# Abstention Protocol for Accuracy and Speed

Amirata Ghorbani*
EE Dept.
Stanford University
amiratag@stanford.edu

Yasaman shirian*
Me Dept.
Stanford University
yshirian@stanford.edu

## Abstract

In order to confidently rely on machines to decide and perform tasks for us, there needs to be classifiers with reliably high accuracy. One of the challenges for having an accurate classifier is that the scarcity, quality, or richness of the training data can limit the Bayes upper bound of the accuracy. The second challenge is that having high accuracy could require complex and slow classifiers while fast and simple classifiers suffer from low accuracy. In this work, we have investigated abstention as a solution to both of the mentioned challenges. Abstention allows us to find how difficult if the task at hand for the classifier. Abstaining from a difficult task would result in higher accuracy. Abstaining from extra computation for an easy task would save computational time.

One solution for the problem of accuracy control for machine learning models is to abstain the from prediction where there is uncertainty. For the problem of speed, propose a method of constructing a classifier, which accuracy can be adjusted, i. e. based on the computational resources available, it can have different accuracy. The idea is simple: most samples are easy to classify and therefore do not need the full computational effort of the classifier to be classified.

In what follows, we first describe existing abstention methods. Then we examine several models behavior in the presense of abstention. At last we propse an algorithm to increase the speed of inference in deep neural networks using abstention.

## 1 Introduction

Recent advances in machine learning has set new performance standards. However, in some applications the knowledge deficit and confounding factors in the available training data put limits on prediction certainty of machine learning systems; applications like disease prediction [14] or protein-DNA binding prediction [7]. For instance,data available based on Electronic Health Record(EHR) for predicting patients health condition is often reported partially and complementary information is not obtainable.

Even in the presence of reliable training data, the state-of-the-art model might not be computationally feasible to implement [16]; consequently, there's a need to implement smaller models with lower accuracy.

*Equal contribution In experiments and write-Up.

## 2 Related Work

The idea of classifiers with rejection option was first introduced by [2] in 1970. There has been several papers in the area of accelerating the netural network classifiers. Denil [4] demonstrated large redundancy in neural networks. Exploiting this feature, there has emerged a new line of research to train high speed compressed networks. Ba & Caruana [1] compressed deep networks into single layer networks. Lebedev [11], Jagerberg [9] used matrix decomposition techniques to speed up CNNs. The common issue with the mentioned works is, despite reduced size and increased speed, they still propose a fixed size model for all input samples.

# 3 Methods

## 3.1 Problem Statement

Given:

- A trained Classifier $\mathscr{F} : \mathbb{R}^d \rightarrow \mathbb{R}^C$ with $C$ classes
- sample $\mathbf{x}_t$ from test distribution with true label $y_t$
- abstention/shortcut function $\mathscr{D} : \mathbb{R}^C \rightarrow \{0, 1\}$ (Abstain from prediction/Shortcut prediction if $D(\mathscr{F}(\mathbf{x}_t)))$
- Desired accuracy $A_0$
- Test sample inference time: $t(\mathbf{x}_t; \mathscr{F})$

The **Accuracy Control Problem** is defined as:

$$\arg\min_{D} \mathbb{E}[D(\mathscr{F}(\mathbf{x}_t))]$$

$$\text{subject to : } Pr[\mathscr{F}((\mathbf{x}_t))] > A_0 given D(\mathscr{F}(\mathbf{x}_t)) = 1$$

which means that we want to have the minimum abstention(rejection) rate for having the desired accuracy. The **Speed Control Problem** is defined as:

$$\arg\min_{D} \mathbb{E}[t(\mathbf{x}_t; \mathscr{F})]$$

$$\text{subject to : } Pr[\mathscr{F}((\mathbf{x}_t))] > A_0$$

which means that we want to have the minimum computation time for having the desired accuracy.

## 3.2 Abstention Methods

In the most recent state of the art architectures ( [10], [8]) there exists a softmax layer as the last layer of the classifier. Consequently, the classifier's output would be a probability vector $\mathbf{P}$ over class labels. Even for simple classifiers like SVM or random forest, the scores can be converted to probability distribution between classes. [13]

- **Confidence Abstention**: The most simple abstention method would be to reject any sample which winning class confidence score is less than a threshold. In [5], however, was shown that for optimal rejection-accuraacy trade-off there needs to be different thresholds for different classes. For classification tasks with few number classes this could be tractable, however, in tasks like Imagenet [3], it is not efficient to search for 1000 thresholds, one for each class.
- **Entropy Abstention**: As mentioned, drawback of the confidence abstention was the need for class-specific threshold. Therefore, as a better abstention criterion, we use entropy of the output probability vector:

  $$H(\mathbf{P}) = \sum_{i=1}^{C} P_i log(P_i)$$

  Entropy as an uncertainty measure takes into account the confidence scores of all of the classes unlike the confidence abstention where only the winning class confidence is considered. In other words, for or two probability vectors with equal winning class confidence score, the entropy would not be the same; e.g.

