions of each of the 31 doctors using a separate softmax for each doctor, with Inception weights shared across doctors. The predictions from the "doctor models" are then arithmetically averaged at test time. For every doctor $j \in \{1, 2, \ldots, 31\}$, denote the 5-dimensional prediction of its model p_j. For training DN, the loss is calculated for each $i \in I$ using the cross entropy between p_i and the i^{th} doctor opinion, and is summed across all $i \in I$ to get the total loss for that training example. The prediction input of DN for evaluation is $\frac{1}{31}\sum_{i=1}^{31} p_i$.
- *Weighted Doctor Net (WDN)*: Fixed weights and predictions of DN with averaging weights learned on top for combining the predictions of the doctor models, with one weight per doctor model. Let constant w_j be the averaging weight for the jth modeled doctor, where $\sum_j w_j = 1$. The prediction input is $\frac{\sum_{i\notin I} p_i w_i}{\sum_{i\notin I} w_i}$ for training and $\sum_{i=1}^{31} p_i w_i$ for evaluation.
- *Image-specific WDN (IWDN)*: WDN with averaging weights that are a function of the last layer of inception (i.e. image-dependent). The loss inputs are the same as those of WDN, except w_j is now different for different images.
- *Bottlenecked IWDN (BIWDN)*: IWDN with a linear bottleneck of size 3 between the last hidden layer of Inception and the 31-way softmax producing averaging weights. This bottleneck reduced the number of parameters for learning the averaging weights by about 10 times. The loss inputs for BIWDN are the same as those of IWDN.

Rather than directly learning the averaging weight for each doctor model in (B)(I)WDN, we learned averaging logits for each model that we could pass through a softmax to produce positive averaging weights. If a doctor model has similar performance to other doctor models but makes very different errors it will tend to be upweighted because it will be more useful in the averaging. This would not occur if we computed the reliabilities of the doctors separately. Note that we not need any extra data for learning the averaging logits beyond those used to learn the weights of DN.

Neural network training and hyperparameter tuning are discussed in Appendices F, G, and H. Our early stopping metric was 5-class classification error rate. We ensembled the predictions for the horizontally and vertically flipped versions of every test set image. We ran 10 replicates of each model and averaged the resulting metrics. We used the same 10 replicates reported for DN as the fixed part of the model for the (B)(I)WDN replicates.

Estimating doctor reliability with EM: Since the foundational work of Dawid & Skene (1979), who model annotator accuracies with expectation-maximization (EM), and Smyth et al. (1995), who integrate the opinions of many experts to infer ground truth, there has a large body of work using

Table 1: Summary of Results.

Test Metric (%)	BN		DN		WDN	IWDN	BIWDN
	$\emptyset$	Welinder	$\emptyset$	Mnih			
5-class Error	23.83	23.74	21.86	22.76	**20.58**	20.63	20.83
Binary Error	9.92	10.12	9.75	10.24	**9.07**	9.12	9.23
Binary AUC	97.11	97.00	97.28	97.42	**97.45**	97.43	97.41
Spec@97%Sens	79.60	79.97	81.81	**83.61**	82.69	82.46	82.46

EM approaches to estimate accurate labels for datasets annotated by multiple experts (Whitehill et al., 2009; Raykar et al., 2009; Raykar & Yu, 2012). Representatively, Welinder & Perona (2010) use an online EM algorithm to estimate abilities of multiple noisy annotators and to determine the most likely value of the labels. We apply their algorithm on our data (details in Appendix I) and used the updated labels to train BN as a competing algorithm for our DN method.

