META LABEL CORRECTION FOR LEARNING WITH WEAK SUPERVISION

Guoqing Zheng, Ahmed Hassan Awadallah, Susan Dumais
Microsoft Research
{zheng, hassanam, sdumais}@microsoft.com

ABSTRACT

Leveraging weak or noisy supervision for building effective machine learning models has long been an important research problem. The growing need for large-scale datasets to train deep learning models has increased its importance. Weak or noisy supervision could originate from multiple sources including non-expert annotators or automatic labeling based on heuristics or user interaction signals. Previous work on modeling and correcting weak labels have been focused on various aspects, including loss correction, training instance re-weighting, etc. In this paper, we approach this problem from a novel perspective based on meta-learning. We view the label correction procedure as a meta-process and propose a new meta-learning based framework termed MLC for learning with weak supervision. Experiments with different label noise levels on multiple datasets show that MLC can achieve large improvement over previous methods incorporating weak labels for learning.

1 INTRODUCTION

Recent advances in deep learning have enabled several natural language processing models to achieve impressive performance. At the core of this success lies the availability of large amounts of annotated data. However, such datasets are not readily available in large scale for many tasks. The problem of learning with weak supervision aims to address this challenge by leveraging weak evidence for supervision, such as corrupted labels, noisy labels,automatic labels based on heuristics or user interaction signals, etc.

Two major lines of work have been proposed to combine trusted (or gold) labeled data with weak or noisy supervision data for better learning. The first approach relies on re-weighting of training instances (Ren et al., 2018). It aims to assign proper importance to each sample in the training set such that the ones with higher weights will contribute more positively to the model training. On the other hand, the second approach relies on the idea of label correction. It aims to correct the noisy/corrupt labels based on certain assumptions about the weak label generation process. In a sense, label correction is a finer way to incorporate the noisy data samples than simply assigning scalar weights to each training instance and has shown to work well even in the setting where a very small set of clean labels is available. Label correction in previous methods relies on the assumption about the weak label generation process and thus often involves estimating a label corruption matrix (Hendrycks & Gimpel, 2016). However, the label correction estimation is done in separation from the main model limiting the flow of information between them.

In this paper, we address the label correction problem from a novel angle based on meta-learning and propose meta label correction (MLC). Specifically, we view the label correction procedure as a meta-process, meaning that its objective is to provide corrected labels for the examples with weak labels. On the other hand, the main supervised model is trained to fit the corrected labels (generated by the meta-model). Both the meta-model and the main model are learned by optimizing the model performances on the gold data set (i.e., a validation set w.r.t. the noisy set) allowing us to co-optimize the label correction process and the main model process.

Meta learning has been successfully used for many applications including hyper-parameter tuning (Maclaurin et al., 2015), model selection (Pedregosa, 2016) and neural architecture search (Liu et al., 2018). To the best of our knowledge, MLC is the first to utilize a meta model to automatically "tune" noisy labels from data and combine it with trusted labels for better learning. The contributions of this paper can be summarized as follows:

- A new learning framework with weak supervision based on meta learning, is proposed based on a novel angle by treating the label correction network as a meta process to provide reliable labels for the main models to learn;
- We conduct experiments on various text classification and gray-scale image recognition experiments and the proposed methods outperform previous best methods on label correction, demonstrating the power of the proposed method.

The rest of the paper is organized as follows: We briefly review the preliminaries on learning with weak supervision, particularly on learning with corrupt/noisy labels 2 and propose a meta-learning based learning framework for weak supervision in Section 3. Empirical evaluations and analysis are conducted in Section 4 and we conclude the paper in Section 6.

2 Preliminaries

We follow a setup of learning with weak supervision that involves two sets of data: a small set of data with clean/reliable labels $\{\mathbf{x}_i, \mathbf{y}_i\}_{i=1}^m$ and a large set of data with weak supervision (noisy/corrupted labels) $\{\mathbf{x}_i, \mathbf{y}'_i\}_{i=1}^M$. Typically the clean set is much smaller compared to the noisy set ($m \ll M$) due to scarcity of expert labels and to labeling costs. Training directly on the small clean set often tends to be sub-optimal, as too little data can easily cause over-fitting. The problems of learning with weak supervision under this setup can then be formulated as how to build a predictive model $f : \mathcal{X} \to \mathcal{Y}$ with the given two sets. Two major lines of work have been proposed to solve this problem.

2.1 Learning with label correction

The first line of work aims to correct the weak labels as much as possible by imposing assumptions of how the noisy labels are generated from its underlying true labels. To be concrete, consider the problem of classifying the data into k categories, where label correction involves estimating a label corruption matrix $C_{k\times k}$ whose entry C_{ij} denotes the probability of observing weak label for class i while the underlying true class label is actually j. Gold loss correction (Hendrycks et al., 2018) falls into this category; a key drawback of this line of work is that the label perturbing matrix is often estimated in an ad-hoc way and also that the estimation process is separate from the main model process, hence allowing no feedback from the main model to the estimation process.

2.2 Learning to re-weight training instances

Knowing that not all training examples are equally important and useful for building a main model, another line of work for learning with weak supervision is to assign learnable weights to each example in the training noisy set. The goal is to assign a a proper weight for each training example such that the main model would perform well on a separate validation set (the clean set) (Ren et al., 2018). The example weights are essentially hyper-parameters for the main model and can be learned by formulating a bi-level optimization problem. Due to the meta-learning characteristic of this framework, the example weights learning and the main model could communicate with each other and a better model could be learned.

3 Meta Label Correction

One advantage of the label correction approach is that it allows us to combine trusted labels and *corrected* noisy labels in the learning process. Our proposed approach adopts the label correction methodology while also co-optimizing the label correction process together with the main model process through an unified meta learning framework. We achieve that by adopting a meta-learning framework where the meta learner (meta model) tries to correct the noisy labels and the main model tries to build the best predictive model with corrected labels coming from the meta model, allowing the meta model and main model to reinforce each other.