  $$H([0.01, 0.47, 0.52]) \neq H([0.24, 0.24, 0.52])$$

- **Dropout Abstention** Gal Ghahramani [6], introduced a novel method for uncertainty approximation in neural networks where the prediction task is performed several times while having dropout in all layers of the neural network. The result would be several probability vectors for the same test sample. Using the mean and variance of predictions, one can estimate the prediction uncertainty and therefore the confidence bounds of the prediction score. The drawback, however, is that this method requires several times more computational power. Performing the classification task on each sample $M$ times while having dropout after each layer will result in having $M$ confidence vectors $\mathbf{P}_1, \ldots, \mathbf{P}_M$. There are several ways of using there probability vectors for abstention:
  - We can take the **mean** of the vectors and then abstain just as the confidence method or the entropy method. The drawback, however, is that the information about variation of scores is not utilized,
  - We can add the variance of each class's score to its average and get an Upper Confidence

Bound(**UCB**) of the prediction confidence. We can also have a lower confidence bound(**LCB**) by subtracting the variance from mean,

– Taking into account that variance of the prediction stands for **uncertainty**, we can reject samples that have high uncertainty.

### 3.3 Shortcut Method

Recent state-of-the-art classifiers consist of large number of layers. It has been shown( [15]) that neural networks extract features layer by layer. Therefore, we expect the separability of data between different classes to increase as we go deeper into the network.

Different test samples have various levels of difficulty to be classified. It is expected for easier samples to be classified correctly without the need of profound feature extraction of a deep network. In other words, using the features extracted in the very first layers would be enough for classifying part of the test distribution. Figure 1 displays the two dimensional projection of MNIST data set using t-SNE [12]. It's clear that for most part of the data set even a simple linear classifier would be able to separate different classes.

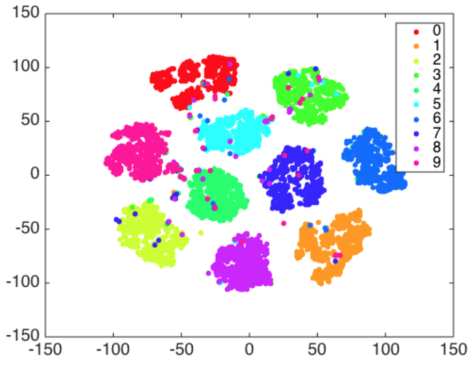


Figure 1: MNIST t-SNE

As a consequence, we can shortcut the prediction of the network for easy samples. The question that arises would be to measure how *easy* a sample is. In this work, our answer is to use abstention.

For each test sample, after computing the operations of first layer (simplest level of feature extraction), the first shortcut (a small fully-connected network) performs the classification task. If the classification result more *confident* than a certain threshold, the sample is classified. If not, the sample goes through the next layer and so on. Figure 3 gives a better understandnig of what we described.

First question would be that the each shortcut layer itself adds to the computational effort. First of all, the number of operations in a shortcut network compared to the original network's number computational operations is negligible in most state-of-the art classifiers. Secondly, in practice, shortcuts are used only in between every few number of layers instead of every layer.

Second question, is the measure to detect whether a sample is confidently classifier at each shortcut. In this work, we used Entropy. At each shortcut, if the entropy of the output probability vector over classes $\mathbf{P}$ is smaller than a threshold, the sample is not abstained from the shortcut and therefore the classification is finished without computing next layers.

Last question to be answered, is how to find the entropy thresholds at different shortcuts. It should be mentioned that the thresholds are not independent. If we increase the threshold of the first shortcut, more samples would be classified at first layer which means that more difficult samples would reach to the second shortcut and therefore its threshold should be adapted. Because of these dependencies, in this project, we used grid search to find the best thresholds. Figure 2 describes the result of grid search for a 5 layer CNN trained on CIFAR10 with original accuracy of 82%. Algorithm 1 describes the specifics. (Figure 3).

