SelectiveNet: A Deep Neural Network with an Integrated Reject Option

Yonatan Geifman [1] **Ran El-Yaniv** [1]

Abstract

We consider the problem of *selective prediction* (also known as *reject option*) in deep neural networks, and introduce SelectiveNet, a deep neural architecture with an integrated reject option. Existing rejection mechanisms are based mostly on a threshold over the prediction confidence of a pre-trained network. In contrast, SelectiveNet is trained to optimize both classification (or regression) and rejection simultaneously, end-to-end. The result is a deep neural network that is optimized over the covered domain. In our experiments, we show a consistently improved risk–coverage trade-off over several well-known classification and regression datasets, thus reaching new state-of-the-art results for deep selective classification.

1. Introduction

Detecting and controlling statistical uncertainties in machine learning processes is essential in many mission-critical machine learning applications such as autonomous driving, medical diagnosis or home robotics. In *selective prediction* our goal is to learn predictive models that know what they do not know. These models are allowed to abstain whenever they are not sufficiently confident in their prediction.

The idea of a reject option was already studied over 60 years ago by Chow (1957). Reject option mechanisms have been considered for many hypothesis classes and learning algorithms. These mechanisms are mainly driven by two related ideas. The first is the cost-based model whereby the selective model is optimized to minimize a loss function that tracks a user-specified cost for abstention. Other mechanisms are built upon functions that serve as proxies for prediction confidence. Typical examples of these proxies are functional margin, distance to nearest neighbor(s), etc.

In this work we focus on selective models for deep neural networks (DNNs). The most recent relevant paper on this topic is Geifman & El-Yaniv (2017), which shows how to construct a probabilistically-calibrated selective classifier using any certainty estimation function for a given (already trained) model. Interestingly, among the best certainty proxies for a trained network is the simple Softmax Response (SR) estimator, which is often used by practitioners.

Within an abstention-constrained framework, whereby the user specifies the desired target coverage, we optimize, end-to-end, a neural network with an integrated reject option, designed to be optimal for the required coverage slice. Our technique yields effective selective models that consistently improve all known techniques.

From a learning-theoretic perspective, it is not hard to argue that learning a specialized rejection model for each coverage slice is potentially better than rejecting based on confidence rates extracted from a pre-trained network. To demonstrate the advantage of such specialized training, consider the following vignette. Bob and Alice are about to take an exam at the end of their machine learning course. It is announced that the exam will contains 10 questions, one for each of the course topics, but the students will be asked to choose exactly five questions. Alice decides to cover only the easiest five topics while Bob's mom forces him to study all ten topics. Who do you think is going to get the higher grade?

The contributions of this paper are:

- A selective loss function that optimizes a specified coverage slice using a variant of the interior point optimization method (Potra & Wright, 2000).
- SelectiveNet: A three-headed network for end-to-end learning of selective classification models.
- SelectiveNet for regression: Providing the first alternative to costly methods such as MC-dropout or ensemble techniques.
- An empirical advantage demonstrated over four datasets for classification and regression with a significant advantage over MC-dropout and SR.

[1]Technion - Israel Institute of Technology. Correspondence to: Yonatan Geifman <yonatan.g@cs.technion.ac.il>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

2. Selective Prediction Problem Formulation

In this work we consider selective prediction problems of any type, classification or regression. A supervised prediction task is formulated as follows. Let $\mathcal{X}$ be any feature space and $\mathcal{Y}$ a label space. For example, in classification, $\mathcal{X}$ could be images, and $\mathcal{Y}$ could be class labels in the case of classification or the coordinates of a bounding box in the case of regression. Let $P(X,Y)$ be a distribution over $\mathcal{X}\times\mathcal{Y}$. A model, $f:\mathcal{X}\to\mathcal{Y}$, is called a *prediction function*, and its *true risk* w.r.t. P is $R(f)\triangleq E_{P(X,Y)}[\ell(f(x),y)]$, where $\ell: Y\times Y\to\mathbb{R}^+$ is a given loss function, for example, the 0/1 error or the squared loss. Given a labeled set $S_m=\{(x_i,y_i)\}_{i=1}^m\subseteq(\mathcal{X}\times\mathcal{Y})^m$ sampled i.i.d. from $P(X,Y)$, the *empirical risk* of the classifier f is $\hat{r}(f|S_m)\triangleq\frac{1}{m}\sum_{i=1}^m\ell(f(x_i),y_i)$.