Modeling label noise: Mnih & Hinton (2012) propose a robust loss function that models asymmetric omission noise to handle label noise when labelling road pixels. They assume that a true, unobserved label m is first generated from an image patch s according to some distribution $p(m|s)$, and the corrupted, observed label $\tilde{m}$ is then generated from m according to an asymmetric binary noise distribution $p(\tilde{m}_i|m_i)$ that is the same for all pixels i. The observed label distribution is then modeled as $p(\tilde{m}|s) = \prod_i \sum_{m_i} p(\tilde{m}_i|m_i)p(m_i|s)$. We used a multi-class extension of their method as an alternative way to improve upon DN to our proposed averaging weight approach. We model the noise distribution prior for all doctors d with the parameters $\theta_{ll'} = p(\tilde{m}_d = l'|m_d = l), l, l' \in \{1,2,3,4,5\}$. We estimated $\theta_{ll'}$ using the 5×5 confusion matrix between individual and average doctor opinions on training images, treating the latter as the truth and averaging proportions calculated from individual doctor matrices across all doctors. Our training loss was the negative log posterior, $-log(p(\tilde{m}|s))$.

4 RESULTS

Averaging modeled doctors beats modeling the average doctor: We saw a reduction in 5-class classification test error of 1.97% from 23.83% (8.27% relative decrease) due to averaging modeled doctors instead of modeling the averaged doctor (DN vs BN). In comparison, using EM-derived labels to train BN only reduced 5-class test classification error by 0.09% (0.38% relative decrease). With the same model checkpoints used for reporting 5-class classification error, DN also showed better test performance on binary AUC, binary classification error, and specificity at 97% sensitivity than using labels derived from the Welinder & Perona (2010) algorithm (Table 1).

Learning averaging weights helps: We saw a further 1.28% decrease in 5-class test error from using WDN (5.37% additional relative decrease). Results from (B)IWDN were slightly worse than WDN's. We might expect a larger improvement from WDN and potentially further benefits from training image-specific averaging logits if we had experts with more varied abilities and greater environmental differences, but for our dataset image-specific averaging logits did not help. Using Mnih & Hinton (2012)'s competing algorithm actually caused DN to perform worse by 0.90% on 5-class test error (3.78% less relative reduction), and was more computationally costly than (B)(I)WDN.

5 CONCLUSIONS

We introduce a method to make more effective use of noisy labels when examples are graded by multiple experts: we learn from the identity of multiple annotators by modeling them individually with a shared neural net that has separate sets of outputs for each expert, and then learning averaging weights for combining their modeled predictions. We evaluated our method on the diagnosis of DR from images of the retina. Our approach lowered classification test error from 23.83% to 20.58% compared to our baseline model of training on the average doctor opinion, a relative reduction of 13.6%. We also attained superior performance to algorithms by Welinder and Perona and by Mnih and Hinton. Our methodology is generally applicable to supervised training systems using datasets with multiple overlapping annotators.

REFERENCES

M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: A system for large-scale machine learning. *OSDI*, November 2016.

A. P. Dawid and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. *Applied Statistics*, 28(1):20–28, March 1979.

D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. *ICCV*, 11(18):2650–2658, December 2015.

V. Gulshan, L. Peng, M. Coram, M. C. Stumpe, A. Narayanaswamy D. Wu, S. Venugopalan, K. Widner, T, Madams, J. Cuadros, R. Kim, R. Raman, P. C. Nelson, J. L. Mega, and D. R. Webster. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22):2402–2410, December 2016.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *ICML*, 37:448–45, March 2015.

D. P. Kingma and J. L. Ba. Adam: A method for stochastic optimization. *ICLR*, July 2015.

V. Mnih and G. E. Hinton. Learning to label aerial images from noisy data. *ICML*, pp. 567–574, July 2012.

G. Pereyra, G. Tucker, L. Kaiser, and G. E. Hinton. Regularizing neural networks by penalizing confident output distributions. *arXiv*, January 2017.

B. T. Polyak and A. B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30:838—855, July 1992.

V. Raykar, S. Yu, L. Zhao, A. Jerebko, C. Florin, G. Valadez, L. Bogoni, and L. Moy. Supervised learning from multiple experts: Whom to trust when everyone lies a bit. *ICML*, 26:889—-896, June 2009.

V. C. Raykar and S. Yu. Eliminating spammers and ranking annotators for crowdsourced labeling tasks. *JMLR*, 13:491—518, 2012.

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. *JMLR*, 115(3):211–252, January 2015.

P. Smyth, U. Fayyad, M. Burl, P. Perona, and P. Baldi. Inferring ground truth from subjective labelling of venus images. *NIPS*, 7:1085–92, 1995.