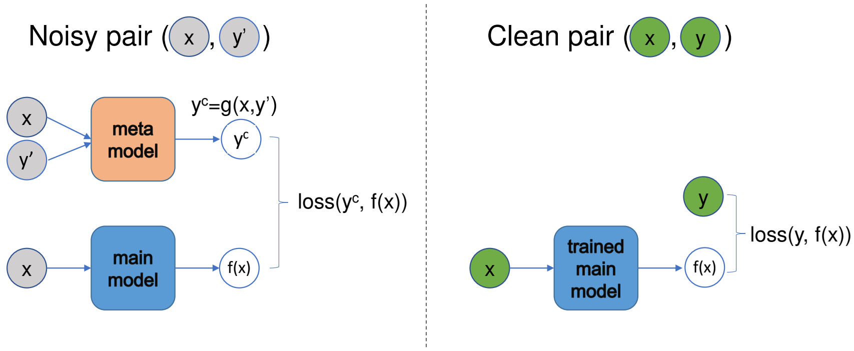

Figure 1: MLC Architecture. Nodes in gray denotes the noisy training examples and green ones denotes the clean ones.The meta model $g_{\boldsymbol{\alpha}}$ (in orange) takes in a noisy example label pair and tries to generate a "corrected" label which will be treated as the correct labels to train the main model $f_{\mathbf{w}}$ (in blue). Thus the trained main model depends on the corrected labels y_c, hence further depends on the parameters of the meta model. The trained main model then will be tested on a separate true clean set, whose loss is to be minimized. Note that when minimizing the loss on the clean examples, the parameters for the main models are not changed, only to let the loss signal on the clean examples to propogate back to the meta network, thus making $g_{\boldsymbol{\alpha}}$ generate better corrections for $\mathbf{x}, y'$. In practice, we won't be able to always get the trained main model to evaluate on the clean examples, thus k-step SGD ahead updated version of the main model is used as a "trained" model.

We describe the framework in detail as follows. Given a set of clean data examples $D = \{\mathbf{x}, \mathbf{y}\}^m$ and a set of noisy data examples $D' = \{\mathbf{x}, \mathbf{y}'\}^M$ with m much smaller than M. To best utilize the information provided by the weak labels, we propose to construct a label correction network (LCN), serving as a *meta model*, which takes a pair of noisy data example and its weak label as input and produces a different version of the weak label. Formally, the label correction network (LCN) is defined as a function with parameters $\boldsymbol{\alpha}$:

$$y_c = g_{\boldsymbol{\alpha}}(\mathbf{x}, y') \tag{1}$$

to correct the weak label y' of example $\mathbf{x}$ to its true label. (Note that the subscription in y_c emphasizes that it's generating a corrected label). Meanwhile, the *main model* f, that we aim to train and use for prediction after training, is instantiated as another function with parameters $\mathbf{w}$,

$$y = f_{\mathbf{w}}(\mathbf{x}) \tag{2}$$

Without linking the two models, there's no way to enforce that 1.) the generated label for an example from the meta model g is indeed a meaningful one, let alone a corrected one, since we cannot directly train the meta model without clean labels for the noisy examples ; 2) The main model f might be fitting onto arbitrary targets, if the labels provided do not align with the unknown true labels. Fortunately, the two models can be linked together via a bi-level optimization problem, by the intuition that *if the labels generated by the meta model are of high quality, then we can use these pairs of examples and their corrected labels as training data to train a good main model, such that the main model achieves low loss on a separate set of clean examples*. This can be instantiated as the following bi-level optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \mathbb{E}_{(\mathbf{x},y)\in D}\, \ell(y, f_{\mathbf{w}^*_{\boldsymbol{\alpha}}}(\mathbf{x})) \\ s.t.\mathbf{w}^*_{\boldsymbol{\alpha}} &= \arg\min_{\mathbf{w}} \mathbb{E}_{(\mathbf{x},y')\in D'}\, \ell(g_{\boldsymbol{\alpha}}(\mathbf{x}, y'), f_{\mathbf{w}}(\mathbf{x})) \end{aligned} \tag{3}$$

where $\ell()$ denotes a chosen differentiable loss function to measure the predictive error and the subscript of $\mathbf{w}$ is to emphasize the dependency of the best main model f on $\boldsymbol{\alpha}$. We term this framework as Meta Label Correction (MLC); Figure 1 provides an overview of the framework.

In this bi-level formulation, since the LCN is parameterized by $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}$ are the upper parameters (or meta parameters) while the main model parameters $\mathbf{w}$ are the lower parameters (or main parameters). Like many other work involving bi-level optimizations, exact solutions of Problem (3) requires solving for the optimal $\mathbf{w}^*$ whenever $\boldsymbol{\alpha}$ is updated. This is often analytically infeasible when the

while *not converged* **do**
  Update meta parameters $\boldsymbol{\alpha}$ by descending $\nabla_{\boldsymbol{\alpha}}\mathcal{L}_D(\mathbf{w} - k\eta\nabla_{\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w}))$
  Update model parameters $\mathbf{w}$ by descending $\nabla_{\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w})$
end