A *selective model* (El-Yaniv & Wiener, 2010) is a pair (f,g), where f is a *prediction function*, and $g:\mathcal{X}\to\{0,1\}$ is a *selection function*, which serves as a binary qualifier for f as follows,

$$(f,g)(x)\triangleq\begin{cases}f(x), & \text{if } g(x)=1;\\ \text{don't know}, & \text{if } g(x)=0.\end{cases}$$

Thus, the selective model abstains from prediction at a point x iff $g(x)=0$. A *soft* selection function can also be considered, where $g:\mathcal{X}\to[0,1]$, and decisions can be taken probabilistically or deterministically (w.r.t. a threshold). The performance of a selective model is quantified using *coverage* and *risk*. Coverage is defined to be

$$\phi(g)\triangleq E_P[g(x)],$$

the probability mass of the non-rejected region in $\mathcal{X}$. The selective risk of (f,g) is

$$R(f,g)\triangleq\frac{E_P[\ell(f(x),y)g(x)]}{\phi(g)}.$$

Clearly, the risk of a selective model can be traded-off for coverage. The entire performance profile of such a model can be specified by its *risk–coverage* curve, defined to be the risk as a function of coverage (El-Yaniv & Wiener, 2010; Wiener & El-Yaniv, 2012).

The true selective risk and true coverage have corresponding empirical counterparts that can be calculated for any given labeled set S_m. The *empirical selective risk* is

$$\hat{r}(f,g|S_m)\triangleq\frac{\frac{1}{m}\sum_{i=1}^m\ell(f(x_i),y_i)g(x_i)}{\hat{\phi}(g|S_m)},$$

and the *empirical coverage* is

$$\hat{\phi}(g|S_m)\triangleq\frac{1}{m}\sum_{i=1}^m g(x_i).$$

An optimal selective model could be defined in one of two ways: by either optimizing the selective risk, given a constraint on the coverage or vice versa. In this work, we focus on the first case, which is defined as follows. For a given coverage rate $0<c\leq 1$ and hypothesis class Θ, the optimal selective model is

$$\theta^*=\arg\min_{\theta\in\Theta}(R(f_\theta,g_\theta)) \quad s.t.\ \phi(g_\theta)\geq c. \tag{1}$$

In this paper, Θ is a set of parameters for a given deep network architecture for f and g.

It is possible to convert any selective model defined as above (with controlled coverage and optimized risk) to a model that controls risk and optimizes coverage using the technique of Geifman & El-Yaniv (2017).

3. Related Work

The literature on reject option techniques is quite extensive and mostly focuses on hypothesis classes and learning algorithms such as SVMs, nearest neighbours and boosting (Fumera & Roli, 2002; Hellman, 1970; Cortes et al., 2016; El-Yaniv & Wiener, 2015). When considering selective prediction in neural networks (NNs), a straightforward and effective technique is to set a threshold over a confidence score derived from a pre-trained network, which was already optimized to predict all points (Cordella et al., 1995; De Stefano et al., 2000; Wiener & El-Yaniv, 2012). Geifman & El-Yaniv (2017), extending this technique for deep neural networks (DNNs), show how to derive a selective classifier based on any confidence score function. They also show how to compute a high confidence guarantee on the selective risk at test time. In the latter work, two techniques for extracting confidence scores from an NN are discussed. The first is the Softmax Response (SR), defined to be the maximal activation in the softmax layer (in classification architectures). The second confidence score is the Monte-Carlo dropout (MC-dropout) technique of Gal & Ghahramani (2016). MC-dropout estimates prediction confidence based on the statistics of numerous feed-forward passes through the network with dropout applied. To the best of our knowledge, the MC-dropout is the only non-Bayesian technique to extract a confidence score for a single network. Unfortunately, the prediction cost of an effective MC-dropout can reach hundreds of feed-forward iterations for each prediction.

Another family of confidence score functions that can be used for selective classification is based on statistics of ensembles of multiple models (Lakshminarayanan et al., 2017; Geifman et al., 2018). In the present work, we focus on selective classification with a single classifier while noting that extending our work for ensembles of selective classifiers is

straightforward and likely to improve the results.

4. SelectiveNet

In this section we first define *SelectiveNet*, a deep neural architecture allowing end-to-end optimization of selective models. In conjunction, we also propose a suitable loss function and optimization procedure to train it for any desired target coverage.

