Reweighing auxiliary losses in supervised learning

Durga Sivasubramanian¹,², Ayush Maheshwari¹, Pradeep Shenoy², Prathosh AP³, and Ganesh Ramakrishnan¹

¹Indian Institute of Technology, Bombay
²Google Research, India
³Indian Institute of Science, Bengaluru

June 8, 2022

Abstract

Apart from the standard supervised learning using hard labels, often auxiliary losses are used in many supervised learning settings to improve the model’s generalisation. For example, knowledge distillation adds a second, teacher mimicking loss to the training of a model, where the teacher may be a pretrained model that outputs a richer distribution over labels. Similarly, in settings with limited labelled data, weak labelling information is used in form of labelling functions. Auxiliary losses are introduced here to combat labelling functions that may be noisy rule-based approximations of true labels. We tackle the problem of learning to combine these losses in a principled manner. We introduce Amal which learns instance-specific weights using meta learning on a validation metric to achieve optimal mixing of losses. Experiments in a number of knowledge distillation and rule denoising domains show that Amal provides noticeable gains over competitive baselines in those domains. We empirically analyze our method and share insights into the mechanisms through which it provides performance gains.

1 Introduction

Deep learning techniques have shown significant impact in a wide range of machine learning applications, driven primarily by the availability of large amounts of reliable labeled data [38]. Despite this progress, supervised learning faces challenges: first, the time and effort needed to obtain large, reliable labeled datasets, and second, the limited information contained in human-annotated labels. Many approaches aim to improve generalization and sample efficiency of supervised learning by incorporating additional sources of information, or learning constraints, into the supervised learning paradigm. For instance, rule-denosing techniques [33, 7] use simple, approximate labeling rules (labeling functions) that provide weak supervision and reduce dependence on data annotation. Other work has combined learning from labeling functions with supervised learning from limited human-annotated data [26, 21, 4, 35, 34]—these approaches leverage the supervised learning objective to offset the noisy labels from labeling functions. A challenge here is how to optimally combine these complementary objectives.

Equally, cardinal labels do not capture the richness of information contained in the input data—e.g., object category labels for images of natural scenes. Some of this imprecision can be mitigated by using more nuanced ‘soft labels’, or distributions over labels, as the target for supervision instead of
the cardinal labels. Knowledge distillation (KD) [17]) proposes using the inherent uncertainty of a supervised model trained on cardinal labels (the “teacher model”) to generate these soft labels for training, in combination with the traditional supervision loss. Indeed, recent work [28] formalizes this process from a Bayesian perspective, showing that when one-hot labels are an imperfect representation of the true probability distribution, KD reduces the variance associated with probability estimates in a student model. Other work examines, from an empirical perspective, when and how distillation may improve upon training from scratch on the labels alone. For instance, an overtrained teacher will likely achieve low/zero error rates w.r.t. the (incomplete) label loss simply by overfitting on random noise in the dataset; in these circumstances, the probabilities output by the teacher do not accurately represent the underlying uncertainty, and students may be led astray.

We propose AMAL, an adaptive loss mixing technique for addressing the challenge of optimally combining supervised learning objectives with these varied auxiliary objectives. Our proposal is driven by the following key insight: the mixing of primary and auxiliary objectives greatly benefit by being regulated on a sample-by-sample basis. This draws from substantial literature showing the promise of instance-reweighting, for example in handling noisy labels or outliers [6, 34]. We therefore propose to learn instance-specific mixing parameters that combine complementary learning objectives. We devise a meta-learning algorithm, based on a separate validation metric, to estimate these instance-specific parameters in an unbiased manner. We demonstrate how our method yields more accurate models when rule-based losses are mixed with limited supervision losses [26] as well as in a knowledge distillation setting (KD [17]).

Our Contributions: Our key contributions are as follows:

1. We propose a general formulation for instance-specific mixing in of auxiliary objectives in supervised learning. This is, to our knowledge, the first proposal of its kind.

2. Amal in rule-denoising setting with limited supervision We show how the problem of semi-supervised data programming can benefit from Amal and report gains of 2-5% on various datasets.

3. Amal in Knowledge Distillation setting We explore a range of settings in KD, including vanilla KD, multi-teacher KD, and early-stopping, showing significant gains over and above SOTA KD approaches in these settings.