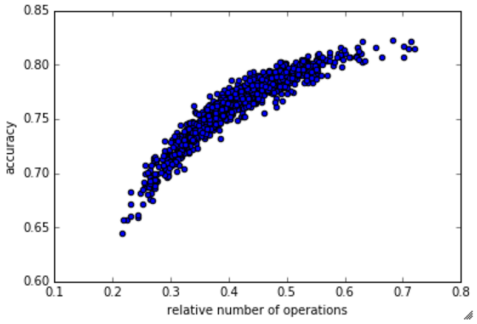


Figure 2: Short-circuiting a 5-layer CNN classifier

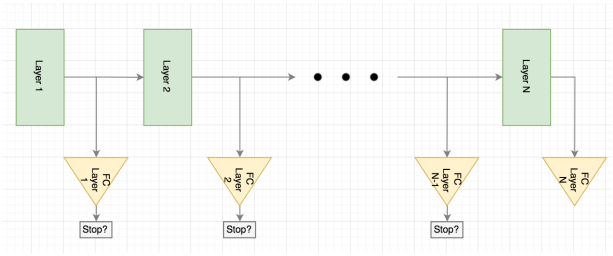


Figure 3: **Speed Control**: Adding short-circuit networks in the middle of deep neural network classifiers

# 4 Experiments

Table 1 describes data sets.

## 4.1 Accuracy Control in Support Vector Classifiers and Random Forest

As a baseline example, we use confidence abstention for Support Vector Classifier because of their power in high dimensional space and Random Forest for as a widely-used baseline in mid-sized and small data sets. For SVMs we have the choice of using kernels to have efficient computations. As displayed in Figure 4, even for these simple classifiers, any desirable accuracy is achievable with the if enough portion of test samples rejected.

## 4.2 Acccuracy Control in Neural Networks

We examined several abstention methods. Entropy abstention was discussed above. There are several ways to implement dropout abstention.

Examining different values for the number of test time forward passes of the dropout abstention method($M$), we realize that performance increases from $M = 1$ to $M = 20$ and remains the same for larger $M$ values. Examining different dropout probabilities, we find that $p_{\text{drop-out}} = 0.5$ results in the best rejection-accuracy trade-off. Final results are depicted in Figure 4.

As the results imply, using LCB, UCB or Mean yields to slightly better results compared to entropy abstention while having 20 times more computational cost. (Using just the uncertainty of prediction yields to the worst result.)

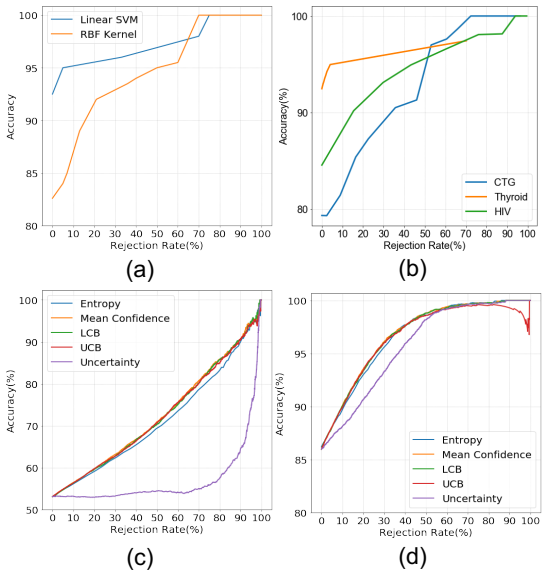


Figure 4: **Accuracy-Rejection Trade-off :** (a) The accuracy-rejection rade-off for the two data sets trained on linear and RBF SVMs using confidence abstention. (b) The accuracy-rejection rade-off for the three data sets trained on random forest tuned using 5-fold cross validation. (c) Result for several abstention methods for a feed-forward neural network classifier trained on CIFAR 10 data sets. (d) Result for several abstention methods for a feed-forward neural network classifier trained on Fashion MNIST data set.

## 4.3 Speed Control

In Figure 6, the speed control (best trade-off between accuracy and number of computational operations) by using shortcuts in a VGG16 architecture [16] (Figure 5) trained on CIFAR10 data set with 88% original accuracy is displayed. We used 5 shortcuts with 1024 hidden neurons each. As depicted in the plot, with half of the original number of operations, we can achieve 95% of the original accuracy. In other words the classification becomes twice faster while accuracy drops from 88% to 84%.

| Data Set | Number of Attributes | Number of Instances | Number of Classes | Setting |
|---|---|---|---|---|
| Cardiotocography(CTG) | 23 | 2126 | 10 | One Vs Rest Linear SVM, Random Forest |
| Thyroid Disease | 21 | 7200 | 3 | One Vs Rest RBF Kernel SVM, Random Forest |
| Fashion MNIST | 784 | 50000 | 10 | Two hidden layer Neural Network |
| HIV | 160 | 7000 | 2 | Random Forest |
| CIFAR10 | 3072 | 50000 | 10 | Three hidden layer neural network |

Table 1: **Data Sets**: The details of the data sets we used in our experiments


Figure 5: **VGG16 structure**

# 5 Conclusion & Future Work

We examined the abstention protocol for accuracy and speed control. We showed that abstention can be used in wide range of classifiers in the sense that by rejecting difficult test samples, the accuracy for the remaining test samples can increase arbitrarily. Accuracy control is necessary for applications such as disease prediction where avoiding from mistake is more important than being able to predict disease for every patient.