4.1. SelectiveNet Architecture

SelectiveNet is a selective model (f, g) that optimizes both $f(x)$ and $g(x)$ in a single DNN model. A schematic view of SelectiveNet is depicted in Figure 1. The input is processed by the main body block, consisting of a number of DNN layers or sub-blocks. The last layer of the main body is termed the *representation layer*. The main body block can be assembled using any type of architecture relevant to the problem at hand (e.g., convolutional, fully connected or recurrent architectures). SelectiveNet has three output heads for prediction, selection and auxiliary prediction. The role of the selection head is to implement the selection function $g(x)$. The role of the prediction head is to implement the prediction $f(x)$, and the auxiliary prediction head (denoted as $h(x)$) predicts a related prediction task in order to enrich or enforce the construction of relevant features in the main body. The auxiliary head is only used for training and its role will be clarified later after we define the loss function. The architectures of these three heads can vary depending on the task type and complexity but it is always the case that the final layer of the selection head $g(x)$ is a single neuron with a sigmoid activation. The final layer of $f(x)$ depends on the application and could be softmax (classification) or linear (regression). At inference time, a sample x is fed to SelectiveNet, which predicts $f(x)$ if and only if $g(x) \geq 0.5$; otherwise, SelectiveNet abstains from predicting the label of x.

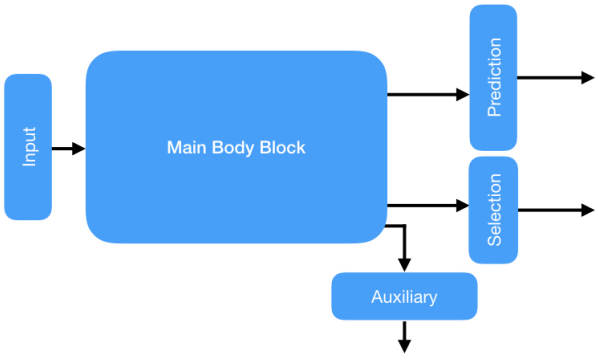


Figure 1. The SelectiveNet schematic architecture

4.2. SelectiveNet Optimization

The selective prediction objective is given in Equation (1). To enforce the coverage constraint, we utilize a variant of the well-known Interior Point Method (IPM) (Potra & Wright, 2000). This results in the following unconstrained objective, which is averaged over the samples in S_m,

$$\begin{aligned} \mathcal{L}_{(f,g)} &\triangleq \hat{r}_\ell(f, g|S_m) + \lambda\Psi(c - \hat{\phi}(g|S_m)) \\ \Psi(a) &\triangleq \max(0, a)^2, \end{aligned} \tag{2}$$

where c is the target coverage, λ is a hyperparameter controlling the relative importance of the constraint, and Ψ is a quadratic penalty function. The choice of λ is discussed in Section 6. In this work we train the auxiliary head, h, using the same prediction task as assigned to f (classification or regression) using a standard loss function,

$$\mathcal{L}_h = \hat{r}(h|S_m) = \frac{1}{m}\sum_{i=1}^{m} \ell(h(x_i), y_i),$$

which is oblivious to any coverage considerations. We optimize SelectiveNet using a convex combination of the selective loss $\mathcal{L}_{(f,g)}$ (2) and the auxiliary loss $\mathcal{L}_h$. Thus, the overall training objective is

$$\mathcal{L} = \alpha\mathcal{L}_{(f,g)} + (1 - \alpha)\mathcal{L}_h.$$

The use of the auxiliary head, h, is essential to optimizing SelectiveNet. Without h, SelectiveNet will focus on a fraction c of the training set, before accurate low level features are constructed. In such a case, SelectiveNet will tend to overfit to the wrong subset of the training set. The auxiliary head exposes the main body block to all training instances throughout the training process. For example, when taking the parameter alpha to be very small, SelectiveNet will result with poor selective risk. In all our experiment we used $\alpha = 0.5$ without any hyperparameter optimization.

5. Coverage Accuracy

When considering learning with constrained ERM as in Equation (1), we expect to see violations of the constraint over a test set. In our case, this means that the true coverage, $\phi(g)$, might be smaller than the desired target coverage c. On the other hand, even if the constraint is not violated, we are susceptible to sub-optimal models, (f, g), where $\phi(g) > c$ (the optimal selective model must satisfy the constraint with equality). When dealing with deep neural models, such coverage inaccuracies can be amplified due to the high capacity of these models.

Indeed, consider Table 1 showing the resulting test coverage obtained using both SelectiveNet and SR. Here, SelectiveNet was optimized with target coverage rates $c \in$

Target coverage	SR test coverage	SelectiveNet test coverage
0.70	60.19 ± 0.23	77.34 ± 0.05
0.75	64.16 ± 0.26	80.17 ± 0.11
0.80	68.01 ± 0.36	82.63 ± 0.12
0.85	72.05 ± 0.30	85.34 ± 0.24
0.90	76.41 ± 0.26	87.74 ± 0.32
0.95	82.30 ± 0.23	90.99 ± 0.67
Average violation	11.98	3.63

Table 1. Selective test coverage rates (%) for various trained coverage rates for SR and SelectiveNet for the Cifar-10 dataset.

