Active Deep Learning To Tune Down the Noise in Labels

Karan Samel
Astound
Menlo Park, CA
karan@astound.ai

Xu Miao
Astound
Menlo Park, CA
xu@astound.ai

ABSTRACT

The great success of *supervised learning* has initiated a paradigm shift from building a deterministic software system to a probabilistic artificial intelligent system throughout the industry. The historical records in *Enterprise* domains can potentially bootstrap the traditional business into the modern data-driven approach almost everywhere. The introduction of the *Deep Neural Networks* (DNNs) significantly reduces the efforts of feature engineering so that *supervised learning* becomes even more automated. The last bottleneck is to ensure the data quality, particularly the label quality, because the performance of *supervised learning* is bounded by the errors present in labels. In this paper, we present a new *Active Deep Denoising* (ADD) approach that first builds a DNN noise model, and then adopts an *active learning* algorithm to identify the optimal denoising function. We prove that under the low noise condition, we only need to query the oracle with $\log n$ examples where n is the total number in the data. We apply ADD on one *Enterprise* application and show that it can effectively reduce $\frac{1}{3}$ of the prediction error with only 0.1% of examples verified by the oracle.

KEYWORDS

Active learning, De-noise, Deep Neural Networks, Classification

ACM Reference Format:
Karan Samel and Xu Miao. 2018. Active Deep Learning To Tune Down the Noise in Labels. In *Proceedings of ACM KDD conference (KDD'18).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

During the past decades, *supervised learning* has achieved a great success across many fields, e.g., natural language processing, computer vision, and information retrieval, with the emergence of *Big data* and *Cloud computing*. For any given task, we collect data, label data, build a model, evaluate the model, revise and repeat. This loop, as shown in Figure 1, has become the new programming paradigm to solve many real world problems, and shifted the major focus from building a high quality software system to building a high quality dataset, e.g., *ImageNets* [9] for object recognition and *SQuAD* [22] for machine comprehension. However, if the collected data contains noisy and conflicting labels, the performance of the *supervised learning* is upper bounded. For example, as a popular crowd sourcing choice, data are labeled through *Mechanic Turk* (MT) where annotators do not necessarily have domain specific knowledge, hence bringing human errors into the dataset. Even with the majority vote mechanism, the quality of the labels is mostly not guaranteed. This is more severe in *Enterprise* domains where labels come from different persons at different locations and times who follow different rules. Recent research [18] studied the effect of low-quality training data on clinical reports, and demonstrated that the difficulty of acquiring high-quality data actually bottlenecks the wide adoption of the data-driven approach in the health domain. Indeed, how can we obtain super-human accuracy from noisy inaccurate data? In this paper, we address this challenge from the *active learning* perspective.

Active learning [3, 11, 15, 19, 25, 30, 34] was proposed to effectively utilize unlabeled data with a costly oracle. When the prior distribution over the hypothesis space is known, algorithms like *Query by Commitee* [11], i.e., pick the most uncertain example to ask, can effectively query the oracle with a guaranteed generalization bound. When the prior distribution is unknown, this approach can bring significant biases to the samples, and ends with statistically inconsistent models. Research [5, 6] shows that with certain search heuristics, e.g., exploitation v.s. exploration, we can avoid this bias and obtain statistical consistency. However, the label complexity remains linear with respect to *supervised learning* because we might still need to query significant amount of examples to obtain the desired accuracy. Theoretical studies [2, 7, 13] have shown that algorithms like *Agnostic Active* (A^2) can be exponentially effective under the low noise condition, but are bound by a large VC-dimension factor. It is practically impossible to combine A^2 with a family of models like *Deep Neural Networks* in the *unsupervised learning* domain as we have to search through a huge hypothesis space to obtain a confident estimation.

Fortunately, in our setting, all data are labeled, although with certain amount of errors. These noisy labels allow us to build a model to address the noise systematically which we use in restricting the hypothesis space. In this paper, we present a new *Deep Neural Network* based *Active Learning* algorithm to correct noisy labels. We first build a DNN noise model to form a de-noising function family. Then, we adopt the A^2 algorithm to learn the optimal de-noising function from the *oracle*. We prove an exponential reduction of the label complexity under low noise condition due to the fact that the constructed de-noising function has the VC-dimension of 1. In addition, an empirical comparison among different choices of noise models demonstrates that the proposed algorithm can effectively tune down the noise with less number of queries to the expensive *oracle*.

KDD'18, August 2018, London, United Kindom

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

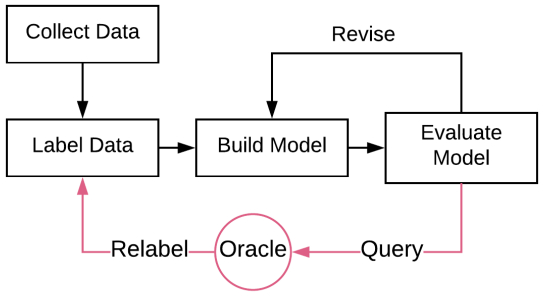

Figure 1: Data quality is the key to a data driven paradigm

Caller: Person_1
Agent: Person_2
Description: Unable to duplicate email distribution lists
Category: Email

Caller: Person_3
Agent: Person_4
Description: Unable to send or receive email because it has spam or possible virus
Category: Resource Management

Figure 2: Examples for IT Incident Categorization. Although both incidents are about email, the second is categorized differently. It might be mis-labeled due to human agent's misunderstanding.