Algorithm 1: MLC - Meta Label Correction

main model f is complex, such as deep neural networks, and also computationally expensive. Instead of solving for the optimal for $\mathbf{w}^*$ for each $\boldsymbol{\alpha}$, we use a k-step look ahead SGD update for $\mathbf{w}$ as an estimate to the optimal main model for a given $\boldsymbol{\alpha}$

$$\mathbf{w}'(\boldsymbol{\alpha}) \approx \mathbf{w} - k\eta\nabla_{\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w}) \tag{4}$$

where $\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w}) = \mathbb{E}_{(\mathbf{x},y')\in D'}\,\ell(g_{\boldsymbol{\alpha}}(\mathbf{x}, y'), f_{\mathbf{w}}(\mathbf{x}))$, thus the proxy optimization problem turns to

$$\min_{\boldsymbol{\alpha}}\ \mathcal{L}_D(\mathbf{w}'(\boldsymbol{\alpha})) = \mathcal{L}_D(\mathbf{w}^*(\boldsymbol{\alpha})) = \mathbb{E}_{(\mathbf{x},y)\in D}\,\ell(y, f_{\mathbf{w}'(\boldsymbol{\alpha})}(\mathbf{x})) \tag{5}$$

Algorithm 1 outlines an iterative procedure to solve the above proxy problem with k-step look ahead SGD for the main model:

The above meta learning algorithm involves computing an expensive second-order partial derivative $\nabla^2_{\boldsymbol{\alpha},\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w})$ followed by a matrix vector product. To speed up training, we propose to approximate the second order gradients with finite differences, as follows

$$\begin{aligned}
\nabla_{\boldsymbol{\alpha}}\mathcal{L}_D(\mathbf{w} - k\eta\nabla_{\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w})) &= -k\eta\nabla^2_{\boldsymbol{\alpha},\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w})\nabla_{\mathbf{w}'}\mathcal{L}_D(\mathbf{w}') && (6)\\
&= -k\eta\nabla_{\boldsymbol{\alpha}}\left(\nabla^\top_{\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w})\nabla_{\mathbf{w}'}\mathcal{L}_D(\mathbf{w}')\right) && (7)\\
&\approx -\frac{k\eta}{2\epsilon}\left(\nabla_{\boldsymbol{\alpha}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w}^+) - \nabla_{\boldsymbol{\alpha}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w}^-)\right) && (8)
\end{aligned}$$

where $\mathbf{w}^\pm = \mathbf{w} \pm \epsilon\nabla_{\mathbf{w}'}\mathcal{L}_D(\mathbf{w}')$, and $\mathbf{w}' = \mathbf{w}' - k\eta\nabla_{\mathbf{w}}\mathcal{L}_{D'}(\boldsymbol{\alpha}, \mathbf{w})$. Similar approximation strategy is also adopted by meta-learning related tasks. (Liu et al., 2018; Finn et al., 2017)

3.1 CONVERTING A CLASSIFIER TO A LABEL CORRECTION NETWORK

There are multiple ways to build mappings from (x, y') to the corrected label with deep neural networks as the desired label correction network. In essence, $g_{\boldsymbol{\alpha}}(x, y')$ behaves also like a classifier with the only difference from conventional classifiers, i.e., it also takes the noisy label y' as input. To ease the effort of designing and working with the MLC framework, we explored several simple strategies, which doesn't require heavy-weight modifications to existing architectures. The one that is adopted in all our experiments is by constructing the LCN as a weighted combination from a classifier $h(x)$ and the weak label y' itself, i.e.

$$g(x, y') \equiv \lambda(x)h(x) + (1 - \lambda(x))y' \tag{9}$$

where λ is a data dependent scalar controlling the mixing weights. We found it helps to have a separate λ for each class, hence different weak classes fed in will be paired with different weights. If doing so, this only requires modifying the last layer of the classifier $h(x)$, to output a vector of dimension $2C$ (C dimensions for the class logits, and the rest C dimensions for the weak label weights λ), instead of C for the original classifier (where C is number of classes),

3.2 SOFT CROSS ENTROPY LOSS FOR LEARNING WITH WEAK SUPERVISION

In the classification scenario, when a clean label is given to a data example typically the cross entropy loss is used to train the classifier. Here, we demonstrate that with a soft label (generated from the label correction network), how the soft cross entropy loss could be beneficial for the weakly supervised setting. Denote the the soft label as $\mathbf{q}$, where $\mathbf{q}$ is a dense vector with $\sum_i q_i = 1$, typically resulted from a softmax layer and denote the predicting probability of the main classifier as $\mathbf{p}$ with $\sum_i p_i = 1$. In this setup, the original cross entropy loss defined for hard labels can be naively extended as

$$CE_{\text{soft}}(\mathbf{p}, \mathbf{q}) = -\sum_i q_i \log p_i = \sum_i q_i \log\frac{q_i}{p_i} - \sum_i q_i \log q_i = \text{KL}(\mathbf{q}, \mathbf{p}) + \text{entropy}(\mathbf{q}) \tag{10}$$

Minimizing this loss w.r.t the parameters of the main model is equivalent to (with the meta model fixed, thus $\mathbf{q}$ fixed) minimizing the KL divergence between the (soft) label and the predicting distribution, similar to the hard label case. And when updating the parameters of the meta model, minimizing this loss function is now equivalent to (with the main model fixed, thus $\mathbf{p}$ fixed) minimizing both the KL divergence between the (soft) label and the predicting distribution, and also the entropy of the soft labels predicted by the meta model, since we would like to have labeling distribution as close to discrete as possible.

3.3 k-STEP LOOK AHEAD SGD LOOK AHEAD IN META MODEL LEARNING

We found it's crucial to use a value of k greater than 1 for MLC to ensure model convergence, particularly in the early phases of training, when both the main model and the meta model are close to random predictors and lacks confidence in their outputs. Using $k = 1$ is likely to confuse both models and thus they won't converge. This is not the case for previous similar works with meta-learning, however we find this to be crucial, as the main model in GLC is not directly trained on any clean examples, thus slightly more explorations from the main model is likely to help training convergence. We'll explore this aspect in the coming section. Due to this requirement, in all our experiments, we use scheduling for k starting from 1500 and decreasing to 500 towards the end of model training.

4 EXPERIMENTS

To test the performance of MLC, we conduct experiments on a set of classification tasks, both from the text and vision domains, and compare results with previous state-of-the-art approaches for learning with weak/noisy labels, under different weak label scenarios.