2 Related work

Knowledge distillation (KD) KD [17] in a supervised learning setting trains a “student” model to mimic the outputs of a larger, pre-trained “teacher” model instead of directly training on the supervised signal. Readers are encouraged to read [12] for a comprehensive survey of distillation methods and their applications. The efficacy of KD can be limited by teacher accuracy (see [28] for some theoretical results), and student representational capacity, among other factors. Interestingly, early stopped teacher models aid better in training the student models [8]; however, identifying the best possible teacher requires repeating the distillation process multiple times on the student model. To bridge the representational gap between the teacher and the students, Teacher Assistants (TA) or intermediate models were introduced [29], and were improved by a stochastic approach (DGKD [37]) for simultaneously training all intermediate models with occasional model dropout. In [24], multiple teacher networks are used with an intermediate knowledge transfer step using latent distillation. To increase robustness, adversarial data augmentation has been explored [16].

Instance-Specific Learning: Significant literature has explored instance-specific learning, for instance in instance hardness identification [45, 46], based on their ease of classification. Some
approaches develop *curricula* for learning from the “right” sequence of easy and difficult examples [46], or instance-specific temperature parameters in supervised learning [36]. Other closely related work [2, 39] learns a per-instance *label uncertainty* parameter to account for potential label noise. In the distillation setting, too, [40, 41], employ adversarial learning for matching the distributions of teacher and student logits, and [44] demonstrate the benefits of learning an instance-level sequence (or curriculum) on training samples in the context of knowledge distillation. In [6], the authors propose a task-agnostic per-sample loss-function representing the reliability of each prediction. Motivated by these, we seek task-agnostic, per-sample, loss mixing coefficients for effective learning.

**Bi-level Optimization and Meta-Learning:** Prior work [20, 5, 9] has explored learning network hyper-parameters via solving a two-level optimization problem—one on the base-task and another on an external model-selection or meta-task, often on validation data. While methods like [5] learn model hyper-parameters, in [20] new hyper-parameters are introduced at each iteration by assigning weights to stochastic gradient samples. These algorithms are similar in spirit to the *learning to learn* literature, typically in multi-task contexts [10, 30, 18, 39]. Typical approaches aim to learn a “meta-algorithm which can generalize across tasks by mimicking the test dynamics (sampling test tasks, in addition to test data, for measuring and optimizing loss) during training [18]. Although this literature, too, employs nested optimization objectives, it differs from our work in that we wish to improve generalization within a single task, rather than across tasks.

### 3 AMAL: Adaptive Mixing of Auxiliary Losses

We consider the scenarios where there are two or more loss terms involved in a supervised learning setting. The loss functions we consider adhere to the form specified in Eq. 1, where there is a primary objective and $K$ auxiliary objectives.

$$
\mathcal{L} = \lambda_p \mathcal{L}_p + \sum_{k=1}^{K} \lambda_{a_k} \mathcal{L}_{a_k}
$$

(1)

Here, $\mathcal{L}_p$ and $\mathcal{L}_{a_k}$ respectively are the primary and auxiliary loss objectives. While this formulation is general, in this paper, we explicate the formulation in two different settings – knowledge distillation (Section 4.1), and rule-denoising (Section 4.2). In these settings, we begin with a labeled dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$ with instances $x_i$ and categorical labels $y_i$ and an unlabelled dataset $U = \{(x_i)\}_{i=N+1}^{N+M}$ with only instances $x_i$. Note that, in the knowledge distillation setting, $U$ will be empty and in the case of rule-based denoising setting $N \ll M$. Our main proposal is to modify the objective in 1 so that there are instance-specific loss-mixing coefficients ($\lambda$s). Formally, we modify the loss function in 1 as follows:

$$
\mathcal{L}(\theta, \lambda) = \sum_{i} \left( \lambda_p \mathcal{L}_p(y_i, x_i|\theta) + \sum_{k=1}^{K} \lambda_{a_k,i} \mathcal{L}_{a_k,i} \right)
$$

(2)

