Learning the Threshold of Labeling Functions in Snorkel

Xuefei Cao

December 11, 2018

1 Introduction

In supervised machine learning approaches, one major limitation is the requirement of high quality labeled data. With the recent development of deep learning methods, how to acquire large-scale data with fully ground-truth labels is becoming a main issue. Most deep learning models require a large amount of labeled data to avoid overfitting. However, creating these labels by hand is often too slow and expensive. Although it is usually difficult to obtain large-scale hand-labeled data, abundant unlabeled data such as image data is relatively easy to download from the internet. Weak supervision technique is thus proposed to deal with this situation. Distant supervision [Mintz et al., 2009, Takamatsu et al., 2012] is one type of weak supervision technique. In distant learning, we utilize existing labeled data or database that is close to our current tasks. Another way to achieve weak supervision is to use weak classifiers (crowdsourcing) [Sheng et al., 2008, Urner et al., 2012].

In Ratner et al., they proposed a new framework Snorkel to train the unlabeled data with labeling functions. Labeling functions are usually given by domain experts. Specifically, they introduced a three-step procedure: writing labeling functions, learning a generative model over the labeling functions and training a discriminative model. Although Snorkel provides a new method to generate large-scale labeled data, it also brings the challenge that users need to provide labeling functions with enough weak supervision information. For example, for the image classification tasks of detecting whether a person is riding a bike, we have d, the distance between the person and bike. One of the labeling function can be defined as $(2 \times \mathbb{1}_{d>\alpha} - 1)$, where α can be given any value by users. Another intuitive example is bone tumor classification from medical images [Varma et al., 2017]. In this example, one of the labeling functions can be based on the area of the tumor in the image. By selecting different threshold if one exists, we can potentially obtain different evaluation accuracy for some tasks. Motivated by this observation, in this project, we will seek an automatic method to select the threshold for the labeling functions.

While cross-validation can be a possible way to determine reasonable cutoff inside of labeling functions, it often requires additional labeled data to verify our chosen parameters. In addition, the grid search method often does not scale well when the dimension of parameters increases. For example, if the number of labeling functions (with thresholds) is n and we choose m values for each parameter, we will need to estimate n^m different models. The idea we investigated is the derivative-free optimization method [Nocedal and Wright, 2006]. The reason we choose derivative-free optimization methods is that the indicator function is not differentiable which makes it difficult to pursue a pure gradient-based method. To evaluate our methods, we explored both synthetic dataset and real-world dataset. For the real-world dataset, we choose Microsoft COCO dataset [Lin et al., 2014] and define our task as finding images of a person biking down a road [Varma et al., 2017].

In this paper, we propose a method to choose the threshold automatically without requiring the verification of additional labeled data. We also compared the results of our method with a baseline method for which the threshold is chosen randomly. The remainder of the paper is organized as follows. We start with a brief review of Snorkel project and its theoretical background. We then introduce our method in section 3. In section 4, we simulated a simple dataset from a Gaussian Mixture model and showed that our method outperformed the baseline method in terms of both F1 score and accuracy. Furthermore, an image classification task was conducted to verify our introduced method. The related work section and discussion of limitations can be found at the end of this paper.

2 Background

There have been some rapid developments in supervised machine learning techniques by creating predictive models and learning from a large amount of labeled training data. Though these methods have achieved great success, it is worth mentioning that in many situations, it is difficult to get strong supervision information and the data acquiring process usually requires a lot of human efforts [Zhou, 2017]. It is becoming more and more popular to obtain training labels from noisy labels or background knowledge. In this paper, we will focus on the method proposed in [Ratner et al., 2016, Bach et al., 2017, Ratner et al., 2017]. They introduced the concept of labeling functions. Based on the labeling functions, they proposed to estimate the posterior probability of labels from the generative model (data programming).