4.1 WEAK SUPERVISION GENERATION

To generate weak supervision data, for each data set we test on, we sample a portion of the entire training set as the clean set. The noisy dataset is generated by corrupting the labels of all the remaining data points based on one of the following two(three) settings:

- Uniform mixture (UNIF)
- Flipped labels (FLIP)
- Weak labels from trained weak classifiers (WEAK)

The first two methods follow the same procedure adopted by (Hendrycks et al., 2018)by either corrupting uniformly all classes or by flipping a label to a different class. To generate the corrupted labels from the true labels, we first devise a corruption probability categorical distribution for each true class, hence for all classes the corruption probability forms a label corruption matrix C. Then, for an example with true label i, we sample the corrupted label from the categorical distribution parameterized by the ith row of C. Note that this is a simplified assumption assuming that the corrupted label does not depend on the data example itself, however we still use this to ensure a fair comparison to (Hendrycks et al., 2018) where the same process was used for generating noisy data.To create a noisy datasets with different levels of noise, we take a convex combination of an identity matrix and the corruption matrix, with the coefficient of the latter serving as an indicator of the noise level (Hendrycks et al., 2018).

To also study scenarios where the noise could be dependent on both the data and the label, we introduce a third more realistic method: WEAK. In this method, weak labels are provided by separate (weak) predictive models that depend on both the data and the labels. To generate different levels of noise, we train multiple weak predictive models with varying accuracies where a lower accuracy corresponds to a higher noise level. Note that in all noise levels, the weak predictive model is performing better than random prediction.

Note that all method are not aware of this true label corruption probability nor do they have knowledge about which data sample in the noisy set is actually corrupted. Since UNIF and FLIP are similar to one another, we report results based on UNIF only and leave the FLIP results for the appendix.

4.2 Baseline methods

We test MLC mainly against the current state-of-the-art model (Hendrycks et al., 2018) for label correction (denoted by GLC hereafter) in various settings where the labels in the noisy set are corrupted by different noise levels, as well as the different ratios the clean set is sampled from the entire training set. Note that GLC was shown to perform consistently better than other models that combine the clean and weak labels using methods such as distillation Li et al. (2017). For completeness, we also compare with the forward loss correction method proposed in (Sukhbaatar et al., 2014) (denoted by Forward hereafter). In lieu with meta learning for learning with weak supervision, we also compare with the method of learning weights of training examples to learn a robust classifier (Ren et al., 2018) (denoted by L2R hereafter).

4.3 Data sets and implementation details

To ensure fair comparison with previous methods as much as possible, we experiment on a broad set of data collections from both , with our best effort to match the pre-processing and hyper-parameter setting from previous methods when experimenting with them on new datasets that were not used in the original papers.

We test on 10 different collections with varying data sizes. To compare with GLC, we test on all three text collections used by (Hendrycks et al., 2018) and on the MNIST dataset. The dataset are::

MNIST: The MNIST dataset contains 28×28 grayscale images of the digits 0-9. The training set has 60,000 images and the test set has 10,000 images. For preprocessing, we rescale the pixels to the interval [0, 1]. We train a 2-layer fully connected network with 128 hidden dimensions. We train with Adam for 20000 iterations using batches of size 128 and a learning rate of 0.001 for the main model and 0.0001 for the meta model.

Twitter: The Twitter Part of Speech dataset (Gimpel et al., 2011) contains 1,827 tweets annotated with 25 POS tags. This is split into a training set of 1,000 tweets, a development set of 327 tweets, and a test set of 500 tweets. We use the development set to augment the training set. We use the same pretrained 50-dimensional word vectors as in (Hendrycks et al., 2018), and for each token, we concatenate word vectors in a fixed window centered on the token. These form our training and test set. We use a window size of 3, and train a 2-layer fully connected network with hidden size 256, and use the GELU nonlinearity (Hendrycks & Gimpel, 2016). We train with Adam (Kingma & Ba, 2014) for 20000 iterations with batch size 128 and learning rate of 0.001 for the main model and 0.0001 for the meta model. Wwe use ℓ_2 weight decay with $\lambda = 3 \times 10^{-4}$ on all the weights.

SST: The Stanford Sentiment Treebank dataset consists of single sentence movie reviews (Socher et al., 2013). We use the 2-class version (i.e. SST2), which has 6,911 reviews in the training set, 872 in the development set, and 1,821 in the test set. We follow the same data and model setups as in g (Hendrycks et al., 2018); the classifier is a word-averaging model with an affine output layer. We use the Adam optimizer for 10000 epochs with batch size 50 and learning rate 0.001. For regularization, we use ℓ_2 weight decay with $\lambda = 1 \times 10^{-4}$ on the output layer.

IMDB: The IMDB dataset contains 50k movie reviews from IMDB, with 25k positive and 25k negative. We use a one-layer LSTM (Hochreiter & Schmidhuber, 1997) for both main model and the meta model.

The above datasets are relatively small text collections, though meaningful to demonstrate the different methods for learning with weak supervision with simple classifier architectures. We also include a range of 6 large scale text classification benchmark datasets including:

AG News: AG is a text classification dataset derived from a large collection of news articles gathered from more than 2000 news sources. Each news article is categorized to 1 of 4 classes, including World, Sports, Business and Sci/Tech.[1]. This dataset has 120,000 training examples and 7,600 examples for testing.