$\{0.7, 0.75, \ldots, 0.9\}$, and the coverage of SR was controlled by choosing an appropriate rejection threshold over the training set. Clearly, both SR and SelectiveNet violate the target coverage rate in all cases. For example, when the target coverage is 75%, the realized coverage of SR is 64.16% and that of SelectiveNet is 80.17%.

While the coverage accuracy of SelectiveNet is significantly better than that of SR (average violation 3.625% vs. 11.98%, see Table 1), we are interested in generating selective models whose test coverage is as close as possible to the target coverage. This goal can be achieved easily using the following simple post-training coverage calibration technique, which relies on an independent *unlabeled* validation set V_n containing n samples. To calibrate to the target coverage, we estimate an appropriate threshold τ for $g(x)$ values and predict using the following decision rule

$$(f, g_\tau)(x) \triangleq \begin{cases} f(x), & \text{if } g(x) \geq \tau \\ \text{don't know}, & \text{otherwise.} \end{cases}$$

Given the validation set V_n, τ is set to be the $100(1-c)$ percentile of the distribution of $g(x_i)$, $x_i \in V_n$.

Noting that the event $g(x_i) \geq \tau$ is a Bernoulli variable, we can straightforwardly use the Hoeffding bound (Hoeffding, 1963) to bound the probability of (double sided) coverage violation greater than ϵ,

$$\mathbf{Pr}\{\epsilon\text{-violation}\} \leq 2e^{2n\epsilon^2}.$$

Equating the right-hand side to a confidence parameter δ, we get that with probability at least $1-\delta$, the resulting selective model will be in the range $[c-\epsilon, c+\epsilon]$ where $\epsilon = \sqrt{\ln(2/\delta)/(2n)}$.

6. Experimental Design and Details

This section describes the experimental details: datasets used, baseline algorithms used for comparison purposes, and our choice of architectures and hyperparamers. Our complete code can be downloaded from the following link, https://github.com/geifmany/SelectiveNet.

6.1. Datasets

Street View House Numbers (SVHN). The SVHN dataset (Netzer et al., 2011) is an image classification dataset containing 73,257 training images and 26,032 test images classified into 10 classes representing digits. The images are digits of house street numbers, which are cropped and aligned, taken from the Google Street View service. Image size is $32 \times 32 \times 3$ pixels.

CIFAR-10. The CIFAR-10 dataset (Krizhevsky & Hinton, 2009) is an image classification dataset comprising a training set of 50,000 images and 10,000 test images classified into 10 categories. The image size is $32 \times 32 \times 3$ pixels (RGB images).

Cats vs. Dogs. The Cats vs. Dogs is an image classification dataset extracted from the ASIRRA dataset. It contains 25,000 images of cats and dogs, 12,500 in each class. We randomly split this dataset into a training set containing 20,000 images, and a test set of 5000 images in a stratified fashion. We also rescaled each image to size 64x64. The average dimension size of the original images is roughly 360x400.

Concrete Compressive Strength. The Concrete Compressive Strength dataset is a regression dataset from the UCI repository (Dheeru & Karra Taniskidou, 2017). It contains 1030 instances with eight numerical features and one target value. The target value is the compressive strength of concrete, which is predicted based on its ingredients (seven features) and age (one feature).

6.2. Baseline Methods

We compare the proposed method with two baselines: SR and MC-dropout, which are described below.

Softmax Response (SR). The SR method estimates prediction confidence by the maximum softmax value activation for a given instance. According to Geifman & El-Yaniv (2017), SR, which is the most straightforward method, is a top performer in selective prediction.

Coverage	SelectiveNet risk	MC-dropout risk	% improvement	SR risk	% improvement
1.00	6.79 ± 0.03	6.79 ± 0.03	0.00	6.79 ± 0.03	0.00
0.95	$\mathbf{4.16 \pm 0.09}$	4.58 ± 0.05	8.98	4.55 ± 0.07	8.56
0.90	$\mathbf{2.43 \pm 0.08}$	2.92 ± 0.01	16.99	2.89 ± 0.03	16.14
0.85	$\mathbf{1.43 \pm 0.08}$	1.82 ± 0.09	21.35	1.78 ± 0.09	19.79
0.80	$\mathbf{0.86 \pm 0.06}$	1.08 ± 0.05	20.39	1.05 ± 0.07	17.87
0.75	$\mathbf{0.48 \pm 0.02}$	0.66 ± 0.05	26.86	0.63 ± 0.04	22.71
0.70	$\mathbf{0.32 \pm 0.01}$	0.43 ± 0.05	26.38	0.42 ± 0.06	23.88

Table 2. Classification experiment with **Cifar-10**. Selective risk (0-1% error) for various coverage rates.