Assume we have data points $x_1, x_2, ..., x_n$ and latent true labels $y_1, ..., y_n$. For the real data used in our paper, x can be images collected from Microsoft COCO dataset while y indicates whether the image contains a person riding bike down a road. Previous weak supervision studies have discussed several labeling methods including extracting labels from other related dataset and collecting labels from crowdsourcing. Ratner et al. proposed to use labeling functions to introduce weak supervision. In particular, we have users to define several labeling functions λ_1, λ_2, ..., λ_m. For each data point x_i, every labeling function λ_j will produce outputs $\Lambda_{i,j}$ (1, -1 or 0). The labeling functions are user-defined that encode some domain heuristic knowledge. The labeling functions can vote to abstain or 0 if they are uncertain about classifications for some examples. In Ratner et al., the labeling function outputs are assumed conditionally independent given the true label. The relationship between Λ and y is governed by accuracy dependencies.

$$\phi_{i,j}^{Acc} = \mathbb{1}_{\{\Lambda_{i,j}=y_i\}} \tag{1}$$

The generative model can thus be specified as

$$p_w(\Lambda, Y) = Z_w^{-1} e^{\sum_{i=1}^{n} w^T \phi_i(\Lambda, y_i)} \tag{2}$$

where Z_w is a normalizing constant and $\phi_i(\Lambda, y_i)$ is the concatenation of outputs from all labeling functions.

Though the conditionally independent model is a common assumption, statistical dependencies are very common in practice, and it is desirable to learn the structure of generative models automatically rather than depending on users to provide correlation structures. In Bach et al., an efficient method for automatically learning the structure of the generative model from unlabeled training data alone was proposed. In their model, labeling function dependencies are added such as correlation dependencies of the form

$$\phi_{i,j,k}^{Corr} = \mathbb{1}_{\{\Lambda_{i,j}=\Lambda_{i,k}\}} \tag{3}$$

Higher-order factors that connect multiple labeling function outputs have also been discussed in Bach et al.. In Ratner et al., for simplicity, they only include the labeling propensity, accuracy, and pairwise correlations of labeling functions. Labeling propensity can be specified as

$$\phi_{i,j}^{Lab} = \mathbb{1}_{\{\Lambda_{i,j} \neq 0\}} \tag{4}$$

The parameters can be estimated by minimizing the negative log marginal likelihood $p_w(\Lambda)$.

$$\hat{w} = \underset{w}{\operatorname{argmin}} - \log(\sum_{Y} p_w(\Lambda, Y)) \tag{5}$$

This objective can be optimized by combining stochastic gradient descent methods and Gibbs Sampling [Hinton, 2002].

Although the labeling function is an innovative way to introduce weak supervision information to unlabeled data, how to define labeling functions is still an unresolved issue. In the rest of this paper, we will start with a small step towards this issue and focus on the problem of selecting the threshold in labeling functions. In the Snorkel framework, the labeling functions are assumed to be defined by domain experts. However, for some problems, it may not be clear whether a threshold in the labeling function is suitable.

3 Method

The parameters can be estimated by minimizing $l(w, \alpha)$

$$\hat{w}, \hat{\alpha} = \underset{w,\alpha}{\operatorname{argmin}} - \log(\sum_{Y} p_w(\Lambda_\alpha, Y)) \tag{6}$$

where labels provided by labeling functions can depend on additional parameter α (thresholds). This objective function does not vary continuously when α is perturbed, thus we propose an (ad-hoc) algorithm to update w and α iteratively. In particular, we first update w when α is fixed using the algorithm proposed in [Ratner et al., 2017]. As a second step, we optimize with respect to α using a derivative free optimization method (Nelder-Mead method [Nelder and Mead, 1965]). These two steps are repeated until convergence. Our algorithm is summarized in 1.

Algorithm 1 Estimation for our method

Input: data X, Labeling functions LF_1, ..., LF_n
Initialize α
repeat
  Obtain weak labels from labeling functions
  Update w of $l(w, \alpha)$ given α by by combining stochastic gradient descent methods and Gibbs Sampling.
  Solve for the minimizer α of $l(w, \alpha)$ given w using appropriate derivative free optimization method (Nelder-Mead method are chosen in our paper)
until convergence
Return α, w

We now review the Nelder-Mead method [Gao and Han, 2012]. Suppose we have objective function f where $f : R^n \to R$. A simplex Δ consists of $n+1$ points (x_1, ..., $x_{n+1} \in R^n$) where $f(x_1) \leq f(x_2) \leq ... \leq f(x_{n+1})$. One iteration of the method is summarized below.