Amazon Reviews (Amazon-2 and Amazon-5): The Amazon-2 and Amazon-5 datasets contain randomly sampled customer reviews from Amazon. The Amazon-2 is a binary polarity rating classification dataset while Amazon-5 is a rating classification dataset on a scale from 1 to 5. The

[1] http://www.di.unipi.it/~gulli/AG_corpus_of_news_articles.html

Table 1: Overview of network architecture used for main model and meta model

Data set	Meta network	Main network
Twitter, SST	Embedding Avg	Embedding Avg
MNIST	3-layer MLP	3-layer MLP
IMDB	1-layer LSTM	1-layer LSTM
AG, Amazon2, Amazon5, Yelp2, Yelp5, DBpedia	BERT-base	BERT-base

Amazon-2 dataset has 3,600,000 training samples and 400,000 testing samples, while the Amazon-5 dataset has 3,000,000 trainig samples and 650,000 testing samples.

Yelp Reviews (Yelp-2 and Yelp-5): The Yelp-2 and Yelp-5 datasets are constructed from the Yelp Dataset Challenge 2015 data, for binary polarity rating classification and 5-way rating classification, respectively. Yelp-2 contains 560,000 training samples and 38,000 testing samples and Yelp-5 contains in total of 650,000 training samples and 50,000 testing samples.

DBpedia DBpedia is a crowd-sourced project aiming to extract structured information from Wikipedia. The DBpedia dataset was covers 14 non-overlapping ontology classes from DBPedia. Each class contains 40,000 training samples and 5,000 testing samples. Hence, the full dataset has 560,000 training samples and 70,000 testing samples.

For all the large scale text classification datasets (AG, Amazon-2 and -5, Yelp-2 and 5 and DBpedia), we adopt a pre-trained BERT-base (Devlin et al., 2018) model for both the main network and meta network. This ensures that we can test the ability of MLC in the weakly supervised setting with strong state-of-the-art base models.

We implement all models and experiments in PyTorch[2]. To ensure fair comparison, we adopt the same main network architecture as much as possible from previous best methods with comparable number of parameters. A brief overview of the neural net architectures used in various settings is listed in Table 1 (Refer to the appendix for a detailed description of the model architectures). Code for reproducing the results in this paper will be made publicly available.

4.4 Main Results

MLC with MLP, LSTM: We investigate multiple settings with an extensive set of different configurations, i.e., two noise types, different noise levels, and different clean ratios. Table 2 presents the averaged accuracies across all these configurations with each one repeated for 5 times. Notice that the results vary per dataset when the news is generated using UNIF (i.e. noise is independent of the data). On the other hand, we notice that the performance of all methods seems to drop when we use WEAK (noise depends on the data and the label). This shows that this is a more realistic and challenging settings. We also observe that MLC performs consistently better in this case.

Table 2: Mean accuracies over an extensive set of experimental configurations. Each cell represents an average over 2 noise types (UNIF and WEAK), 3 clean ratios(0.1%, 1.0% and 5%), 11 noise levels for UNIF (0 - 1.0 with 0.1 step and 3 different weak classifiers for WEAK. Every experiment was repeated 5 times

	Datasets	**Twitter**	**SST**	**IMDB**	**MNIST**
UNIF	Forward	0.484	**0.739**	0.735	0.844
	GLC	0.743	0.736	**0.739**	**0.924**
	L2R	0.763	0.614	0.702	0.905
	MLC	**0.780**	0.646	0.712	0.855
WEAK	Forward	0.226	0.631	0.626	0.407
	GLC	0.295	0.615	**0.628**	0.451
	L2R	0.435	0.592	0.628	0.608
	MLC	**0.729**	**0.635**	0.623	**0.843**

[2] https://pytorch.org

MLC with BERT. Table 3 presents the error rates of MLC on 6 large text data sets with pre-trained BERT-base as its main model and meta models. Note that these are much larger scale dataset and that the baselines and the base models for both the meta and the main learner are using Bert. We notice that MLC consistently outperforms the baselines (except for L2R on the AG dataset).

Table 3: Error rates comparison on 6 large text data sets

Datasets Fully supervised (# labeled examples)	**AG** (120k)	**Yelp-2** (560k)	**Yelp-5** (650k)	**Amazon-2** (3.6m)	**Amazon-5** (3m)	**DBpedia** (560k)
$\text{BERT}_{\text{LARGE}}$ (Xie et al., 2019)	-	1.89	29.32	2.63	34.17	0.64
SSL (# labeled examples)		(20)	(2.5k)	(20)	(2.5k)	(140)
$\text{BERT}_{\text{BASE-512}}$	-	13.60	41.00	26.75	44.09	2.58
$\text{BERT}_{\text{LARGE-512}}$	-	10.55	38.90	15.54	42.30	1.68
WSL (# labeled examples, $p = 0.6$)	(60)	(20)	(2.5k)	(20)	(2.5k)	(140)
GLC - $\text{BERT}_{\text{BASE-128}}$	18.74	8.87	44.79	10.19	48.08	3.10
L2R - $\text{BERT}_{\text{BASE-128}}$	**8.37**	10.00	38.74	10.77	42.93	2.08
MLC - $\text{BERT}_{\text{BASE-128}}$	9.25	**8.18**	**37.69**	**9.54**	**42.53**	**1.70**

4.5 Detailed Results

We investigate how the noise levels in the weak labels affect MLC training. Due to space limitations, we only present detailed results on Twitter and MNIST. Detailed experiments on SST and IMDB can be found in the appendix.

Twitter. Figure 2 presents the detailed performances with different clean data ratio and label noise levels. For the UNIF setting, both loss correction methods (GLC and MLC) works better than using only clean data to train the classifier, emphasizing the importance of incorporating those weakly supervised examples. With 1% and 5% only clean data, MLC achieves consistently higher accuracies over the range of high noise levels, implying the robustness of MLC with severe noise present. In the WEAK setting, where the weak labels are generated by weak classifiers, GLC performs worse than MLC since it assumes that the noisy labels are only dependent on the true label but not on the data. In contrast, MLC gains significant edge over the other methods as MLC doesn't make such assumptions.