Note that formulation in (2) is a generalization of Eq. (1), with an instance-specific value of mixing parameter $\lambda_p$ and $\lambda_{a_k,i}$ corresponding to the $i$th training instance $x_i$. Jointly optimizing the objective in Eq.(2) with respect to both sets of parameters $\theta, \lambda$ on the training dataset alone can lead to severe overfitting. To mitigate this risk, we instead attempt to solve the bi-level minimization problem in (3) using a meta-learning procedure:

$$
\underset{\lambda_p, \lambda_{a_1}, \lambda_{a_2}, \cdots, \lambda_{a_K}}{\text{argmin}} \mathcal{L}_{CE}\left( \underset{\theta}{\text{argmin}} \mathcal{L}(\theta, \lambda_p, \lambda_{a_1}, \lambda_{a_2}, \cdots, \lambda_{a_K}), \mathcal{V} \right)
$$

(3)
Algorithm 1 Algorithm for learning $\lambda$s via meta learning

Require: Training data $U$, Validation data $V$, $\theta^{(0)}$ model parameters initialization, $\tau$: Temperature, $\eta$: learning rate, $\eta_{\lambda}$: learning rate for updating $\lambda$.

Require: $LL_{CE}$ Primary Supervised Loss, $L_a$ auxiliary loss, max iterations $T$

1: Initialize model parameters $\theta^{(0)}$ and $\lambda_0^p, \lambda_0^a_1, \lambda_0^a_2, \ldots, \lambda_0^a_K$.

2: for $t \in \{0, \ldots, T\}$ do

3: Update $\theta^{t+1}$ by Eq. 9.

4: if $t \% L == 0$ then

5: $x_{\text{train}}, y_{\text{train}} \leftarrow \text{SampleMiniBatch}(U)$

6: $x_{\text{val}}, y_{\text{val}} \leftarrow \text{SampleMiniBatch}(V)$

7: Compute one step update for model parameters as function of $\lambda_{p_{t_{\frac{t}{L}}}}^j, \lambda_{a_{1_{t_{\frac{t}{L}}}}}^j, \lambda_{a_{2_{t_{\frac{t}{L}}}}}^j, \ldots, \lambda_{a_{K_{t_{\frac{t}{L}}}}}^j$ by Eq.(4).

8: Update $\lambda_{p_{t_{\frac{t}{L}}}}^j, \lambda_{a_{1_{t_{\frac{t}{L}}}}}^j, \lambda_{a_{2_{t_{\frac{t}{L}}}}}^j, \ldots, \lambda_{a_{K_{t_{\frac{t}{L}}}}}^j$ by Eq.(7) and Eq.(8).

9: end if

10: end for

By solving the inner level minimization, we wish to obtain model parameters $\theta$ that minimise the objective in Eq. (2). The outer minimization yields $\lambda$s such that the standard cross-entropy loss is minimised on the validation set $V$. This problem is a bi-level optimisation problem since model parameters $\theta$ are dependent on $\lambda$s and computation of $\lambda$s is dependent on model parameters $\theta$ as shown in Eq.(3).

Since the inner optimisation problem cannot be solved in a closed form in (3), we need to make some approximations in order to solve the optimization problem efficiently. We take an iterative approach, simultaneously updating the optimal model parameters $\theta$ and appropriate $\lambda$s in alternating steps as depicted in Figure 1. We first update the model parameters by sampling a mini-batch with $n$ instances from the training set, and simulating a one step look-ahead SGD update for the loss in Eq.(2) on model parameters $(\theta^t)$ as a function of $\lambda^t$, resulting in Eq.(4), with $L$ being a hyperparameter governing how often the lambda values are updated.

$$\theta^{t+1} = \theta^t - \eta \sum_{i=1}^{n} \nabla_{\theta^t} L_i(\theta^t, \lambda_{p_{t_i}}^j, \lambda_{a_{1_{t_i}}}^j, \lambda_{a_{2_{t_i}}}^j, \ldots, \lambda_{a_{K_{t_i}}}^j)$$  \hspace{1cm} (4)$$

Using the approximate model parameters obtained using the one step look-ahead SGD update, the
We update any (student) model having output logits as $a^{(s)} = \text{StudentModel}(x)$. This can be re-written as,

$$\nabla_{\lambda} L_{CE}(\theta^{t+1}, V) = \nabla_{\theta} L_{CE}(\theta^{t+1}, V). \nabla_{\lambda} \frac{1}{n} \sum_{i=1}^{n} \nabla_{\theta} L_i(\theta^t, a^{(s)}_{\lambda}, a^{(t)}_{\lambda_2, i}, \ldots, a^{(t)}_{\lambda_{K,i}})$$

To achieve this, we first define a knowledge distillation objective for the student as a KL-divergence between the predictions of the student and the teacher:

$$L_{KD} = \tau^2 KL(y^{(s)}, y^{(T)})$$

4 Two Application Scenarios for AMAL

In this Section, we present two application scenarios for AMAL, described in the previous Section 3, viz., knowledge distillation 4.1 and learning with limited supervision and rule-denoising in Subsection 4.2.