C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Re-thinking the inception architecture for computer vision. *CVPR*, pp. 2818–2826, June 2016.

P. Welinder and P. Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. *CVPR Workshop*, pp. 25–32, June 2010.

J. Whitehill, P. Ruvolo, T. Wu, J. Bergsma, and J. Movellan. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *NIPS*, 22:2035—-2043, 2009.

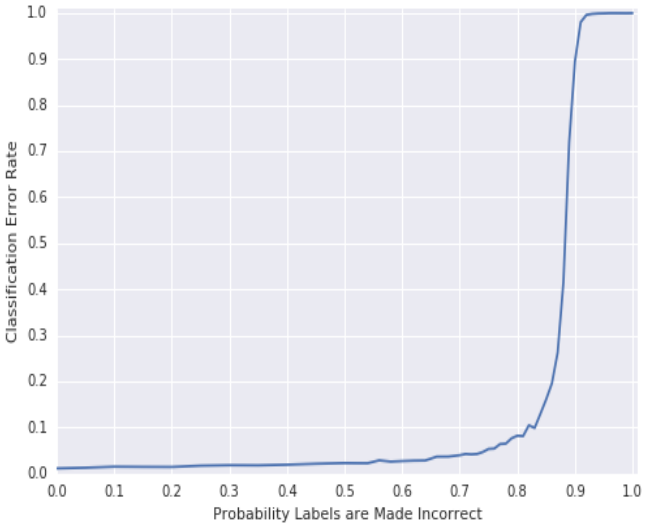


Figure 1: Performance of a deep neural net when trained with noisy labels.

A CODE

The TensorFlow code used in this paper will be made publicly available.

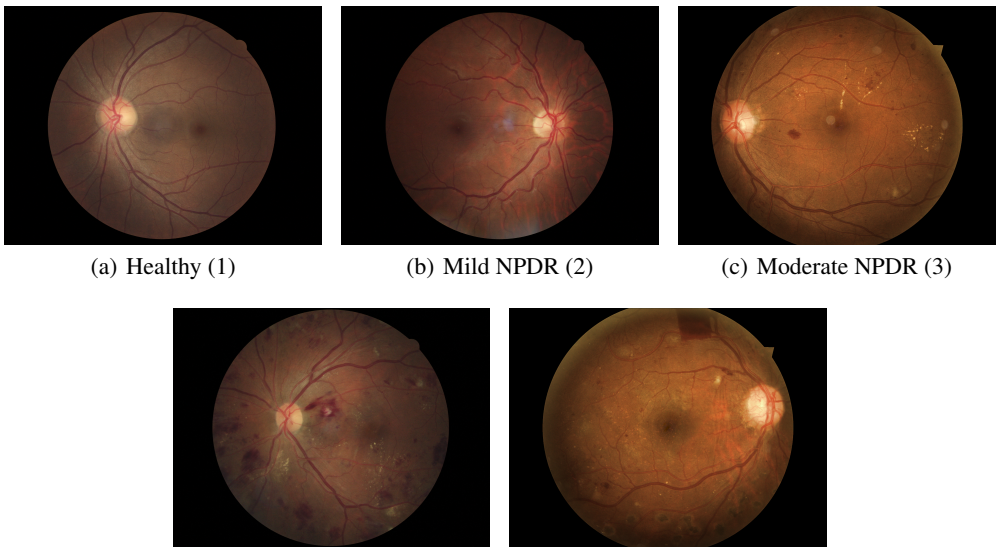
B A MOTIVATING DEMONSTRATION

Intuitively, we would expect the quality of the training labels to provide an upper bound on the performance of the trained net. In this appendix we show that this intuition is incorrect.