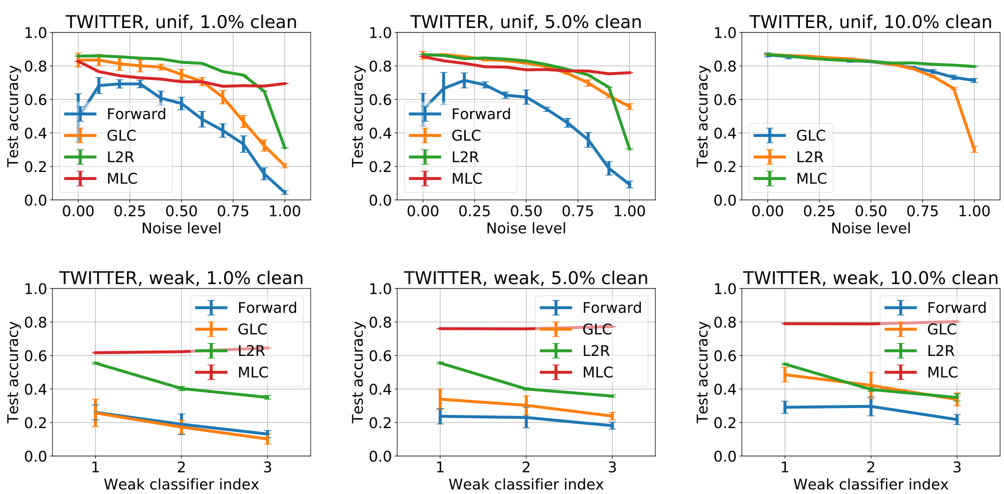

Figure 2: Results on Twitter. All numbers reported are accuracies on the test set.

MNIST. Figure 3 presents the detailed performances with different gold data ratio and corruption levels. Similar trends could be observed as previously seen in Twitter and SST. On the UNIF setting, MLC is not as good as GLC and L2R; however in the upper range of noise levels, MLC catches up

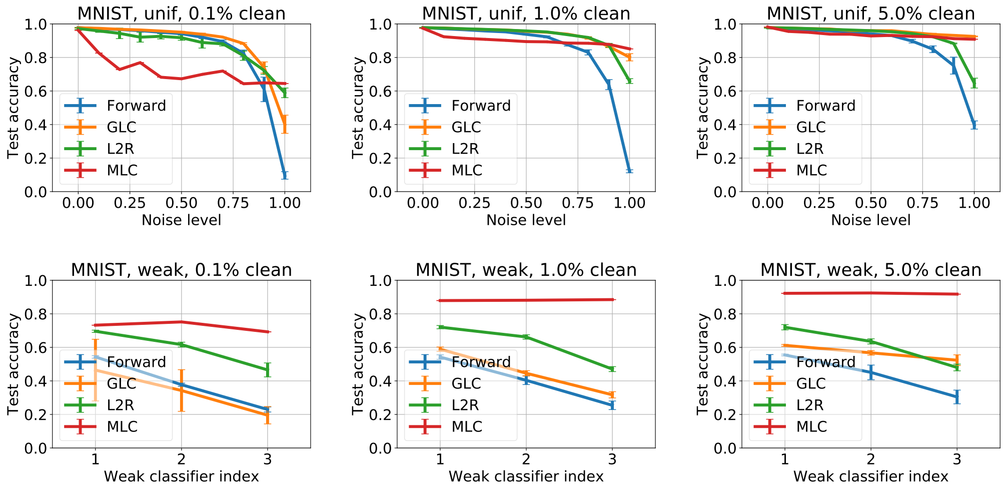

Figure 3: Results on MNIST. All numbers reported are accuracies on the test set. Best results in terms of mean accuracies are printed in black.

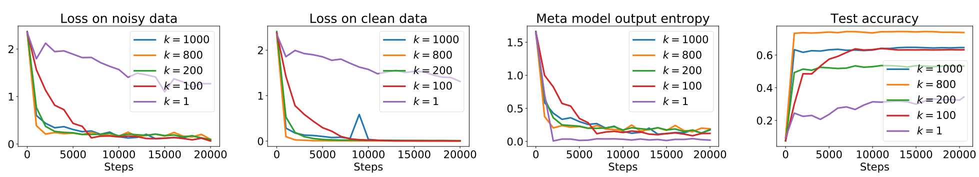

Figure 4: (a) Loss curve on noisy data; (b) Loss curve on clean data; (c) Entropy of the label correction distribution from the meta model; (d) Test set accuracy changes over the iterations

and leads the way for both GLC and L2R. Moreover, in the WEAK setting, GLC loses to MLC again due to its simplistic assumption about the noises with a large margin.

4.6 Analysis and ablation studies

In this section, we tap into the details of how MLC behaves in terms of training dynamics and what the meta networks learns.

4.6.1 Training dynamics

Figure 4 shows the training progress for one run on the MNIST data sets. We monitor a set of different metrics in training, including the loss function on the noisy data (thus with corrected labels), loss function on clean data, the entropy of the output distribution from the meta-model (since it's a soft label). Another key factor is the parameter k for the look ahead SGD. It turned out that with $k = 1$ the model basically diverges, thus picking a value larger than 1 is crucial to MLC training.

4.6.2 Meta net evaluation

After training, besides the main model that serve as the predictive for inference, we also obtain the meta model, a trained label correction network. In this section, we investigate what actually the meta model learns after convergence. To achieve this we follow the UNIF setting, i.e., we corrupt the labels for examples in the test set according to the label corruption matrix used in the weak label generation process and feed the corrupted test pair into the LCN to check it could recover the correct label. Note that by doing this we ensure that the MLC framework doesn't see the instance in training.

It's clear that after training, both main and meta models have the ability to predict correct labels. The main network could be used for prediction, while the other serve as a good label correction network.