Monte Carlo-dropout (MC-Dropout) (Gal & Ghahramani, 2016). Prediction confidence is quantified using variance statistics of multiple feed-forward passes of an instance through the network with dropout applied. In both classification and regression, MC-dropout was implemented as recommended by its creators. Specifically, in classification, we used a dropout rate of $p = 0.5$ and 100 feed-forward MC iterations. In regression, we used a dropout rate of $p = 0.05$ and 200 feed-forward MC iterations.

6.3. Architectures and Hyperparameters

For the convolutional neural network (CNN) experiments, we used the well-known VGG-16 architecture (Simonyan & Zisserman, 2014), optimized for the small datasets and image sizes as suggested by Liu & Deng (2015). The changes from the original VGG-16 architecture are: (i) we used only one fully connected layer with 512 neurons (the original VGG-16 has two fully connected layers of 4096 neurons). (ii) we added batch normalization (Ioffe & Szegedy, 2015) (iii) we added dropout (Srivastava et al., 2014). We used standard data augmentation consisting of horizontal flips, vertical and horizontal shifts, and rotations. The network was optimized using *stochastic gradient descent* (SGD) with a momentum of 0.9, an initial learning rate of 0.1, and a weight decay of 5e-4. The learning rate was reduced by 0.5 every 25 epochs, and trained for 300 epochs. With this setting, we reached *full coverage* validation accuracy of 96.79% for SVHN, 93.21% for Cifar-10 and 96.42% for Cats vs. Dogs. These are close to the best known results for the VGG-16 architecture.

The main body block of SelectiveNet is the VGG-16 architecture (see Figure 1). Both the prediction (f) and auxiliary heads (h) of SelectiveNet are fully-connected softmax layers. The selection head (g) is a fully connected hidden layer with 512 neurons, batch normalization and ReLU activation, followed by a fully connected layer to one output neuron with sigmoid activation. The value of α (the convex combination between the selective loss and the auxiliary loss) was set to 0.5 for all experiments, and λ was set to 32 (this value was found to be large enough to preserve the constraint through the training process).

In the regression experiment, which considers a tabular dataset from the UCI repository, we used a fully connected network as the main body block. This block consists of one hidden layer of 64 ReLU activated neurons. Both the prediction and auxiliary heads are fully connected layers with one linearly activated neuron. The selection head g was computed with one hidden layer, with 16 ReLU activated neurons, followed by a fully connected layer with one Sigmoid activated neuron. Batch normalization was applied in every hidden layer on all modules (main body block, prediction, auxiliary and selection heads). The regression model was optimized using the ADAM algorithm (Kingma & Ba, 2014) with a learning rate of 5×10^{-4} and mini-batch size of 256 and 800 epochs. We used squared loss with weight decay of 1×10^{-4} during optimization.

7. Experiments

In this section we report on the results of several experiments that validate the effectiveness of SelectiveNet. We begin with three image classification experiments where the main body block (see Section 4) has a CNN architecture. We then experiment with a regression task where the main body block comprises a fully connected architecture.

7.1. Selective Classification

We consider the Cifar-10 dataset first. We trained a version of SelectiveNet whose main body block is based on the VGG-16 architecture; see the full description of our architecture and implementation details in Section 6. We compared calibrated SelectiveNet instances, which were trained with coverage rates $c \in \{0.7, 0.75, 0.8, 0.85, 0.9, 0.95\}$ to two standard selective classifiers over these coverage rates. The standard classifiers were constructed using the well-known SR and MC-dropout confidence estimates applied to a trained network consisting of a multi-class head (identical to f in SelectiveNet) on top of the same main body block of SelectiveNet. The standard model was trained on Cifar-10 at full coverage as proposed in (Geifman & El-Yaniv,

Coverage	SelectiveNet risk	MC-dropout risk	% improvement	SR risk	% improvement
1.00	3.21 ± 0.08	3.21 ± 0.08	0.00	3.21 ± 0.08	0.00
0.95	1.40 ± 0.01	1.40 ± 0.05	0.00	1.39 ± 0.05	-0.77
0.90	$\mathbf{0.82 \pm 0.01}$	0.90 ± 0.04	9.05	0.89 ± 0.04	8.47
0.85	$\mathbf{0.60 \pm 0.01}$	0.71 ± 0.03	15.26	0.70 ± 0.03	13.61
0.80	$\mathbf{0.53 \pm 0.01}$	0.61 ± 0.01	14.07	0.61 ± 0.02	14.07

Table 3. Classification experiment with **SVHN**. Selective risk (0-1% error) for various coverage rates.