To demonstrate that a trained neural net can perform far better than its teacher we use the well-known MNIST benchmark for which the true labels are known and we create unreliable training labels by corrupting the true labels. This corruption is performed just once per experiment, before training starts, so the noise introduced by the corruption cannot be averaged away by training on the same example several times. MNIST has 60,000 training images and 10,000 test images of isolated, normalized, hand-written digits and the task is to classify the image into one of ten classes. Each image has 28×28 pixels. For the purposes of this demonstration, we used a very simple neural net containing two hidden convolutional layers each with 1,024 rectified linear units and 64 patches followed by a fully connected hidden layer of 32 rectified linear units followed by a 10-way softmax layer. We trained the net on 50,000 examples using stochastic gradient descent on mini-batches of size 200 with the Adam optimizer and we used the remaining 10,000 training images as a validation set to select good values for the learning rate and the magnitude of the initial random weights. On the test data, the net that performed best on the validation set had an test error rate of 1.01% when the training labels were all correct.[1] If the labels are corrupted by changing each label to one of the other nine classes with a probability of 0.5, the test error rate only rises to 2.29%. Even if each training label is changed to an incorrect label with probability 0.8 so that the teacher is wrong 80% of the time, the trained net only gets 8.23% test error. If the teacher is even less reliable there comes a point at which the neural net fails to "get the point" and its error rate rises catastrophically, but this does not happen until the teacher is extremely unreliable as is shown in Figure 1.

This demonstration shows that the performance of a neural net is not limited by the accuracy of its teacher, provided the teacher's errors are random. One obvious question is how many noisily labeled training cases are worth the same as one case that is known to be correctly labeled. This question can be answered, at least approximately, by computing the mutual information between the label and the

[1] We did hyperparameter tuning for data where each training label was changed to an incorrect label with probability 0.8. If we had tuned for data where all labels were correct, the corresponding error rate would have been lower.



(a) Healthy (1) (b) Mild NPDR (2) (c) Moderate NPDR (3)

(d) Severe NPDR (4) (e) Proliferative DR (5)

Figure 2: Sample fundus images from each DR class.

truth, assuming random noise. Empirically, N perfectly labeled training cases give about the same test error as $NI_{\text{perfect}}/I_{\text{noisy}}$ training cases with noisy labels, where I_{noisy} is the mutual information per case between a noisy label and the truth and I_{perfect} is the corresponding mutual information for perfect labels. For ten classes, the mutual information (in nats) is $I_{\text{perfect}} = 2.3 = -log(0.1)$, but when the noisy label is 20% correct on average, the mutual information is:

$$I_{\text{noisy}} = 0.044 = -log(0.1) - 10 \times 0.02 \times log\left(\frac{0.1}{0.02}\right) - 90 \times 0.1 \times \frac{0.8}{9} log\left(\frac{0.1}{0.1 \times 0.8/9}\right).$$

So if the learning is making good use of the mutual information in the noisy labels we can predict that 60,000 noisy labels are worth $60,000 \times 0.044/2.3 \approx 1148$ clean labels. In reality we needed about 1,000 clean labels to get similar results.

C DATASET DETAILS

Sample images from the 5 classes of DR are shown in Figure 2.

The training dataset consists of $126,522$ images sourced from patients presenting for DR screening at sites managed by 4 different clinical partners: EyePACS, Aravind Eye Care, Sankara Nethralaya, and Narayana Nethralaya. 119,589 of our training images are the same as those used in the training dataset of Gulshan et al. (2016) (which consists of 128,175 images). The images removed from the training dataset used by Gulshan et al. (2016) are detailed here: (i) 4,204 out of the 128,175 were removed to create a separate validation dataset for experiments within the research group, and (ii) 4,265 out of the 128,175 images were excluded since they were deemed ungradable by every ophthalmologist that graded them. Unlike Gulshan et al. (2016), we do not predict image gradeability in this work and hence exclude those images. (iii) 117 out of the 128,175 fail our image normalization preprocessing step and were also excluded. There were 6,933 subsequently labeled images added to this training set which are not present in the training set of Gulshan et al. (2016).

The validation dataset consists of 7,963 images obtained from EyePACS clinics. These images are a random subset of the 9,963 images of the EyePACS-1 test set used in Gulshan et al. (2016). The remaining 2,000 images were included as part of the test set in this work. In practice, only 7,805 of the 7,963 validation images have at least one label, since the remaining 158 images were of poor quality and considered ungradable by all ophthalmologists that labeled them.