2 PROBLEM STATEMENT AND RELATED WORK

Noisy labels are ubiquitous for all real world tasks. We study the problem on *Enterprise* domains which are traditionally built as a system of records. These historical data facilitate bootstrapping the process from a deterministic *software* system to a data-driven *AI* system. For example, *IT service desk* provides assistance to employees to troubleshoot their IT related issues. The system creates an *incident* to engage with a user, and human agents classify them into different categories and route them to different IT teams for resolution. See examples in Figure 2. The accuracy of the classification is very crucial for incidents being resolved with low latency because the mis-routing can cause significant delay within the system. The machine learning task is to learn to categorize from the data recorded by human agents. The challenge is that these historical data contain significant errors in labels that bottlenecks the performance of the supervised learning approach. First of all, human agents make mis-takes due to lack of training or operational errors. Secondly, different agent might have different understanding of the system, and they conflict with each other in many cases. At last, the system routing has been changed over the time which renders the problem *non-stationary*. All of these factors result in noisy labels. Experts exist in the system who can help clean up the labels, but they are very expensive. Having them go through the entire historical records is not economic. The cost is even higher if the AI service is provided from a third party. In fact, effectively utilizing expert oracles to bootstrap an *AI* system has become a common challenge for *Enterprise* AI applications.

Actively de-noising labeled data is not a new topic [10, 12, 14, 20, 21, 24, 26, 28]. The problem has been studied under the *inductive logic programming* framework in the early days of Machine Learning. For example, a polynomial time algorithm was proposed in [1] that identifies concepts in the form of k-CNF formulas if the labeling error rate is less than half. Research [8] introduced an *EM* algorithm to simultaneously estimate annotator biases and latent label classes if multiple annotators labeled the same example. Other work [26] applied a statistical classifier to predict the true label for each example given multiple annotations for each example. These early work focuses on the case where multiple annotators are available for each data point, which is quite restricted in the nowadays. Especially in many enterprise domains, it is almost impossible to collect multiple annotations for each datapoint. *CorrActive* [20] proposed an algorithm to estimate the confidence of mis-labeling according to a probability model and iteratively query the oracle with the most likely mis-labeled data. This algorithm does not require multiple annotators for every data point and demonstrates the effectiveness on one real application. However, it did not provide any theoretical or empirical study about how to best estimate the confidence of mis-labeling.

Support Vector Machines based approaches have been widely studied too [10, 24, 28, 30, 32, 36]. The main observation is that mis-labeled data are likely to be support vectors, and margin is directly related to the uncertainty of the data. In spite of the success in the past, we do not base our work on SVMs for several reasons. First of all, recent trend of *Deep Neural Networks* has significantly reduced the feature engineering efforts during modeling, and become the de facto approach for dealing with text, audio and image data. Secondly, research study in [4, 7] has shown that uncertainty driven approach is biased if significant noise present. The algorithm spends a lot of time on querying the same uncertain area repeatedly without an escaping mechanism. At last, SVMs are good to model the noise from the input signals, but not to utilize the output label information.

A clustering based approach was proposed in [6] to address the inconsistency issue for the uncertainty driven active learning approach. The example to query is picked according to maximum information gain, i.e., how much information we can gain with respect to the distribution over both data and hypothesis. After the true label is acquired, the neighborhood's label is flipped collectively, e.g., by a majority vote mechanism. In this paper, we follow the similar approach, estimate the neighborhood through a DNN and propose the examples from highly likely mis-labeled clusters. DNN provides a low-dimensional dense vector embedding to represent the semantic meaning of the original data [16, 27, 29, 35]. This representation

models well on the joint manifold of both inputs and output labels, and creates more accurate estimation of mis-labeling clusters.

We study the approach from both Bayesian and Non-Bayesian perspectives. *Bayesian Deep Learning* [31] combines *Probabilistic Graphical Models* [17, 33] and DNNs to model noises and uncertainty within the Deep Learning framework. Algorithms like *Evidence Lower Bound Optimization* (ELBO) [23] allow fast approximate optimization to avoid intractability of the normalization function in PGMs. *Non-Bayesian* perspective further relaxes the conditional independency among data points, i.e., clusters share similar noise distributions. The uncertainty estimation is more robust with a local neighborhood smoothing than just a point estimation. Our empirical study also shows that combining two perspectives together provides the best results.

3 ACTIVE DEEP DE-NOISING (ADD)

Deep neural networks have demonstrated a great advantage representing unstructured data, e.g., texts and images. Unlike the unsupervised dimensionality reduction approach that does not take the output space into consideration, DNNs learn a metric space towards the maximal discrimination even with present of significant noises. This potentially helps the active learning process identify similar mis-labeled patterns and correct them collectively. As shown in Figure 3, we model the $P(Y = y|\mathbf{x}; \Theta_2) = \text{softmax}(y; \mathbf{x}, \Theta_2)$.