1. Sort x_1, ..., x_{n+1} so that $f(x_1) \leq f(x_2) \leq ... \leq f(x_{n+1})$.
2. Compute the reflection point x_r from $x_r = \bar{x} + \alpha_0(\bar{x} - x_{n+1})$. If $f(x_1) \leq f(x_r) < f(x_n)$, replace x_{n+1} with x_r.
3. If $f(x_r) < f(x_1)$, then compute the expansion point x_e by $x_e = \bar{x} + \beta_0(x_r - \bar{x})$. If $f(x_e) < f(x_r)$, replace x_{n+1} with x_e; otherwise replace x_{n+1} with x_r.
4. If $f(x_n) \leq f(x_r) < f(x_{n+1})$, compute the outside contraction point by $x_{oc} = \bar{x} + \gamma_0(x_r - \bar{x})$. If $f(x_{oc}) \leq f(x_r)$, replace x_{n+1} with x_{oc}.
5. If $f(x_r) \geq f(x_{n+1})$, compute the inside contraction point by $x_{ic} = \bar{x} - \gamma_0(x_r - \bar{x})$. If $f(x_{ic}) < f(x_{n+1})$, replace x_{n+1} with x_{ic}.
6. For $i = 2, ..., n+1$, update $x_i = x_1 + \delta_0(x_i - x_1)$

where $\bar{x}$ is the centroid of $x_1, ..., x_n$ and $(\alpha_0, \beta_0, \gamma_0, \delta_0)$ are tuning parameters where $\alpha_0 > 0$, $\beta_0 > 1$, $0 < \gamma_0 < 1$ and $0 < \delta_0 < 1$. [Gao and Han, 2012] proposed a method with adaptive shrinkage parameters according to the problem dimension n, which outperforms the standard Nelder-Mead method for large-scale problems.

It is worth noting that our proposed algorithm may raise the computational issue for large-scale problems. We do not know how many iterations the algorithm needs to reach convergence. It is also possible that this proposed algorithm does not converge to a local minimum for some particular problems.

4 Simulation

In this section, we evaluate the performance of the proposed method on a simulated dataset. For the baseline method, we randomly sample the threshold within a range and compute F1 score and accuracy by the algorithm proposed in [Ratner et al., 2017] using fixed thresholds.

In this simulation, we generate our data from Gaussian mixture model.

$$\begin{aligned} X|Y = 1 &\sim N(\mu_1, \Sigma) \\ X|Y = -1 &\sim N(\mu_0, \Sigma) \end{aligned} \tag{7}$$

where $\mu_1 = [1, 1, 1]$, $\mu_0 = [-1, -1, -1]$ and $diag(\Sigma) = [1, 0.25, 0.04]$. The off-diagonal elements of Σ are set to zero. We simulate $n = 500$ samples for both positive and negative label. We can thus define three labeling functions based on each dimension of simulated data. For $i = 1, 2, 3$,

```
def LF_i(x, thre):
    if x[i] < -thre:
        return -1
    elif x[i] > thre:
        return 1
    else:
        return 0
```

In general, we may have some information for labeling functions such as the range of thresholds. For example, the *thre* is uniformly selected from the range $[0, 2]$ in our experiment and this is also the initial value for α in our proposed method. We show comparison of our proposed algorithm with the baseline method in table 1 and table 2. These two tables illustrate that our method achieves a higher accuracy and F1 score both in the case we don't use discriminative model and the case a linear discriminant analysis classifier is applied.

Table 1: Averaged simulation results from 50 trials (without using discriminative model)

	Accuracy	F1 score
our method	**0.9898**	**0.9898**
baseline method	0.9134	0.8955

Table 2: Averaged simulation results from 50 trials (with linear discriminant analysis)

	Accuracy	F1 score
our method	**1.0000**	**1.0000**
baseline method	0.9588	0.9489

5 Real-world Application

In this section, we apply our method to an image classification task. We select 540 images from Microsoft COCO dataset [Lin et al., 2014]. Two examples with positive label (a person riding the bike) and negative label are showed in Figure 2. We first balanced our data with half negative examples and half positive examples. We also separated our data into training and testing data (25%) in order to provide a reliable metric. As a first step, we define three weak labeling functions