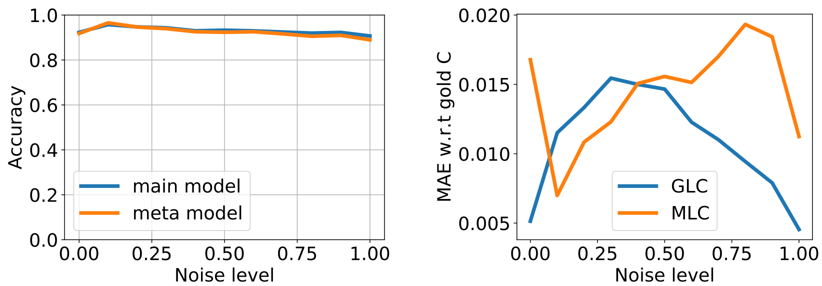

Figure 5: (Left) We test the label correction ability of the trained meta model from MLC as a classifier taking a pair of data and its noisy label as input and assess its accuracy. For reference, we also plot the accuracies for corresponding trained main model; (Right) Comparison of discrepancies of the estimated label corruption matrix against the ground truth one. Both GLC and MLC are shown.

By measuring the MAE of the estimated label corruption matrix, we verify that the corrected label distribution aligns well to the ground truth for the UNIF setting (so is GLC, however GLC cannot be used to correct future unseen examples).

5 Related Work

Labeled data largely determines whether a machine learning system can perform well on a task or not, as noisy label or corrupted labels could cause dramatic performance drop (Nettleton et al., 2010). The problem gets even worse when an adversarial rival intentionally injects noises into the labels (Reed et al., 2014). Thus, understanding, modeling, correcting, and learning with noisy labels has been of interest at large in the research communities (Natarajan et al., 2013; Frénay & Verleysen, 2013).

Several works (Mnih & Hinton, 2012; Patrini et al., 2017; Sukhbaatar et al., 2014; Larsen et al., 1998) have attempted to address the weak labels by modifying the model's architecture or by implementing a loss correction. (Sukhbaatar et al., 2014) introduced a stochastic variant to estimate label corruption, however the methods have to have access to the true labels, rendering it inapplicable when no true labels are present. A forward loss correction adds a linear layer to the end of the model and the loss is adjusted accordingly to incorporate learning about the label noise. (Patrini et al., 2017) also make use of the forward loss correction mechanism, and propose an estimate of the label corruption estimation matrix which relies on strong assumptions, and does not make use of clean labels that might be available for a portion of the data set..

Following (Charikar et al., 2017), we assume that during training the model has access to a small set of clean labels besides a large set of weak labels. This assumption has been leveraged by others for the purpose of label noise robustness, most notably (Veit et al., 2017; Li et al., 2017; Xiao et al., 2015; Ren et al., 2018). (Veit et al., 2017) use human-verified labels to train a label cleaning network by estimating the discrepancies between the noisy and clean labels in a multi-label classification setting. This assumes that, for a subset of the data, both trusted and noisy labels are available. This work avoids this limitation by proposing a meta learning approach that does not require trusted and noisy data to be available for the same instances.

6 Conclusions

In this paper, we address the problem of learning with weak supervision from a meta-learning perspective. Specifically, we propose to use a meta network to correct the noisy labels from the noisy data set, and a main classifier network is trained to fit the example to a provided label, i.e., corrected labels for the noisy examples and true labels for the clean examples. The meta network and main network are jointly optimized in a bi-level optimization framework; to address the computation challenge, we employ a k-step ahead SGD update for the model weights of the main model. Empirical experiments on several benchmark datasets including text and graysacle images demonstrates the effectiveness of MLC.

References

Moses Charikar, Jacob Steinhardt, and Gregory Valiant. Learning from untrusted data. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, pp. 47–60. ACM, 2017.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2013.

Kevin Gimpel, Nathan Schneider, Brendan OConnor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. Part-of-speech tagging for twitter: Annotation, features, and experiments. *ACL HLT 2011*, pp. 42, 2011.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Dan Hendrycks, Mantas Mazeika, Duncan Wilson, and Kevin Gimpel. Using trusted data to train deep networks on labels corrupted by severe noise. In *Advances in Neural Information Processing Systems*, pp. 10477–10486, 2018.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8): 1735–1780, 1997.

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

Jan Larsen, L Nonboe, Mads Hintz-Madsen, and Lars Kai Hansen. Design of robust neural network classifiers. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181)*, volume 2, pp. 1205–1208. IEEE, 1998.

Yuncheng Li, Jianchao Yang, Yale Song, Liangliang Cao, Jiebo Luo, and Li-Jia Li. Learning from noisy labels with distillation. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International Conference on Machine Learning*, pp. 2113–2122, 2015.

Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *Proceedings of the 29th International conference on machine learning (ICML-12)*, pp. 567–574, 2012.

Nagarajan Natarajan, Inderjit S Dhillon, Pradeep K Ravikumar, and Ambuj Tewari. Learning with noisy labels. In *Advances in neural information processing systems*, pp. 1196–1204, 2013.

David F Nettleton, Albert Orriols-Puig, and Albert Fornells. A study of the effect of different types of noise on the precision of supervised learning techniques. *Artificial intelligence review*, 33(4): 275–306, 2010.

Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1944–1952, 2017.

Fabian Pedregosa. Hyperparameter optimization with approximate gradient. *arXiv preprint arXiv:1602.02355*, 2016.

Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. *arXiv preprint arXiv:1803.09050*, 2018.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1631–1642, 2013.

Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

Andreas Veit, Neil Alldrin, Gal Chechik, Ivan Krasin, Abhinav Gupta, and Serge Belongie. Learning from noisy large-scale datasets with minimal supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 839–847, 2017.

Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2691–2699, 2015.

Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Unsupervised data augmentation for consistency training. *arXiv preprint arXiv:1904.12848*, 2019.