Coverage	SelectiveNet risk	MC-dropout risk	% improvement	SR risk	% improvement
1.00	3.58 ± 0.04	3.58 ± 0.04	0.00	3.58 ± 0.04	0.00
0.95	$\mathbf{1.62 \pm 0.05}$	1.92 ± 0.06	15.40	1.91 ± 0.08	15.09
0.90	$\mathbf{0.93 \pm 0.01}$	1.10 ± 0.05	16.13	1.10 ± 0.08	15.56
0.85	$\mathbf{0.56 \pm 0.02}$	0.78 ± 0.06	28.30	0.82 ± 0.06	32.40
0.80	$\mathbf{0.35 \pm 0.09}$	0.55 ± 0.02	36.39	0.68 ± 0.05	48.16

Table 4. Classification experiment with **Cats vs. Dogs**. Selective risk (0-1 % error) for various coverage rates.

2017). The results are presented in Table 2. It can be seen that SelectiveNet significantly outperforms the two baseline methods (SR and MC-dropout). The relative advantage of SelectiveNet varies from 8.5% at 0.95 coverage compared to SR, and a maximal advantage of 26.8% compared to MC-dropout at 0.75 coverage. It is also evident that the MC-dropout and SR performance is quantitatively similar as was already reported in (Geifman & El-Yaniv, 2017).

We turn now to the SVHN dataset. Here again, we trained SelectiveNet with the same VGG-16-based architecture for coverage rates $c \in \{0.8, 0.85, 0.9, 0.95\}$ (coverage rates smaller than 0.8 result in nearly perfect models in this dataset). The results in Table 3 show SelectiveNet's significant superiority for all coverage rates, with the exception of the 0.95 coverage, where all methods are statistically indistinguishable. The maximal relative advantage over this dataset is 14.07% for 0.8 coverage rate. We also observe that here again, the performance of SR and MC-dropout is similar for all coverage rates.

Finally, for the Cats vs. Dogs dataset, whose input image size is larger (64×64), we adapted the same architecture (to this image size) for all the contenders and repeated the same experiments with the same coverage rates as in the SVHN experiment. Presented in Table 4, the results are unsurprising and again indicate a consistent and significant advantage of SelectiveNet over the two baselines.

Risk-coverage curves depicting the performance of SelectiveNet and SR for the Cifar-10 and Cats vs. Dogs datasets appear in Figure 2. These curves nicely demonstrate the consistent advantage of SelectiveNet.

7.2. Selective Regression

We ran the regression experiment over the Concrete Compressive Strength dataset from the UCI repository. For this set we trained a SelectiveNet variant adapted for a regression of tabular data. The main body block is now a fully connected neural network and the prediction heads utilize a linear unit as described in Section 6. The results appear in Table 5. Clearly, the SR baseline cannot be applied in this setting because there is no softmax layer in regression networks. We, therefore, compared SelectiveNet only to the MC-dropout method, currently the only non-ensemble/Bayesian method for deep selective regression. In this task, SelectiveNet exhibits a distinct advantage over MC-dropout for all coverage rates below 0.9. There, SelectiveNet significantly outperforms MC-dropout, with a relative advantage that varies between 26.81% to 30.48%.

8. Empirical Observations

We now empirically analyze some properties of SelectiveNet, related to calibration and representation structure.

8.1. Coverage Calibration

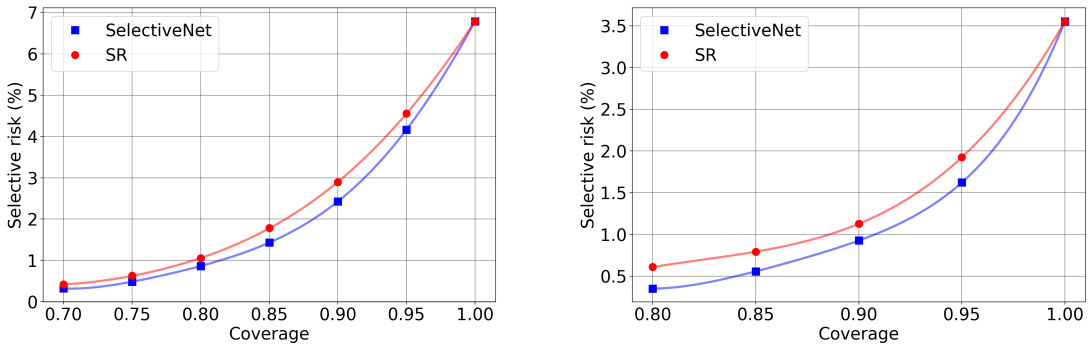
We evaluate the effectiveness and limitations of the post-training coverage calibration technique. We used this calibration to evaluate five models for Cifar-10 trained with five different coverage rates, $c \in \{0.7, 0.75, 0.8, 0.85, 0.9\}$. We calibrated each model to match each coverage value. Table 6 is the resulting coverage "confusion matrix" where element (i, j) in the table is the selective risk of model i calibrated to coverage j. For example, the third value in the last column (2.40) is the selective risk of SelectiveNet