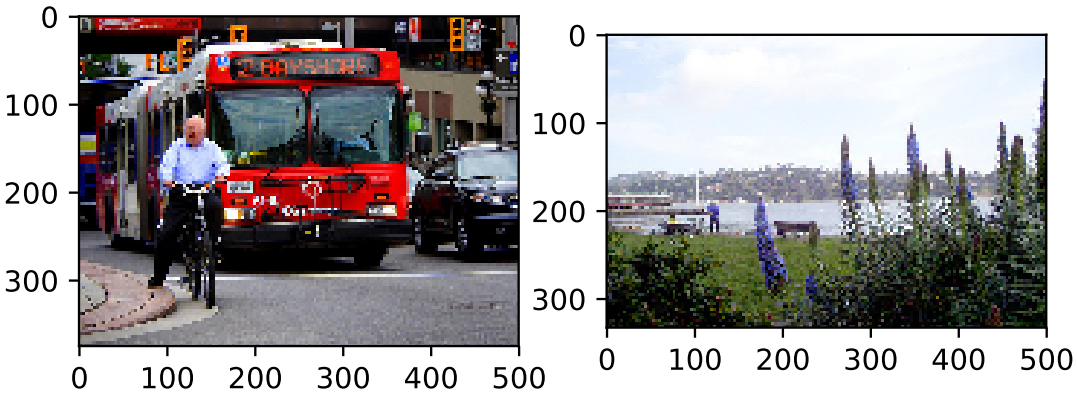

Figure 1: Two representative examples of images in our task. Left panel is a positive example while the right is a negative example.

as follows. For simplicity, we only consider one labeling function LF_3 with threshold variable. In this labeling function, we examined different distance between humans and bikes. The first labeling

function uses additional cars as an indicator function while the second explores whether the image contains the road. It is obvious that these three labeling functions are correlated thus we employed structure learning proposed in [Bach et al., 2017] during our iterative algorithm.

```
def LF_1(image):
    if image contains cars:
        return 1
    else:
        return -1
def LF_2(image):
    if image contains road:
        return 1
    else:
        return -1
def LF_3(image, thre):
    if image contains human:
        if image contains bikes:
            if bike_human_distance <= thre:
                return 1
            else:
                return -1
        else:
            return 0
    else:
        return 0
```

We repeated our experiment 10 times from different initial values that were uniformly sampled from the interval $[0, 50]$. After obtaining weak labels from our method and baseline method (threshold randomly selected), we transformed our image data into feature vectors by using convolutional neural network (ResNet-18) and then pass these features into a logistic regression classifier. Fine tuning the classic deep neural networks [Simonyan and Zisserman, 2014, He et al., 2016] for classification tasks in computer vision area can potentially improve final results. However, due the time limit, we chose to directly use these networks as a feature extractor. Table 3 shows the comparison

Table 3: Averaged simulation results from 10 trials (with logistic regression classifier)

	Accuracy	F1 score
our method	**0.60**	**0.62**
baseline method	0.58	0.58

of our method with the baseline method. From the table, we can observe our method achieves a better F1 score and accuracy than the baseline method.

Figure 2 shows an example of how negative log-likelihood function's value changes when the threshold in LF_3 varies. The Nelder-Mead method aims to find the threshold that minimizes this function. However, it is obvious that this function is non-convex. It is very likely for this method to stuck in local minimum if the initial simplex is not well chosen.

It is worth mentioning our iterative algorithm usually converges in several iterations (less than 5) for this application. Thus it only takes a few minutes to finish one independent trial of our experiment. It is possible to add more weak labeling functions to further improve classification