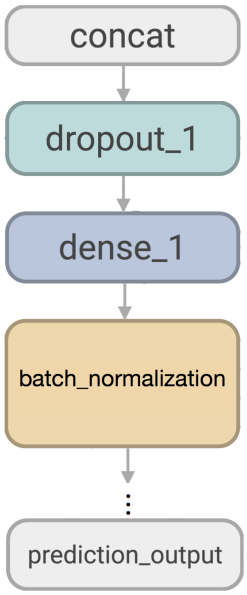

Figure 3: Deep Neural Network structure for outputting $softmax(y; \mathbf{x})$ for Θ_1 and Θ_2. Our concatenated description feature vectors are fed through multiple iterations of dropout, fully connected (dense), and normalization layers. The prediction output is where the $softmax$ of the last normalization layer is computed to give our y.

A probabilistic graphical model shown in Figure 4 describes the noise model where X is the input variables, Y is the hidden true label and Z is the given noisy label.

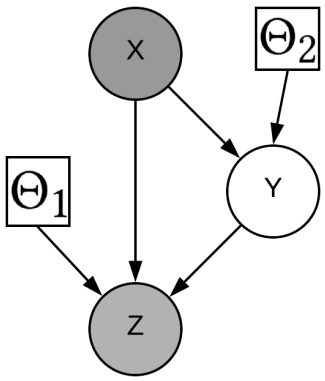

Figure 4: Probabilistic Graphical Model For De-noising. X is the input data, i.e., caller, agent and description. Y is the hidden true label. Z is the given noisy label. Θ_1 and Θ_2 are model parameters

3.1 Modeling noises

A high quality noise model provides the foundation to engage an active de-noising process. We keep track of a model that predicts the original labels, given the inputs and our estimated true label, in order to verify that our label changes result in systematic improvement and understanding of the underlying noise. From the Bayesian perspective, we maximize the likelihood of the two models by marginalizing the hidden true label Y. In Equation 2, we adopt the *Evidence Lower Bound* optimization approach to approximate the marginalization.

$$
\begin{aligned}
& \text{maximize}_{\Theta_1, \Theta_2} \\
& \mathbb{E}[\log \sum_Y P(Z, Y|X; \Theta_1, \Theta_2)] \qquad (1) \\
= \; & \mathbb{E}[\log \sum_Y P(Z|Y, X; \Theta_1) P(Y|X; \Theta_2)] \\
\geq \; & \mathbb{E}_{X,Z}\left[\mathbb{E}_{Y|X,Z}[\log P(Z|Y, X; \Theta_1) P(Y|X; \Theta_2)] + \mathbb{H}_{Y|X,Z}\right] \\
\geq \; & \frac{1}{|S|} \sum_{\mathbf{x}, z \in S} \sum_y P(y|\mathbf{x}, z)(\log P(z|y, \mathbf{x}; \Theta_1) + \log P(y|\mathbf{x}; \Theta_2)) \\
& - \frac{1}{|S|} \sum_{\mathbf{x}, z \in S} \sum_y P(y|\mathbf{x}, z) \log P(y|\mathbf{x}, z) - \frac{\gamma}{2}(\|\Theta_1\| + \|\Theta_2\|)
\end{aligned}
$$

Both $P(z|y, \mathbf{x}; \Theta_1)$ and $P(y|\mathbf{x}; \Theta_2)$ are DNN models. $P(y|\mathbf{x}, z)$ is the posterior estimation of the true label given the noisy label and input data. The relaxed objective is to minimize the expected errors for both predicting noisy labels and true labels with regularizations over the posterior and the parameters. Posterior estimation by the Bayesian rule is easier to compute, but also prone to bias when data is sparse. The non-Bayesian perspective relax the posterior estimation to a parametrized model $P(y|\mathbf{x}, z; \Theta_3)$, e.g., the kNN smoothing.

ELBO allows the optimization of Equation 2 directly if only Bayesian approach is taken. For non-Bayesian smoothing, jointly optimizing three models over a large input and label space becomes infeasible. Instead we propose to optimize each model individually

as a coordinate descent approach, until the overall loss converged. The algorithm is presented in Algorithm 1

Algorithm 1: Modeling Noises

input : $S = \{\mathbf{x}, z\}$
output : $\Theta_1, \Theta_2, \Theta_3$

begin
  Let $y = z$, $P(y|\mathbf{x}, z; \Theta_3) = 1$ if $y = z$; 0 otherwise;
  repeat
    $\Theta_2 = \arg\max_{\Theta_2} \sum_{\mathbf{x},z} \sum_y P(y|\mathbf{x}, z; \Theta_3) \log P(y|\mathbf{x}; \Theta_2) - \frac{\gamma}{2}\|\Theta_2\|$;
    $P(y|\mathbf{x}, z; \Theta_3) = \text{Posterior}(S, \Theta_1, \Theta_2)$;
    $\Theta_1 = \arg\max_{\Theta_1} \sum_{\mathbf{x},z} \sum_y P(y|\mathbf{x}, z; \Theta_3) \log P(z|\mathbf{x}, y; \Theta_1) - \frac{\gamma}{2}\|\Theta_1\|$;
  until *Equation 2 converges*;
end

Θ_1, and Θ_2 models are neural networks shown in Figure 3. The estimation of the posterior determines potential label changes as the de-noising process. If the true labels conform to the input attributes of similar features, $P(y = z|\mathbf{x}, y; \Theta_3) \approx 1$. Otherwise, the probability decreases. Determining how to estimate these probabilities is crucial, thus multiple methods for estimating Θ_3 were proposed and compared.