The test set consists of 1,748 images of the Messidor-2 dataset and the remaining 2,000 out of the 9,963 images of the EyePACS-1 test dataset used in Gulshan et al. (2016). The labels for the test

Table 2: Class distributions of training and validation datasets (as %).

Grade	Training	Validation
1	51.03	72.69
2	24.75	17.62
3	16.81	7.27
4	4.17	1.20
5	3.23	1.21

set were obtained through an adjudication process: 3 retina specialists graded all images in the test dataset, and any disagreements were discussed until a consensus label was obtained. 1,803 of the 2,000 images from the EyePACS-1 test set, and 1,744 of the 1,748 images of the Messidor-2 were considered gradable after adjudication and were assigned labels.

D NET ARCHITECTURE DETAILS

The networks used in this paper are shown schematically in Figures 3 and 4. The explicit inputs of the cross entropy loss being minimized during training of each model are shown in Table 3. Below we discuss the network architectures in greater depth, with references to Figure 3.

For BN, the outputs of the last hidden layer of Inception (pink box in BN model in Figure 3) were multiplied by a weight matrix (lavender box) to compute the logits used in the 5-way softmax output layer (yellow box). For DN, the last hidden layer of Inception was passed through the same kind of output layer as in BN, but with one separate softmax per doctor. In this way, every doctor is modeled separately by the incoming weights of its softmax. This is illustrated in the figure by the 31 yellow prediction outputs each being computed using its own set of lavender parameters and resulting from an individual 5-way softmax. For evaluation, the predictions from the softmax "doctor models" were arithmetically averaged to give a single 5-class prediction. For subsequent nets ((B)(I)WDN), the parameters of the DN model were frozen (faded lavender boxes) and only the averaging weights for the doctor models were learned.

For WDN (not shown in the figure), one averaging weight per doctor was learned, by training averaging logits that were then passed through a softmax. For IWDN, the averaging weights were made a function of the image by multiplying the last hidden layer of Inception with a weight matrix to compute the logits that were passed through a 31-way softmax (31 is the number of doctor models trained). In the IWDN model of the figure, this corresponds to the rightmost lavender parameter box being used to calculate the 31 averaging weights which were the output of a single 31-way softmax. For BIWDN, a linear bottleneck layer of size 3 (pink bottleneck box) was added between the last hidden layer of Inception (which has dimension 2048) and the 31-way softmax of IWDN as a precautionary measure against model underfitting.

To train the averaging logits, we used the opinions of the doctors who actually labeled a training image to define the target output distribution for that image. We then combined the predictions of the models of all the other doctors using the weights defined by their current averaging logits. Finally we updated our parameters by backpropagating with the cross entropy loss between the target distribution and the weighted average prediction. This way all of the training cases that a doctor did not label can be used to learn the averaging logit for that doctor, and no extra data were needed beyond those used to learn the weights of DN.

E OUR BASELINE VS PUBLISHED BASELINE

Besides using different datasets, including an adjudicated test set, other differences between our baseline and that published by Gulshan et al. (2016) are the following.

- Unlike in Gulshan et al. (2016), we remove grades of doctors who graded test set images from training and validation sets to reduce the chance that the model is overfitting on certain