4.1 Knowledge distillation

Any (student) model having output logits as $a^{(s)} = \text{StudentModel}(x)$, is traditionally trained by optimizing a cross-entropy based loss $L_s$ defined as follows:

$$L_s = L_{CE}(\text{softmax}(a^{(s)}), y)$$

Let us say we have access to a pretrained teacher model (typically of higher learning capacity) which outputs the logits $a^{(T)} = \text{TeacherModel}(x)$. We can frame a teacher matching objective for the student as a KL-divergence between the predictions of the student and the teacher:

$$L_{KD} = \tau^2 KL(y^{(s)}, y^{(T)})$$
Then the training of the student model can be performed using both teacher matching objective and the traditional cross entropy loss as,

$$L_{\text{student}}(\theta, \lambda) = \sum_i \lambda_p L_s(y_i, x_i | \theta) + \lambda_a L_{KD}(y_i^{(S)}, y_i^{(T)})$$

(12)

This is the standard knowledge distillation loss, where typically a temperature parameter $\tau$ is used to control the softening of the KD loss in Eq. (11), and therefore we have $y^{(S)} = \text{softmax}\left(\frac{a^{(S)}}{\tau}\right)$ and $y^{(T)} = \text{softmax}\left(\frac{a^{(T)}}{\tau}\right)$. Clearly, here $L_p$ would be $L_s$ and $L_a$ would be $L_{KD}$. We present results of applying AMAL to adaptively mix these losses in section 5.1.

Distilling with multiple teachers Although, the Knowledge distillation (KD [17]) was introduced with only one teacher model and its corresponding loss, learning from multiple teachers has been shown to be useful [29, 37, 8]. Specifically we adapt our method to improve DGKD [37], where we perform knowledge distillation with multiple teachers and perform knowledge distillation with early stopped teachers [8]. Here, we redefine the Eq 12 as,

$$L_{\text{student}}(\theta, \lambda) = \sum_i \lambda_p L_s(y_i, x_i | \theta) + \sum_{k=1}^K \lambda_a L_{KD}(y_i^{(S)}, y_i^{(T_k)})$$

(13)

We can also adapt it to knowledge distillation from an early stopped teacher as presented in [8], by simply gathering multiple teacher checkpoints, and using them together as multiple teachers in knowledge distillation. We discuss the results and implication of this approach in Section 5.1.1.

4.2 Learning with limited supervision and rule-denoising

Several rule-denoising approaches [26, 4, 7, 32] encode multiple heuristics in the form of rules (or labeling functions) to weakly associate labels with instances. These weak labels are aggregated to determine the probability of the correct labels using generative models [7, 32] without requiring labeled data. In contrast, recent approaches [26, 21, 4, 35, 34] assume that a small labeled dataset is available in conjunction with the noisy rules. Motivated by the success of rule denoising approaches,
we propose adaptive loss mixing to leverage a small labeled set while being trained in a joint manner. We directly adopt the model and loss formulations from the most recent of these approaches [26], since it performs consistently better than the previous ones [25, 4, 35, 34] (see Section 5.2.

Our setting borrowed from SPEAR [26] is as follows: In addition to the setting described in Section 3, we also have access to $m$ rules or labelling functions (LF) $lf_1$ to $lf_m$. We modify $D$ to be $D' = \{(x_i, y_i, l_i)\}_{i=1}^N$ and $U$ to be $U' = \{(x_i, l_i)\}_{i=N+M}^N$, where $l_i = (l_{i1}, l_{i2}, \ldots, l_{im})$ is a boolean vector with $l_{ij} = 1$ if the corresponding $j^{th}$ LF is activated on example $x_i$ and $l_{ij} = 0$ otherwise. Exactly as per [26], our model is a blend of the feature-based classification model $f_\theta(x)$ and the rule-based model $P_\phi(l, y)$. We have two types of supervision in our joint objective. First, we have access to $y$ for the labeled instances $D'$ and to $l_{ij}$ for all the labeled as well as unlabeled instances $D' \cup U'$. We measure the loss of $P_\theta$ and $P_\phi$ on the small labeled set $D'$ through standard cross-entropy. Second, we model interaction between $P_\theta$ and $P_\phi$ on the union of labeled and unlabeled sets. Intuitively, the rule denoising model $P_\phi$ learns with respect to the clean labeled set $D'$ and simultaneously provides labels over $U$ that can be used to train the feature model $P_\theta(y|x)$. We want both the models to agree in their predictions over the union $D' \cup U'$. (Please refer to Supplementary Section E for details about individual loss components.)