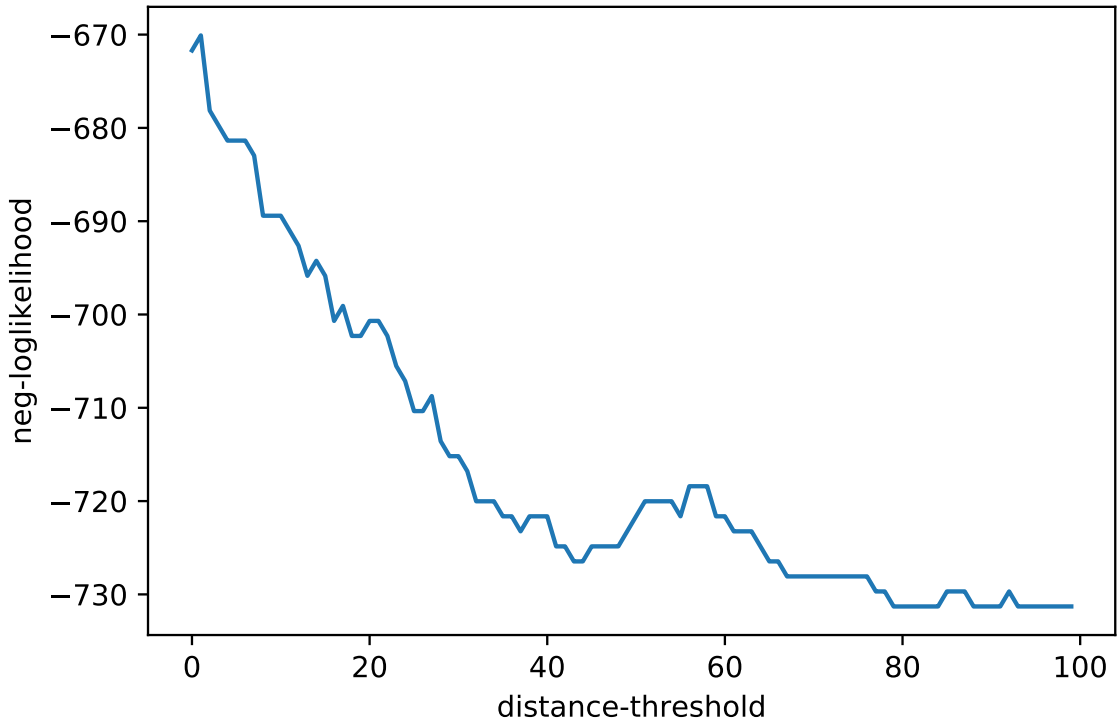

Figure 2: The figure shows one example of functions that Nelder-Mead method needs to optimize.

accuracy. We also calculated the accuracy and f1 score using a fully supervised method and achieved 0.64 and 0.63 respectively. The f1 score is very close to what we obtained from our method.

6 Related Work

In general, there are three types of weak supervision [Zhou, 2017]. The first is incomplete supervision which means a small subset of training data is given with labels while the other data remain unlabeled. Two major techniques in this setting are active learning [Prince, 2004] and semi-supervised learning [Chapelle et al., 2009]. The second type is inexact supervision where we only have coarse-grained labels. One such example is multiple instance learning [Dietterich et al., 1997] where the training dataset consists of labeled bags, each of which is a group of unlabeled data points. The third type is inaccurate supervision in which the given labels are noisy and not always ground truth. Related studies include distant supervision [Craven et al., 1999, Mintz et al., 2009, Takamatsu et al., 2012], learning from weak teachers or crowdsourcing [Brabham, 2008, Sheng et al., 2008, Dekel and Shamir, 2009, Urner et al., 2012], encoding weak supervision in the form of labeling functions [Ratner et al., 2016, Bach et al., 2017, Ratner et al., 2017].

After introducing the threshold as variables in our generative model, the objective function does not vary smoothly when we perturb threshold variables. It is thus reasonable for us to explore derivative-free optimization methods in the literature. The development of derivative-free algorithms dates back to the simplex-based algorithms [Spendley et al., 1962, Nelder and Mead, 1965]. The weakness of this method is that it is not very efficient for a problem with a large number of variables. Torczon introduced generalized pattern search methods (GPS) for

unconstrained optimization as another local search method. Some global search algorithms have also been proposed in the literature including simulated annealing [Metropolis et al., 1953], genetic algorithms [Holland, 1992], particle swarm algorithms [Eberhart and Kennedy, 1995] and multilevel coordinate search [Huyer and Neumaier, 1999].

There is also some development on Nelder-Mead method for high dimensional optimization problem [Gao and Han, 2012]. In [Gao and Han, 2012], they proposed a new implementation of the Nelder-Mead method (adaptive Nelder-Mead simplex (ANMS) method) in which the parameters for expansion, contraction, and shrink depend on the dimension of the optimization problem. Their proposed method aims to avoid the rapid reduction of simplex diameter during the iterative algorithm.