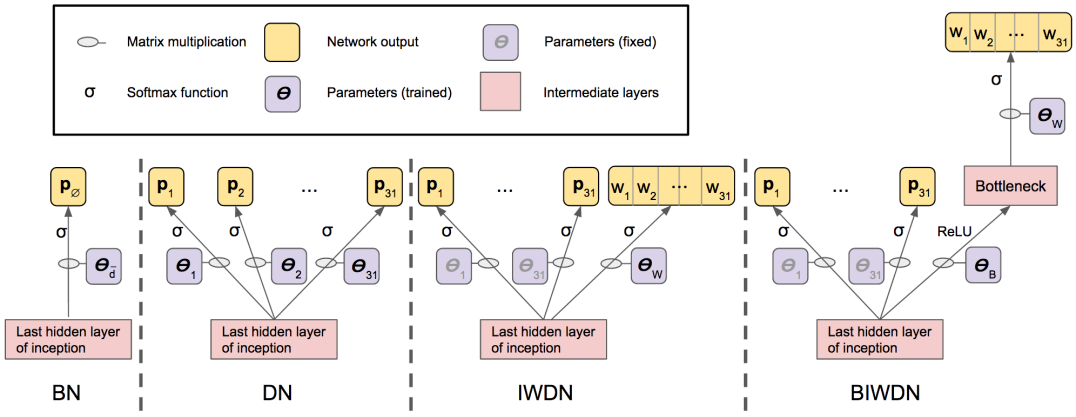


Figure 3: Schematic diagram of nets. These schematics show how the parameters, network outputs, and averaging weights for doctor models are connected. Section 3 described how the outputs are used in a loss function for training. In WDN (not shown in figure), the averaging logits are not connected to the last hidden layer of Inception and are just initialized from a constant vector.

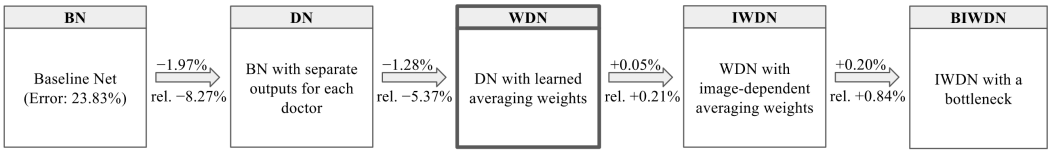


Figure 4: Description of nets used in paper. The baseline net had 5-class classification error rate of 23.83% on the test dataset. The numbers above arrows refer to absolute changes in test error while the numbers below arrows refer to relative changes (negative values represent improvements). WDN (highlighted) was the optimal net.

Table 3: Prediction inputs to cross entropy loss for each model during training. The notation is given in the text. Note that the target is always $\frac{1}{|I|}\sum_{i\in I} l_i$.

Model	Training	Evaluation
BN	$p_\emptyset$	$p_\emptyset$
DN	$p_i, \forall i \in I$	$\frac{1}{31}\sum_{i=1}^{31} p_i$
(B)(I)WDN	$\frac{\sum_{i\notin I} p_i \cdot w_i}{\sum_{i\notin I} w_i}$	$\sum_{i=1}^{31} p_i \cdot w_i$

experts. This removal handicaps our performance vis-à-vis their paper, especially because we exclude the most expert doctors from model development, but ensures generalizability of our results.

- Gulshan et al. (2016) define referable diabetic retinopathy as the presence of moderate and worse diabetic retinopathy *or* referable diabetic macular edema, while we ignore information on the latter.
- Gulshan et al. (2016) used binary loss with early stopping on binary AUC, while we used 5-class loss with early stopping on 5-class classification error in all cases except when we tested BN with binary loss and showed that it performed worse than BN with a 5-class loss (Appendix J). For this comparison, BN binary was validated with binary AUC and BN with a 5-class loss was validated with 5-class classification error. Note also that Gulshan et al. (2016) only reported binary classification metrics while we reported both binary and 5-class classification metrics.
- We did not ensemble replicates as Gulshan et al. (2016) did because we focused on comparing different methods of using the labels rather than squeezing the last drop of performance from one method.
- If a doctor grades a single image multiple times, as often occurs, Gulshan et al. (2016) treats these as independent diagnoses while we collapse these multiple diagnoses into a single diagnosis which may be a distribution over classes.
- We employed higher resolution images (587×587 pixels versus 299×299), and a variety of image preprocessing and theoretical techniques unused in Gulshan et al. (2016): the images were scale normalized by detecting the circular fundus disk and removing the black borders around them and then training data were augmented with random perturbations to image brightness, saturation, hue, and contrast.

In light of all these distinctions, the numbers we report are only roughly comparable to those reported in Gulshan et al. (2016) and this paper's own baseline net should be used for model comparisons.