The first method is a bayesian posterior calculated by simply inferring the hidden label from Θ_1 and Θ_2. This gives us a smooth probability distribution over a single data point $\{\mathbf{x_i}, z_i\}$, but does not take into account similar examples with varying z.

Algorithm 2: Posterior by simple Bayesian rule

input : $S = \{\mathbf{x}, z\}, \Theta_1, \Theta_2$
output : $P(y|\mathbf{x}, z; \Theta_3)$

begin
  $P(y|\mathbf{x}, z; \Theta_3) = \frac{P(z|\mathbf{x}, y; \Theta_1) P(y|\mathbf{x}; \Theta_2)}{\sum_y P(z|\mathbf{x}, y; \Theta_1) P(y:\mathbf{x}; \Theta_2)}$
end

Bayesian point estimation tends to be biased in regions with sparse data. Taking a neighborhood of $\mathbf{x}$, we can take into account variations to smooth out the posterior estimation. The most common surrounding label for similar inputs also indicates the true label, regardless of what given label was assigned to that point. This majority vote mechanism is basically the k-nearest-neighbor (kNN) classifier where the neighborhood is based on the manifold learned from noisy data by the DNNs.

The neighborhood is taken from a dense layer from our Θ_2 model during inference. We cluster the data based on the last dense layer as they represent the high level features in a low dimensional space. The low dimensional manifold learned from the noisy data provides a higher quality clustering than any unsupervised approach commonly used in other active learning algorithms. Note that this is one of the key reasons that ADD succeeds.

Algorithm 3: Posterior by neighborhood majority

input : $S = \{\mathbf{x}, z\}, \Theta_2$
output : $P(y|\mathbf{x}, z; \Theta_3)$

begin
  foreach $\mathbf{x_i}, z_i \in \{\mathbf{x}, z\}$ **do**
    $\mathbf{x_s} = \text{kNN}(\text{HiddenLayers[-1]}(\mathbf{x_i}))$;
    $\mathbf{w} = \sum_{\mathbf{x_{sj}} \in \mathbf{x_s}} \text{softmax}(\mathbf{x_{sj}}; \Theta_2) \circ \mathbb{I}_{x_{sjk} = \max(\text{softmax}(\mathbf{x_{sj}}; \Theta_2)); x_{sjk} \in \mathbf{x_{sj}}}$;
    $\hat{y} = \arg\max(w)$;
    $P(y_i|\mathbf{x_i}, z_i; \Theta_3) = \mathbb{I}_{z=\hat{y}}$;
  end
end

While the majority-vote algorithm yields the consensus of the neighborhood, the posterior distribution is effectively a one hot encoded, which does not provide us a range of values to use when computing Algorithm 1. We explore a combined approach that use the Bayesian estimation to smooth out the neighborhood.

Algorithm 4: Posterior by neighborhood smoothing

input : $S = \{\mathbf{x}, z\}, \Theta_1, \Theta_2$
output : $P(y|\mathbf{x}, z; \Theta_3)$

begin
  foreach $\mathbf{x_i}, y_i \in \{\mathbf{x}, z\}$ **do**
    $\mathbf{x_s} = \text{kNN}(\text{HiddenLayers[-1]}(\mathbf{x_i}))$;
    $\mathbf{w} = \sum_{\mathbf{x_{sj}} \in \mathbf{x_s}} \text{softmax}(\mathbf{x_{sj}}; \Theta_2) \circ \mathbb{I}_{x_{sjk} = \max(\text{softmax}(\mathbf{x_{sj}}; \Theta_2)); x_{sjk} \in \mathbf{x_{sj}}}$;
    $P(y_i|\mathbf{x}, z; \Theta_3) = \mathbf{w}$;
  end
end

3.2 Actively de-noising labels

After the noise model is built, we select samples that our models suggest and verify these changes with the oracle. The verification process is expensive, and we are allowed with K examples to get the high quality verifications. If K has be linear on the sample size $|S|$, basically the oracle is required to relabel most of the data to achieve the optimal accuracy. If $K = O(\log(|S|)$, the active de-noising save the expensive cost exponentially. We show that the exponential speedup of the label complexity is feasible under the low noise condition as we construct a threshold based de-noising function that has the VC-dimension of 1.

Since we were looking for examples which could generalize to other examples for relabeling, we looked at the clustering methods that are used to flip the labels during modeling in Section 2. When constructing Θ_3 to determine the true label, the estimations were made for every single point using a learned hidden representation. This representation is used to create the vectors that are clustered to determine similarly structure examples. This led our true label model Θ_2 to achieve a much higher accuracy than any base model Θ_1. In selection, we have the knowledge of Θ_2, but need to verify

those changes with an oracle. We use the metric in Equation 2 to indicate the score of mis-labeling, and select samples accordingly.

$$f(\mathbf{x}, z) = \max_{y \neq z} \log P(y|\mathbf{x}, z; \Theta_3) - \log P(z|\mathbf{x}; \Theta_1, \Theta_2) \quad (2)$$

Proposition 3.1 proves that the true entropy is upper bounded by the entropy based on the noisy labels and lower bounded by the entropy after de-noising the labels. The active learning process is to effectively query the oracle to approximate the true entropy with less samples. Intuitively, we just need to sort the data points according to the scores, and verify the point from the oracle that all data points above are mis-labels. We adopt the *Agnostic Active Learning* approach to estimate the target threshold.