Coverage	SelectiveNet risk	MC-dropout risk	% improvement
1.00	38.45 ± 0.90	39.01 ± 0.70	1.43
0.90	35.35 ± 1.31	36.97 ± 0.46	4.39
0.80	$\mathbf{30.48 \pm 0.93}$	35.53 ± 0.61	14.23
0.70	$\mathbf{27.94 \pm 1.12}$	33.70 ± 0.58	17.09
0.60	$\mathbf{27.12 \pm 0.99}$	31.23 ± 0.71	13.18
0.50	$\mathbf{26.81 \pm 1.36}$	28.90 ± 0.77	7.23

Table 5. Regression experiment with **Concrete Compressive Strength**. Selective risk (MSE) for various coverage rates.

	Calibration				
Training	0.70	0.75	0.80	0.85	0.90
0.70	$\mathbf{0.32 \pm 0.01}$	0.55 ± 0.00	0.88 ± 0.05	1.48 ± 0.05	2.65 ± 0.11
0.75	$\mathbf{0.31 \pm 0.04}$	$\mathbf{0.48 \pm 0.02}$	$\mathbf{0.85 \pm 0.04}$	1.47 ± 0.07	2.63 ± 0.06
0.80	$\mathbf{0.34 \pm 0.01}$	0.55 ± 0.04	$\mathbf{0.86 \pm 0.06}$	$\mathbf{1.42 \pm 0.05}$	$\mathbf{2.40 \pm 0.09}$
0.85	0.37 ± 0.02	0.54 ± 0.03	$\mathbf{0.84 \pm 0.04}$	$\mathbf{1.43 \pm 0.08}$	$\mathbf{2.47 \pm 0.07}$
0.90	0.51 ± 0.12	0.58 ± 0.09	$\mathbf{0.79 \pm 0.03}$	$\mathbf{1.35 \pm 0.03}$	$\mathbf{2.43 \pm 0.08}$

Table 6. Selective risk of SelectiveNet trained on the Cifar-10 dataset where rows represent a training coverage value and columns represent post-training calibration coverage values. All results were multiplied by 100 and correspond to % of errors.



(a) Cifar-10 (b) Cats vs. dogs

Figure 2. Risk coverage curves comparing SelectiveNet and SR over (a) Cifar-10 and (b) Cats vs. Dogs

that was optimized with coverage 0.8 during training and then calibrated for coverage 0.9. All the numbers inside the table were multiplied by 100 and reflect the error percent. Each selective risk in the table is an average over three independent runs.

The minimal element in each column appears in bold red, and all other elements in that column, which have one standard error overlap with it, appear in bold red as well. Thus, the red elements in each column represent all nearly-optimal errors in their column. Clearly, the diagonal of this matrix is bold red, indicating that the calibrated SelectiveNet, which was trained for coverage c, is nearly optimal among the other SelectiveNets that were trained with coverage rates other than c. Another interesting observation is that there is no significant compromise due to the calibration process, provided that we calibrate towards a coverage rate that is close to the trained (target) coverage. The fact that the off-diagonal elements often tend to admit inferior selective risks indicates that the calibrated SelectiveNet effectively optimizes its coverage-specific selective risk.

8.2. Learned Representation

It is interesting to inspect the representation learned by SelectiveNet and those learned by standard methods such as SR or MC-dropout. Considering SelectiveNet's architecture, one may expect that the main body block will produce fea-

(a) SelectiveNet (b) Softmax response (c) Legend

Figure 3. t-SNE visualization of the embedding representation of (a) SelectiveNet and (b) SR.

tures that are effective for both the prediction and rejection tasks. In contrast, the standard method relies on standard models that are oblivious to coverage considerations. To this end, we use the well-known t-Distributed Stochastic Neighbor Embedding (t-SNE) technique (Maaten & Hinton, 2008) to visualize the learned representation layer of SelectiveNet and SR that were trained for the Cifar-10 dataset with $c = 0.7$. We computed the t-SNE transformation of the embedding representation consisting of all 512 neurons in the embedding layer. The results are depicted in Figure 3. In Figure 3(b) we see the t-SNE of the SR model. Each color in the figure corresponds to a class label (among the 10 classes in Cifar-10). For example, the turquoise cluster is 'dog' and the orange cluster corresponds to 'cat'. Rejected instances appear as dots in bold colors and the covered points are faded dots. Similarly, Figure 3(a) presents the t-SNE of SelectiveNet.