F Training Details

We trained our nets using distributed SGD (Abadi et al. (2016)) with the Adam optimizer (Kingma & Ba (2015)) on mini-batches of size 8. We trained using TensorFlow with 32 replicas and 17 parameter servers, with 1 Tesla K80 GPU per replica. To speed up the training, we used batch normalization (Ioffe & Szegedy (2015)), pre-trained Inception-v3 with the imagenet dataset (Russakovsky et al. (2015)), and set the learning rate on the weight matrix producing prediction logits to one-tenth of the learning rate for the other weights. We prevented overfitting using a combination of L1 and L2 penalties, dropout, and a confidence penalty (Pereyra et al. (2017)), which penalizes a model for having an output distribution with low entropy. We also used Polyak averaging Polyak & Juditsky (1992) to get the final weights used in BN and DN.

G Dealing with class distribution

G.1 Log Prior Correction

To deal with differences in class distribution between the datasets (Table 2), we used log prior correction during evaluation. This entails adding to the prediction logits, for each class, the log of the ratio of the proportion of labels in that class in the evaluation dataset to the proportion of labels in that class in the training set. Our assumed test class distribution for computing the log prior correction was the mean distribution of all known images (those of the training and validation sets). So for each image under evaluation we update the prediction logit for class c by adding:

$$log\left(\frac{q_{valid}(c)}{q_{train}(c)}\right) \qquad \text{for the validation dataset, and}$$

$$log\left(\frac{q_{valid\cup train}(c)}{q_{train}(c)}\right) \qquad \text{for the test dataset,}$$

Table 4: Optimal Hyperparameters from Grid Search. Note that the learning rate for doctor models is one-tenth the learning rate for the rest of the network. WD=weight decay, wel=welinder

Hyperparameter	BN binary	BN	BN wel	DN	DN mnih	WDN	IWDN	BIWDN
Learning rate	0.0001	0.0003	0.0003	0.001	0.0003	0.03	1×10^{-6}	3×10^{-7}
Dropout for Inception	0.75	0.95	0.95	0.85	0.95	-	-	-
Dropout for output heads	0.8	0.85	0.85	0.9	0.9	-	-	-
Entropy weight	0.0125	0.025	0.015	0.0175	0.02	0.0225	0.005	0.0125
L2 WD for Inception	0.01	0.01	0.01	0.001	0.004	-	-	-
L1 WD for doctor models	0.001	0.00004	0.0001	0.001	0.01	-	-	-
L2 WD for doctor models	0.01	0.004	0.001	0.01	0.04	-	-	-
L1 WD for averaging logits	-	-	-	-	-	0.4	0.02	4
L2 WD for averaging logits	-	-	-	-	-	15	0.4	110
Bottleneck size	-	-	-	-	-	-	-	3

where $q(c)$ is the proportion of labels in that class. Note that we did not follow the standard machine learning paradigm of assuming that the test dataset has the same distribution as the validation dataset because our test data are sourced differently from the validation data. We saw improvement from the application of log prior correction and all our reported results use it.

G.2 Mean Class Balancing

In addition to log prior correction of class distributions, we also attempted mean class balancing wherein examples from less frequent classes are upweighted and more frequent classes are down-weighted in the cross entropy loss, in inverse proportion to their prevalence relative to the uniform distribution across classes. Explicitly, we weight each example of class c by:

$$\alpha_c = \frac{\bar{q}}{q(c)} = \frac{1}{|c|q(c)},$$

Eigen & Fergus. (2015) employ a similar method for computer vision tasks although they use medians instead of means. In our case, using mean class balancing lowered performance, possibly because it made too many assumptions on the hidden test distribution, and was not employed.

H Hyperparameter tuning

All hyperparameter tuning and early stopping were performed on validation datasets.

For the MNIST experiment we used default Adam optimizer hyperparameters β_1 = 0.9, β_2 = 0.999, and ϵ = 1×10^{-8}. We did a grid search on learning rates in the set {0.000003, 0.00001, 0.00003, ..., 0.003} and standard deviations of the initial random normal weights in the set {0.0001, 0.0003, 0.001, ..., 0.01} and found optimal values of 0.00003 for the former and 0.001 for the latter.