PROPOSITION 3.1. *Let ξ represent the true noise level in the labels. We have*

$$\xi = -\frac{1}{S} \sum_{\mathbf{x}, z, \bar{y} \in S} P(\bar{y}|\mathbf{x}, z) \log P(\bar{y}|\mathbf{x}, z) \quad (3)$$

$$\xi \leq -\frac{1}{S} \sum_{\mathbf{x}, z \in S} \sum_{y} P(y|\mathbf{x}, z; \Theta_3) \log P(z|\mathbf{x}, y; \Theta_1) P(y|\mathbf{x}; \Theta_2) \quad (4)$$

$$\xi \geq -\frac{1}{S} \sum_{\mathbf{x}, z \in S} \sum_{y} P(y|\mathbf{x}, z; \Theta_3) \log P(y|\mathbf{x}, z; \Theta_3) \quad (5)$$

PROOF. (Sketch) For inequality 4, it can be divided into two cases, $y = z$ and $y \neq z$, in both cases, the true probability is greater or equal to the noisy label. For inequality 5, Θ_3 is the optimal parameter that optimize Equation 2, hence can only be greater than the true parameter. □

Therefore, we design the upper bound function and lower bound function to be Equation 6 and 7. These two functions guide the active learning process to find the optimal threshold h for the mis-labeling score $f(\mathbf{x}, z)$. We denote $\Delta = \{\mathbf{x}, z, \bar{y}\}$ to be the set of data with feedbacks from the oracle, and $\bar{y}$ is the expert true label. We denote $\hat{y}(\mathbf{x}, z; h)$ to be the de-noised label which is the maximal $y \neq z$ if $f(\mathbf{x}, z) > h$; z, otherwise;

$$U(h, \Delta) = -\frac{1}{|\Delta|} \sum_{\mathbf{x}, z, \bar{y} \in \Delta} P(\bar{y}|\mathbf{x}, z; \Theta_3) \log P(z|\mathbf{x}, \hat{y}; \Theta_1) P(\hat{y}|\mathbf{x}; \Theta_2) \quad (6)$$

$$L(h, \Delta) = -\frac{1}{|\Delta|} \sum_{\mathbf{x}, z, \bar{y} \in \Delta} P(\bar{y}|\mathbf{x}, z; \Theta_3) \log P(\hat{y}|\mathbf{x}, z; \Theta_3) \quad (7)$$

The active de-noising algorithm is defined in Algorithm 5. Note that the disagreement function only needs to check the disagreements on the boundary points, i.e., $H = [h_1, h_2)$. Corollary 3.2, as a straight application of A^2 (Agnostic Active) algorithm's result, proves that under the low noise condition the label complexity is on the log-scale, otherwise it is on quadratic.

COROLLARY 3.2. *If $\xi \leq \frac{\epsilon}{16}$, with a high probability $1 - \eta$, Algorithm 5 makes $O(\ln(\frac{1}{\epsilon} + \frac{1}{\eta}))$ calls to the oracle. Otherwise, it makes $O(\frac{\xi^2 \ln \frac{1}{\eta}}{\epsilon^2})$ calls.*

Algorithm 5: Active De-noising

input : $S = \{\mathbf{x}, z\}$, $\Theta_1, \Theta_2, \Theta_3$, Oracle O and small error ϵ
output : $\hat{h}$

```
begin
    Let H = (−∞, +∞), Δ = {};
    repeat
        Sample 2Δ + 1 from S where ŷ(x, z; h) disagrees on
          different h ∈ H;
        Query the oracle O for ȳ;
        Add these samples {x, z, ȳ} to Δ;
        ĥ = arg min_{h∈H} U(h, Δ);
        H = {h ∈ H; L(h, Δ) ≤ U(ĥ, Δ)} = [h1, h2);
    until |ŷ(x, z; h1) ≠ ŷ(x, z; h2)| decrease less than half or
      U(ĥ, Δ) ≤ ε;
end
```

4 EXPERIMENTS AND ANALYSIS

For our experiments we use one of our incident categorization datasets with 150k incidents available for training, validation, and testing. In total the dataset contains 403 labels that needed to be classified. The distribution of these category labels are highly skewed, which has to be taken into account during model training (figure 5).

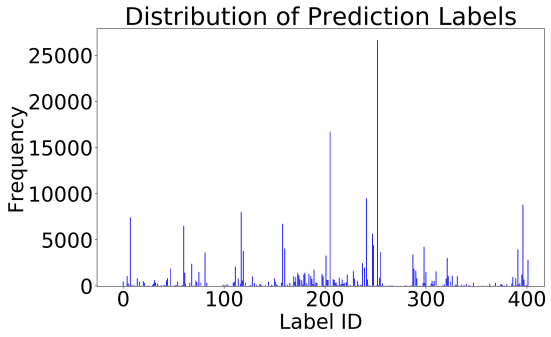

Figure 5: The label distribution used for prediction

We focus on modeling the accuracy gains given a similar DNN model architecture, over the accuracies of the DNN models themselves. The DNN structure, as described in figure 3, contains 3 sets of dropout, dense, and normalization layers before the final softmax prediction layer. Even though improving individual model accuracies was not our goal, different model architecture yields comparable performance.