The striking observation is that the SR clusters are relatively well separated regardless of rejection but, in contrast, the rejected instances in the SelectiveNet case are very weakly separated. For example, we see severe confusion between rejected points of classes 'airplane' (maroon), 'ship' (light brown), and 'bird' (blue). Similarly, we see that the rejected instances of the rest of the animals ('cat', 'dog', 'deer', 'horse' and 'frog') are not well separated as well. A plausible interpretation of this phenomenon is that SelectiveNet is not wasting representational capacity to separate between rejected points, which may leave more capacity to learn relevant features in order to separate non-rejected points. Another observation is that most of the rejected points were positioned in a central cluster. This structure can make it easier for these points to be captured by the selection function g.

9. Concluding Remarks

We presented SelectiveNet, an effective deep model for selective classification and regression. The power of this new model stems from the mutual training of both the selection and prediction models within the same network, thus forcing the models to focus on the more relevant instances that will not be rejected in production. Known techniques, such as MC-dropout or SR, which typically compose a rejection mechanism over a standard prediction model (trained with full coverage), cannot benefit from this advantage. Our empirical results indicate that the new model currently provides the most accurate classification for a given abstention rate. For selective *regression*, SelectiveNet provides the best and only fast solution, where the only other (inferior) alternative is MC-dropout. This result motivates the use of SelectiveNet in various deep regression applications that require fast inference, such as visual tracking or detection.

We leave a number of issues for future research. As is the case in many deep learning applications, also in selective prediction, ensembling techniques improve performance of basic models. In Lakshminarayanan et al. (2017), it was recently shown that an ensemble of 2-15 models can improve upon the basic SR rejection method. It would be interesting to check if ensembling can improve SelectiveNet as well. Another interesting direction is to examine the influence of capacity (architecture choice) of the selection function g on the overall performance of SelectiveNet. Here, we have not tried to optimize this choice, but we anticipate that different datasets and perhaps different coverage rates may require different architectures also for the selection head.

Acknowledgments

This research was supported by The Israel Science Foundation (grant No. 81/017).

References

Chow, C. K. An optimum character recognition system using decision functions. *IRE Transactions on Electronic Computers*, (4):247–254, 1957.

Cordella, L. P., De Stefano, C., Tortorella, F., and Vento, M. A method for improving classification reliability of multilayer perceptrons. *IEEE Transactions on Neural Networks*, 6(5):1140–1147, 1995.

Cortes, C., DeSalvo, G., and Mohri, M. Boosting with abstention. In *Advances in Neural Information Processing Systems*, pp. 1660–1668, 2016.

De Stefano, C., Sansone, C., and Vento, M. To reject or not to reject: that is the question-an answer in case of neural classifiers. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 30 (1):84–94, 2000.

Dheeru, D. and Karra Taniskidou, E. UCI machine learning repository, 2017. URL http://archive.ics.uci.edu/ml.

El-Yaniv, R. and Wiener, Y. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11:1605–1641, 2010.

El-Yaniv, R. and Wiener, Y. Agnostic pointwise-competitive selective classification. *Journal of Artificial Intelligence Research*, 52:171–201, 2015.

Fumera, G. and Roli, F. Support vector machines with embedded reject option. In *Pattern recognition with support vector machines*, pp. 68–82. Springer, 2002.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, pp. 1050–1059, 2016.

Geifman, Y. and El-Yaniv, R. Selective classification for deep neural networks. In *Advances in neural information processing systems*, pp. 4878–4887, 2017.

Geifman, Y., Uziel, G., and El-Yaniv, R. Bias-reduced uncertainty estimation for deep neural classifiers. 2018.

Hellman, M. E. The nearest neighbor classification rule with a reject option. *IEEE Transactions on Systems Science and Cybernetics*, 6(3):179–185, 1970.

Hoeffding, W. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, March 1963. URL http://www.jstor.org/stable/2282952?

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

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

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. 2009.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.

Liu, S. and Deng, W. Very deep convolutional neural network based image classification using small training sample size. In *Pattern Recognition (ACPR), 2015 3rd IAPR Asian Conference on*, pp. 730–734. IEEE, 2015.

Maaten, L. v. d. and Hinton, G. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov): 2579–2605, 2008.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, volume 2011, pp. 5, 2011.

Potra, F. A. and Wright, S. J. Interior-point methods. *Journal of Computational and Applied Mathematics*, 124(1-2): 281–302, 2000.

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

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Wiener, Y. and El-Yaniv, R. Pointwise tracking the optimal regression function. In *Advances in Neural Information Processing Systems (NIPS)*, pp. 2042–2050, 2012.