We, then, proposed an algorithm that using abstention tries to avoid from *over-classification* of easy samples. It was shown that using the proposed algorithm, one could exploit a trade-off between speed and accuracy of a classifier. The importance of this algorithm is that in settings where the local computational power is limited (e.g. mobile phone), most of the task could be handled locally and in the case of a difficult task, the built-in classifier could query a more complex in-sever classifier.

So far, all the abstention methods we discussed are test-time abstention methods that are applied on an already trained classifier(Dynamic Abstention). One other line of research would be to examine incorporating abstention as a criterion in the the training phase of a classifier. (Static Abstention) on protocol for utilization of abstention. We a better algorithm for finding entropy thresholds in the proposed speed control algorithm as the current method is not scalable.

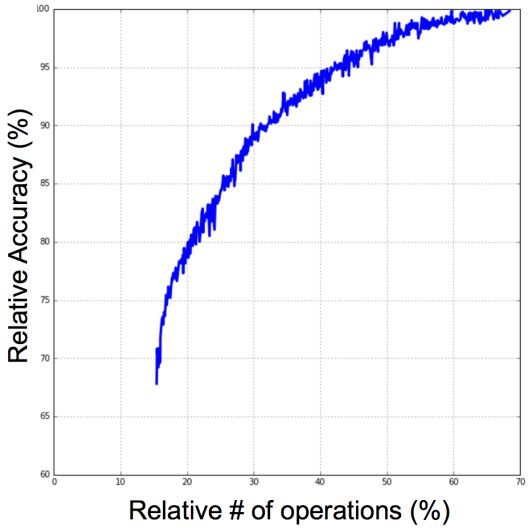


Figure 6: **Speed control for VGG16 architecture trained on CIFAR10.** The curve depicts the smallest number of operations compared to the original network's number of operations for a any given accuracy.

# Appendix: Speed Control Algorithm

input sample $\mathbf{x} \in R^D$, A N layer network with layers: $L_1, L_2, \ldots, L_N$, an entropy threshold For each layer's shortcut : $T_1, T_2, \ldots, T_N$;
i = 1;
**while** *classification not finished* **do**
  Activations of the i'th layer are calculated and are passed through a shortcut fully connected network to get probability vector $\mathbf{P}^i()$;
  **if** $H(\mathbf{P_i}) < T_i$ *or* $i = N$ **then**
    Classification done;
    $C(x) = arg\max_{j=1,\ldots,C} P_{ij}$;
  **else**
    $i = i + 1$
  **end**
**end**

**Algorithm 1:** Adding shortcuts to deep networks

# References


[1] J. Ba and R. Caruana. Do deep nets really need to be deep? *NIPS*, 2014.

[2] C. K. Chow. On optimum error and reject trade-of. *IEEE Transactions on Information Theory*, 16:41–46, 1970.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[4] M. Denil, B. Shakibi, L. Dinh, N. de Freitas, and . Predicting parameters in deep learning. *NIPS*, 2013.

[5] G. Fumera, F. Roli, and G. Giacinto. Reject option with multiple thresholds. *The journal of the pattern recognition society*, 33:2099–2101, 2000.

[6] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016.

[7] G. L. D. K. G. Haoyang Zeng, Matthew D. Edwards. Convolutional neural network architectures for predicting dnaprotein binding. *Bioinformatics, Volume 32, Issue 12, 15*, 2016.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[9] M. Jaderberg, A. Vedaldi, and A. Zisserman. Speeding up convolutional neural networks with low rank expansions. *arXiv preprint arXiv:1405.3866*, 2014.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *NIPS*, 2012.

[11] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky. Speeding-up convolutional neural networks using fine-tuned cp-decomposition. *arXiv preprint arXiv:1412.6553*, 2014.

[12] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(Nov):2579–2605, 2008.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[14] B. K. J. D. R Miotto, L Li. Deep patient: An unsupervised representation to predict the future of patients from the electronic health records. *Scientific Reports. 2016;6:26094.*, 2016.

[15] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *CoRR*, abs/1312.6034, 2013.

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