4.1 De-noising comparisons

We first demonstrate the de-noising accuracy gains by comparing the Bayesian and neighborhood Θ_3 modeling methods as described in Algorithm [2,3,4]. We also tested two different approaches for embedding our feature vectors to input into our DNN. We tried both an average word vector embedding (figure 6), and a learned recurrent word vector representation (figure 7). Our convergence criterion was estimated as follows:

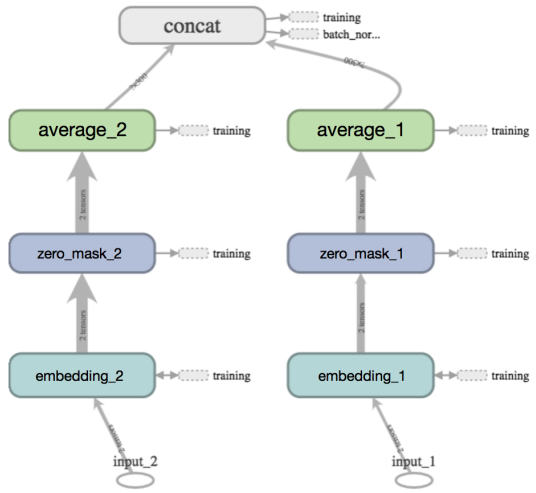

Figure 6: Average embedding representation. The inputs descriptions are taken in and embedded as vectors. Then the non zero vectors are extracted, and an average vector is computed before the features are concatenated.

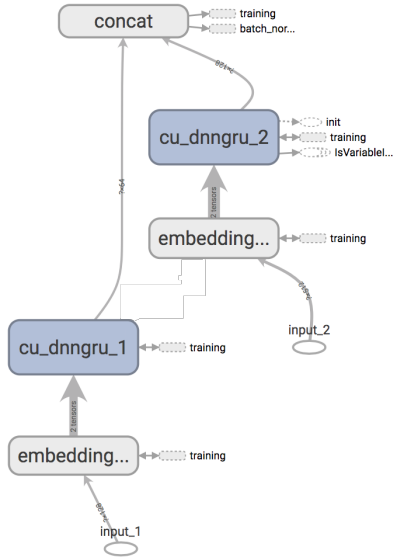

Figure 7: An embedding using GRU layers. Like the average embedding, the inputs are taken and converted into word vectors. These word vectors are fed into the GRU layers which output temporal representations of words embeddings provided by the inputs.

$$|(\Theta_1 \text{ accuracy}+\Theta_2 \text{ accuracy})_{t-1}-(\Theta_1 \text{ accuracy}+\Theta_2 \text{ accuracy})_t| \leq \delta$$

Here t is the iteration number for the repeat loop described in algorithm 1. In our tests we set the $\delta = 0.01$ for convergence test. Once the models are computed, we observe a batch of annotations where $|\Delta| = 100$ using the de-noising methodology described in algorithm 5. Those annotations are applied as $\hat{y}(\mathbf{x}, z; h)$ within our H thresholds. Our final model, $\Theta_{denoised}$ was then trained. We repeated this for every combination of average/GRU embedding and Θ_3 modeling methodology.

For each run we keep track of a couple of metrics. Our key metric is the $\Theta_{denoised} - \Theta_{base}$ to view our final impact of our verified label changes on our newly trained model over our previous best. We calculate the entropy over the predictions for all the data and keep the mean entropy reduction over all mean class entropies. This is the entropy reduction calculated from Θ_{base} to $\Theta_{denoised}$. This metric shows us if our modifications to the labels increased our overall confidence to predict each class, due to the decrease of noisy labels. For a sanity check we kept track of the percentage of labels flipped while the process is carried.

We analyze the effect of accuracy increase given the number of annotations provided to the oracle. We compare this to a baseline test of randomly sampling examples and making the label changes to the data directly (Figure 8). This clearly demonstrates the effectiveness of the algorithm while a random sampling approach represents the label complexity for the traditional *supervised learning*.

4.2 Denoising results

From the results in table 1, the average embedding using the smooth kNN approach gave us the best results in our data. Intuitively, this makes sense, as with the smooth estimation for y it had an edge when generalizing for our test data. We were expecting the GRU performance to be on par or exceeding that of the average embedding, but the accuracy improvements in test were about half. Looking at our data, this result is not surprising, as the descriptions were often largely segmented and discontinuous pieces of text.

The entropy reduction scores gave us some interesting results. The entropy drop for our best accuracy improvement, average embedding with smooth kNN, was less than that of the GRU kNN approach. We proceeded to explore the loss reductions per class basis, looking at the most frequent classes. We noticed that for the top 15 recurring classes (Table 2), the average loss reduction flips. Here we see an average loss reduction of 0.07 versus 0.04 for the average and the GRU methods respectively. With the only difference being the embedding methodology we inferred that the patterns learned by the GRU generalized over many classes. Observing the description of incidents shows us that they are structured very similarly for different classes, for example: "I am not able to login to x.", where x can be any concept ranging from payroll, email, environments, etc. We suspect that our GRU is able to capture this general structure, but we could not achieve a robust enough (without overfitting) model to distinguish the nuances with "x", each of which may belong to different classes. In comparison, the averaging embedding model is able to better focus on these discrepancies.