A ADDITIONAL RESULTS

A.1 SST AND IMDB

SST. Figure 6 presents the detailed performances with different gold data ratio and corruption levels. On this binary classification task, it's surprising to observe that using clean data solely is only achieving results that are slightly better than random guessing (an accuracy of 0.5). Again with the help of label correction for the noisy examples, the performance boosts by quite a margin with GLC. So does MLC over GLC, demonstrating the potential power of using a meta network as a label correction procedure.

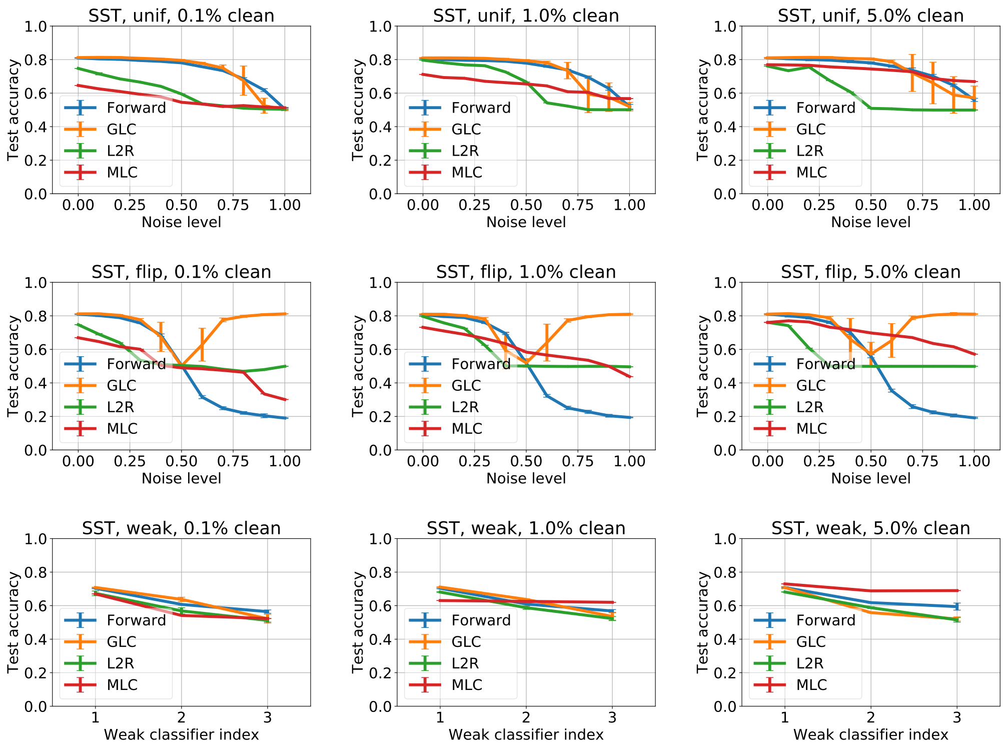

Figure 6: Results on SST. All numbers reported are accuracies on the test set. For references, using gold data only to train a model yields test accurucies of 0.541?, 0.647? and 0.741?, for three gold data ratios respectively.

IMDB. Figure 7 presents the detailed results on all three noisy settings.

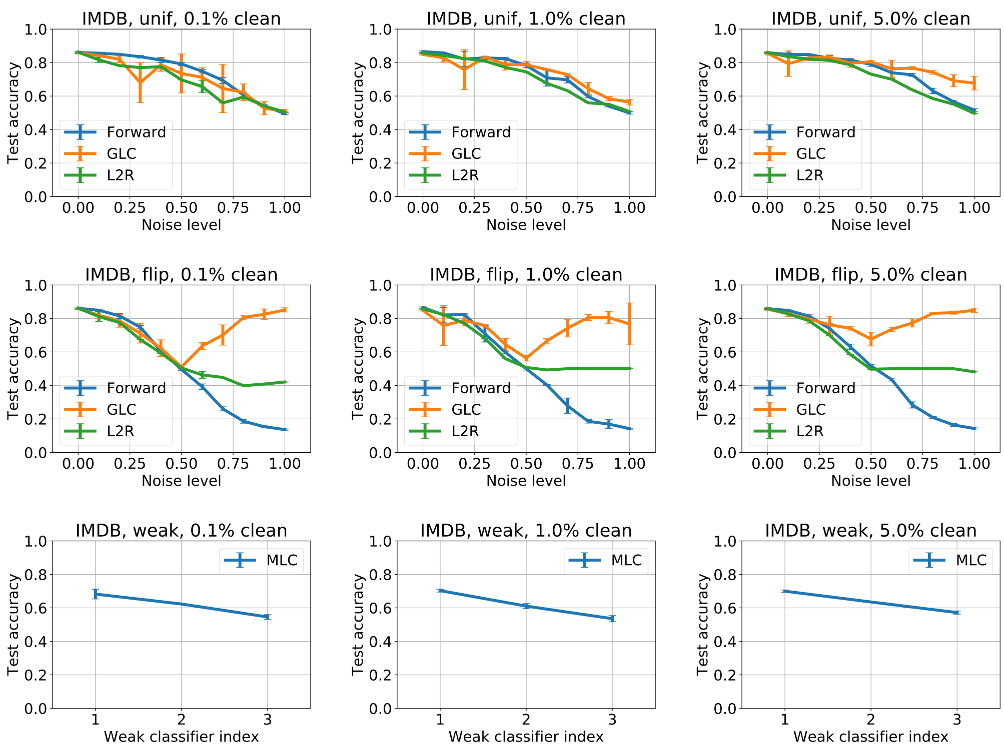

Figure 7: Results on IMDB. All numbers reported are accuracies on the test set. For references, using gold data only to train a model yields test accurucies of (0.541?, 0.647? and 0.741?, for three gold data ratios respectively.