5 Results

In this section, we present results for the two application scenarios for Amal as outlined in Section 4.

5.1 Results with Knowledge Distillation

In this section, we report a range of experimental results from the knowledge distillation (KD) scenario as described in Section 4.1. We performed a range of experiments comparing Amal against several SOTA knowledge distillation approaches on several real-world datasets, with a special focus on those settings wherein we found the gap between the teacher and student models to be large.

Datasets The datasets in our experiments include CIFAR100 [23], Stanford Cars [22] and FGVC-Aircraft [27]; characteristics of the datasets are summarized in Table 3 in the Appendix. For the CIFAR datasets we used the standard RGB images of size $32 \times 32$, whereas for the other datasets we used RGB images of size $96 \times 96$. Model architecture and experimental setup

We explored two families of models, viz., (i) Wide Residual Networks (WRN-16-1, WRN-16-3, WRN-16-4, WRN-16-6, WRN-16-8) [43], and (ii) ResNet (8, 20, 32, 56, 110) models [15] to show the effectiveness of our method across the different model families. We also perform a distillation on ResNet8 with WRN-16-3, WRN-16-4, WRN-16-6 and WRN-16-8 as teachers to show the effect of our technique in the cross-model distillation.

For datasets without pre-specified validation sets, we split the original training set into new train (90%) and validation sets (10%) (see Table 3 for details). Training consisted of SGD optimization with an initial learning rate of 0.05, momentum of 0.9, and weight decay of 5e-4. We divided the learning rate by 0.1 on epochs 150, 180 and 210 and trained for a total of 240 epochs.

Effect of optimal $\lambda$ on Knowledge Distillation

In the first experiment, we examine effective transfer of learned knowledge from various teachers to a student model which has fewer parameters. We compares test accuracies obtained with KD, Amal, TAKD [29] and DGKD [37] and SSKD [42]. TAKD takes taking multiple KD training hops, with each step reducing the model complexity from teacher to student by a small amount. DGKD introduces all the intermediate teachers in a single KD training step and tries to optimise a loss similar to Eq. (13), although only using a single $\lambda$, across all training instances, for each teacher. In
addition, stochastic DGKD was proposed where a subset of teachers is introduced at each training step, determined by a binomial (hyperparameter) variable.

**Additional experimental setup** We perform KD with ResNet (20, 32, 56, 110) as teacher and ResNet8 as student models on CIFAR100, Wide Residual Networks (WRN-16-3, WRN-16-4, WRN-16-6, WRN-16-8) as teacher and WRN-16-1 as student models on Stanford Cars and with Wide Residual Networks (WRN-16-3, WRN-16-4, WRN-16-6, WRN-16-8) as teachers and Resnet8 as student on FGVC-Aircraft. For TAKD and DGKD we use ResNet14 for CIFAR100 and WRN-16-2 for Stanford Cars and FGVC-Aircraft as teaching assistant models. In all our knowledge distillation experiments we use temperature $\tau = 4$ and $\lambda_a = 0.9$ (weights associated with KD loss) except in case of AMAL. For DGKD we use set the binomial variable to be 0.75, best reported in the paper.

| Student Model → | WRN-16-1 | DenseNet-40-12 |
|-----------------|----------|---------------|
| Method ↓        |          |               |
| KD              | 66.47    | 76.57         |
| AMAL with early stopped teachers | 67.69 | 76.79 |
| DGKD            | 67.88    | 76.86         |
| DGKD+ AMAL      | 68.58    | 77.66         |

Table 1: Learning from early-stopped teacher: AMAL was trained with a range of teacher checkpoints (early stopping) in the multi-teacher setting. As obtained from AMAL’s meta learning process helps identify a good teacher. Similarly, AMAL beats SOTA DGKD while learning from different teacher model. Here, the teacher model was WRN-16-8, and CIFAR-100 was used as dataset with knowledge distillation from the teacher model. However, this also transfer errors from the higher level to the lower level models [37]. It is important to note that AMAL doesn’t require any additional intermediate model to be trained like TAKD and DGKD and therefore has a lesser memory footprint and training time.