The label change percentage results fell within the range of what we expected. Previously, when manually combing over our data, we observed that about 10-20% were incorrectly labeled. In early experiment iterations we saw between 30-60% label changes. In these earlier results we were generating large kNN clusters over the denoised incidents and imposed a hard relabeling rule. This effect reflected the distribution of our data presented in figure 5, as too many incidents were mapped to these large categories. The active

Table 1: Modeling Accuracies and statistics across different combinations

Embedding	Method	Θ_1	Θ_2	Θ_{base}	$\Theta_{denoised}$	$\Theta_{denoised} - \Theta_{base}$	Entropy Reduction	% Labels Changed
Average Vector	Bayesian	0.7557	0.9534	0.7572	0.7877	0.0305	0.0127	13.2531
	Neighbor-Majority	0.7599	0.9604	0.7572	0.8543	0.0971	0.0981	13.1244
	Neighbor-Smooth	0.7606	**0.9613**	**0.7562**	**0.8570**	**0.1008**	**0.1097**	13.1501
GRU	Bayesian	0.7244	0.8888	0.7288	0.7676	0.0388	-0.0169	8.9279
	Neighbor-Majority	0.7244	0.9066	0.7253	0.7795	0.0542	0.1093	9.0586
	Neighbor-Smooth	0.7238	0.8982	0.7283	0.7825	0.0542	0.1226	9.0174

Label ID	Average Embedding Reduction	GRU Reduction	Counts
252	0.006431	0.000886	22099
205	0.245329	0.110417	16684
241	0.207836	0.190456	8819
396	0.028042	0.013118	7362
117	0.024486	0.014939	6488
7	0.081698	0.013575	4434
158	0.013308	0.010448	4322
60	0.014110	-0.003562	3968
247	0.028504	0.003326	3717
248	0.015550	0.008940	3492
298	0.164879	0.110065	3167
160	0.018674	0.005872	3126
391	0.105027	0.078930	3065
119	0.032742	0.018033	2443
255	0.054637	0.026784	2145
Average	0.069416	0.040148	-

Table 2: Assessing the reduction of mean entropy per label after generating $\Theta_{denoised}$ versus Θ_{base}. This compares how the average embedding and GRU representations compared when it came to reducing model loss in the most frequent categories.

de-noising, with soft uncertainty relabeling led to more reliable label changes on a class by class basis. In addition, we added sample weight parameters when training our DNN to mitigate this bias.

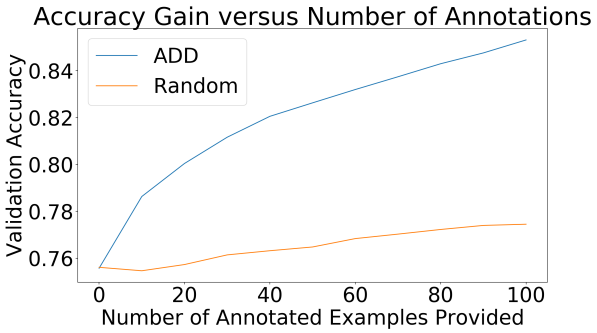

Figure 8: The accuracy v.s. the number of labels queried

Figure 8 shows how the accuracy improves during the first few iterations for our ADD method. This is partially due to the fact that confirmation from the oracle within our h_1 and h_2 bounds leads to more corrections, as these bounds are larger over noisier data. As the labels become de-noised, the corresponding H range shrinks. In comparison, randomly corrected samples increase the accuracy linearly over time. Most importantly, a DNN noise model effectively represents both input and output space in a smooth manifold so that the neighborhood clustering effectively groups similar errors, and allow the de-noising function to correct them collectively.

5 CONCLUSION AND FUTURE WORK

In this paper, we propose a DNN based label de-noising approach. We first model the noise with a Bayesian DNN. Furthermore, non-Bayesian neighborhood-based approach is combined with the Bayesian approach. Then, we query the oracle with an active de-noising algorithm selects examples that provide label verification for similarly structured data points. Once these mis-labels of similar examples are corrected, the DNN is trained on these more confident labels and repeat. We apply this approach on the IT incident categorization dataset. The experiment results demonstrate a significant accuracy improvements by just querying a small portion of mis-labeled data. We compare our results among Bayesian, non-Bayesian (neighborhood majority), and combined approach (neighborhood smooth). The combined approach demonstrates the best accuracy improvement. To our surprise, GRU based DNN produces an inferior result than a simple Average Embedding based DNN. As ADD can be generalized to other data, we look forward to comparing its noise detection performance on public datasets as well.