7 Discussion and Future works

There are several limitations in this work. One main limitation of our proposed algorithm is that it may not be suitable for high dimensional problems. We have observed one example of negative log-likelihood functions in our experiment section. The function is non-convex even if there is only one threshold value. We could imagine the problem would become more complicated when the dimension of unknown thresholds is large. One possible solution is that we ask domain experts to give a relatively reasonable initial value for our algorithm. Another possible remedy is to use a semi-supervised learning method to find reliable initial values. One simple way to do this is to solve a one-dimensional classification problem (if we assume only one threshold for each labeling function) for each labeling function involved with the unknown threshold. We can also apply transductive support vector machine method [Joachims, 1999].

Another limitation is that we treat thresholds and the remaining parameters separately. We have noticed that the indicator function is not differentiable thus it raises the challenge for us to optimize all the parameters together. Can we instead use a smoothed version of indicator function? This might be an interesting future work direction.

Furthermore, due to the time limit, we are unable to test and verify our proposed algorithm in more applications in this paper. It is not clear whether this method will suffer other unknown problems with more complex applications. It is also interesting to explore different derivative-free optimization methods such as particle swarm algorithm [Eberhart and Kennedy, 1995]. We can then gain insight into how different methods perform in various situations.

8 Conclusions

In this paper, we develop a method to automatically pick the threshold inside of labeling functions of Snorkel framework. Our method uses a data-driven method to choose a "good" threshold for labeling functions. This paper aims to reduce human efforts when users choose and define labeling functions. We have shown in both simulated data and a simple image classification task, our method outperforms the baseline method in terms of accuracy and F1 score.

References

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics, 2009.

Shingo Takamatsu, Issei Sato, and Hiroshi Nakagawa. Reducing wrong labels in distant supervision for relation extraction. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 721–729. Association for Computational Linguistics, 2012.

Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 614–622. ACM, 2008.

Ruth Urner, Shai Ben David, and Ohad Shamir. Learning from weak teachers. In *Artificial Intelligence and Statistics*, pages 1252–1260, 2012.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *Proceedings of the VLDB Endowment*, 11(3):269–282, 2017.

Paroma Varma, Bryan D He, Payal Bajaj, Nishith Khandwala, Imon Banerjee, Daniel Rubin, and Christopher Ré. Inferring generative model structure with static analysis. In *Advances in neural information processing systems*, pages 240–250, 2017.

Jorge Nocedal and Stephen J Wright. Numerical optimization 2nd, 2006.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Zhi-Hua Zhou. A brief introduction to weakly supervised learning. *National Science Review*, 5(1): 44–53, 2017.

Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In *Advances in neural information processing systems*, pages 3567–3575, 2016.

Stephen H Bach, Bryan He, Alexander Ratner, and Christopher Ré. Learning the structure of generative models without labeled data. *arXiv preprint arXiv:1703.00854*, 2017.

Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

John A Nelder and Roger Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

Fuchang Gao and Lixing Han. Implementing the nelder-mead simplex algorithm with adaptive parameters. *Computational Optimization and Applications*, 51(1):259–277, 2012.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Michael Prince. Does active learning work? a review of the research. *Journal of engineering education*, 93(3):223–231, 2004.

Olivier Chapelle, Bernhard Scholkopf, and Alexander Zien. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews]. *IEEE Transactions on Neural Networks*, 20(3):542–542, 2009.

Thomas G Dietterich, Richard H Lathrop, and Tomás Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial intelligence*, 89(1-2):31–71, 1997.

Mark Craven, Johan Kumlien, et al. Constructing biological knowledge bases by extracting information from text sources. In *ISMB*, volume 1999, pages 77–86, 1999.

Daren C Brabham. Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence*, 14(1):75–90, 2008.

Ofer Dekel and Ohad Shamir. Good learners for evil teachers. In *Proceedings of the 26th annual international conference on machine learning*, pages 233–240. ACM, 2009.

WGRFR Spendley, George R Hext, and Francis R Himsworth. Sequential application of simplex designs in optimisation and evolutionary operation. *Technometrics*, 4(4):441–461, 1962.

Virginia Torczon. On the convergence of pattern search algorithms. *SIAM Journal on optimization*, 7(1):1–25, 1997.

Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

John Henry Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

Russell Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on*, pages 39–43. IEEE, 1995.

Waltraud Huyer and Arnold Neumaier. Global optimization by multilevel coordinate search. *Journal of Global Optimization*, 14(4):331–355, 1999.

Thorsten Joachims. Transductive inference for text classification using support vector machines. In *ICML*, volume 99, pages 200–209, 1999.