For computer-aided diagnosis of DR we did a grid search on the following hyperparameter spaces: dropout for Inception backbone $\in$ {0.5, 0.55, 0.6, ..., 1.0}, dropout for doctor models $\in$ {0.5, 0.55, 0.6, ..., 1.0}, learning rate $\in$ {1×10^{-7}, 3×10^{-7}, 1×10^{-6}, ..., 0.03}, entropy weight $\in$ {0.0, 0.0025, 0.005, ..., 0.03} $\cup$ {0.1}, weight decay for Inception $\in$ {0.000004, 0.00001, 0.00004, ..., 0.1}, L1 weight decay for doctor models $\in$ {0.000004, 0.00001, 0.00004, ..., 0.04}, L2 weight decay for doctor models $\in$ {0.00001, 0.00004, ..., 0.04}, L1 weight decay for averaging logits $\in$ {0.001, 0.01, 0.02, 0.03, ..., 0.1, 0.2, 0.3, ..., 1, 2, 3, ..., 10, 100, 1000}, L2 weight decay for averaging logits $\in$ {0.001, 0.01, 0.1, 0.2, 0.3, ...,1, 5, 10, 15, 20, 30, ..., 150, 200, 300, 400, 500, 1000}, and bottleneck size (for BIWDN) $\in$ {2, 3, 4, 5, 6, 7}. The optimal values for these hyperparameters are displayed in Table 4. We used a learning rate decay factor of 0.99 optimized for BN. The magnitudes of the image preprocessing perturbations were also tuned for BN.

I Welinder and Perona Algorithm Details

Welinder & Perona (2010)'s full algorithm also actively selects which images to label and how many labels to request based on the uncertainty of their estimated ground truth values and the desired level of confidence, and selects and prioritizes which annotators to use when requesting labels. In this

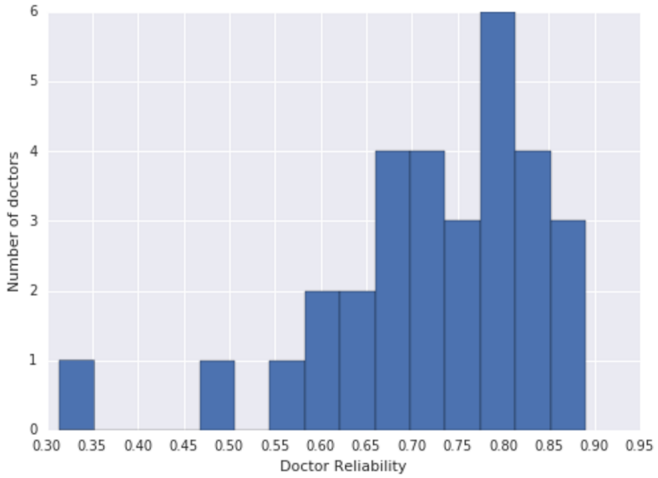


Figure 5: Histogram of doctor reliabilities. These were calculated from Welinder & Perona (2010)'s expectation-maximization algorithm on our training data.

Table 5: Test metrics from Multi-class vs Binary loss for BN.

Test Metric (%)	Trained with binary loss	Trained with 5-class loss
Binary AUC	95.58	97.11
Binary Error	11.27	9.92
Spec@97%Sens	63.12	79.60

work, we only use the expectation-maximization part of their algorithm because labels for all images in our dataset have already been collected. The doctor reliabilities we calculate with this method are shown in Figure 5.

J FIVE-CLASS LOSS VS BINARY LOSS

We also trained a version of BN where the output prediction is binary instead of multi-class, as was done in Gulshan et al. (2016). The binary output was obtained by thresholding the 5-class output at the *Moderate NPDR* or above level, a commonly used threshold in clinics to define a referable eye condition. For this BN-binary network, the area under the ROC curve was used as the evaluation metric on the validation set.

We found that training BN with a 5-class loss improved test binary AUC (defined with a standard top-3-class threshold for referable DR) by 1.53% from 95.58% (1.60% relative increase) compared to training with a binary loss, as did Gulshan et al. (2016) (Table 5). Intuitively this fits with our thesis that generalization is improved by increasing the amount of information in the desired outputs. All results reported in Table 1 were from training with 5-class loss.