REFERENCES

[1] Dana Angluin and Philip Laird. 1988. Learning from noisy examples. *Machine Learning* 2, 4 (1988), 343–370.
[2] Maria-Florina Balcan, Alina Beygelzimer, and John Langford. 2009. Agnostic active learning. *J. Comput. System Sci.* 75, 1 (2009), 78–89.
[3] David Cohn, Les Atlas, and Richard Ladner. 1994. Improving generalization with active learning. *Machine learning* 15, 2 (1994), 201–221.
[4] Sanjoy Dasgupta. 2005. Analysis of a greedy active learning strategy. In *Advances in neural information processing systems*. 337–344.
[5] Sanjoy Dasgupta. 2006. Coarse sample complexity bounds for active learning. In *Advances in neural information processing systems*. 235–242.
[6] Sanjoy Dasgupta and Daniel Hsu. 2008. Hierarchical sampling for active learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, 208–215.
[7] Sanjoy Dasgupta, Daniel J Hsu, and Claire Monteleoni. 2008. A general agnostic active learning algorithm. In *Advances in neural information processing systems*. 353–360.
[8] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied statistics* (1979), 20–28.
[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.
[10] Rajmadhan Ekambaram. 2017. *Active Cleaning of Label Noise Using Support Vector Machines*. Ph.D. Dissertation. University of South Florida.
[11] Yoav Freund, H Sebastian Seung, Eli Shamir, and Naftali Tishby. 1997. Selective sampling using the query by committee algorithm. *Machine learning* 28, 2-3 (1997), 133–168.

[12] Dragan Gamberger, Nada Lavrac, and Saso Dzeroski. 2000. Noise detection and elimination in data preprocessing: experiments in medical domains. *Applied Artificial Intelligence* 14, 2 (2000), 205–223.

[13] Steve Hanneke. 2007. A bound on the label complexity of agnostic active learning. In *Proceedings of the 24th international conference on Machine learning*. ACM, 353–360.

[14] Dominik Henter, Armin Stahl, Markus Ebbecke, and Michael Gillmann. 2015. Classifier self-assessment: active learning and active noise correction for document classification. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*. IEEE, 276–280.

[15] Ashish Kapoor, Eric Horvitz, and Sumit Basu. 2007. Selective Supervision: Guiding Supervised Learning with Decision-Theoretic Active Learning.. In *IJCAI*, Vol. 7. 877–882.

[16] Yoon Kim. 2014. Convolutional neural networks for sentence classification. *arXiv preprint arXiv:1408.5882* (2014).

[17] Daphne Koller and Nir Friedman. 2009. *Probabilistic graphical models: principles and techniques*. MIT press.

[18] Diego Marcheggiani and Fabrizio Sebastiani. 2017. On the Effects of Low-Quality Training Data on Information Extraction from Clinical Reports. *J. Data and Information Quality* 9, 1, Article 1 (Sept. 2017), 25 pages.

[19] Andrew Kachites McCallumzy and Kamal Nigamy. 1998. Employing EM and pool-based active learning for text classification. In *Proc. International Conference on Machine Learning (ICML)*. 359–367.

[20] Ramesh Nallapati, Mihai Surdeanu, and Christopher Manning. 2009. Corractive learning: Learning from noisy data through human interaction. In *IJCAI Workshop on Intelligence and Interaction*.

[21] Natalie Parde and Rodney Nielsen. 2017. Finding Patterns in Noisy Crowds: Regression-based Annotation Aggregation for Crowdsourced Data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. 1907–1912.

[22] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100, 000+ Questions for Machine Comprehension of Text. In *EMNLP*.

[23] Rajesh Ranganath, Sean Gerrish, and David Blei. 2014. Black box variational inference. In *Artificial Intelligence and Statistics*. 814–822.

[24] Umaa Rebbapragada and Carla E Brodley. 2007. Class noise mitigation through instance weighting. In *European Conference on Machine Learning*. Springer, 708–715.

[25] B. Settles. 2012. *Active Learning*. Morgan & Claypool.

[26] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y Ng. 2008. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics, 254–263.

[27] Lucas Sterckx, Thomas Demeester, Johannes Deleu, and Chris Develder. 2014. Using active learning and semantic clustering for noise reduction in distant supervision. In *4e Workshop on Automated Base Construction at NIPS2014 (AKBC-2014)*. 1–6.

[28] Jack W Stokes, Ashish Kapoor, and Debajyoti Ray. 2016. Asking for a second opinion: Re-querying of noisy multi-class labels. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, 2329–2333.

[29] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.

[30] Simon Tong and Daphne Koller. 2001. Support vector machine active learning with applications to text classification. *Journal of machine learning research* 2, Nov (2001), 45–66.

[31] Dustin Tran, Matthew D Hoffman, Rif A Saurous, Eugene Brevdo, Kevin Murphy, and David M Blei. 2017. Deep probabilistic programming. *arXiv preprint arXiv:1701.03757* (2017).

[32] Hamed Valizadegan and Pang-Ning Tan. 2007. Kernel based detection of mislabeled training examples. In *Proceedings of the 2007 SIAM International Conference on Data Mining*. SIAM, 309–319.

[33] Martin J Wainwright, Michael I Jordan, et al. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* 1, 1–2 (2008), 1–305.

[34] Manfred K Warmuth, Gunnar Rätsch, Michael Mathieson, Jun Liao, and Christian Lemmen. 2002. Active Learning in the Drug Discovery Process. In *Advances in Neural Information Processing Systems 14*, T. G. Dietterich, S. Becker, and Z. Ghahramani (Eds.). MIT Press, 1449–1456.

[35] Shusen Zhou, Qingcai Chen, and Xiaolong Wang. 2013. Active deep learning method for semi-supervised sentiment classification. *Neurocomputing* 120 (2013), 536–546.

[36] Xingquan Zhu, Xindong Wu, and Qijun Chen. 2003. Eliminating class noise in large datasets. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 920–927.