### 5.1.1 Knowledge Distillation with multiple teachers

In the multi teacher setup we use WRN-16-8 as a teacher model and perform knowledge distillation on DenseNet-40-12 [19] and WRN-16-1. Here we consider two settings to address the teacher student gap viz. performing knowledge distillation with a teacher stopped at some intermediate stage [8] and learning from multiple teachers of different learning capacities (DGKD) [37].

The former approach raises a new challenge: how do we find an appropriate stopping point for the teacher, without having to train a large number of student models corresponding to teacher stopping points? We adapt our multi-teacher setting (Section 4.1) to solve this as follows: we train a single student model, with *multiple teacher models*, each stopped at intermediate points of the teacher training process. For this experiment, we trained a teacher model (WRN-16-8) on CIFAR-100 at the 80th, 160th, 200th epochs as well as the final model, and trained two different student architectures

---

We define model compression ratio as (no. of learnable parameters in teacher model)/(no. of learnable parameters in student model); higher is better
using multi-teacher Amal. Table 1 shows that Amal with early-stopped teachers consistently outperforms the standard KD as presented in [8] with one step knowledge distillation.

DGKD introduced a stochastic variant where only a subset of teachers is introduced at each training step, determined by a binomial (hyperparameter) variable. This presents a need to control the contribution of each of the teachers in a systematic manner; therefore, we adapt our multi-teacher setting (Section 4.1) with WRN-16-3 as the additional teacher model. We present results in Table 1, and observe that Amal yields performance gains over simple DGKD.

5.1.2 How does Amal work in KD?

We also took a closer look at the above hypothesis: that early stopping of the teacher helps by creating ‘simpler’, easy-to-mimic teacher outputs. We used two WRN-16-8 teachers—early stopped, and final, on CIFAR-100—to train WRN-16-1 Student models corresponding to each teacher. We then examined the change in lambda value (i.e., teacher weight) across the two student models, as a function of change in teacher confidence, i.e., the probability output by the teacher to the ground-truth label in training data. Figure 3 shows a very interesting result: as the teacher increased in confidence, the $\lambda$ values chosen by Amal decreased whereas if teacher confidence was largely unchanged, the $\lambda$ values were increased. This suggests that the primary challenge in the final teacher model is overfitting, with even noisy or ambiguous labels being confidently predicted by the teacher. Our validation-based objective, however, is able to identify those instances that the teacher is overconfident on (as they do not improve validation accuracy), and able to downweight the teacher in those instances.

In Supplementary Section C, we report the use of the validation data in different forms to strengthen baseline, but all those efforts either weakened or did not add any value to the existing baselines.

5.2 Amal with limited supervision and rule-denoising

In this section, we report our experimental results for the scenario of limited supervision combined with weak supervision from labeling functions (also referred to as semi-supervised data programming [26]), as summarized in Section 4.2. Datasets We used three datasets in our experiments, namely, YouTube, SMS and IMDB. YouTube [1] is a spam classification task over YouTube comments; SMS [3] is a binary spam classification containing 5574 documents; IMDB is a movie plot genre binary classification dataset.

In Table 2, we compare our approach with the following approaches: (1) Only-$\mathcal{L}$: We train the classifier $P_{\theta}(y|x)$ only on the labeled data. To ensure fair comparison, we use the same classifier
model for different datasets as mentioned in [26]. We choose this as a baseline and report gains over it.

(2) **L2R** [34]: This is an online reweighting algorithm that leverages validation set to assign weights to examples based on gradient directions. It learns to re-weigh weak labels from domain specific rules and learn instance-specific weights via meta-learning. However, weights are not learnt to mix losses.

(3) **Imply Loss** [4]: This is a rule-exemplar approach that jointly trains a rule denoising network and leverages exemplar-based supervision for learning instance-specific and rule-specific weights. In addition, it also learns a classification model with a soft implication loss in a joint objective. 

(4) **SPEAR** [26]: Finally, we compare with another rule-denoising approach that uses same objective as **Amal** and is trained on both feature-based classifier and rule-classifier using a small labeled set. **Amal** with all $\lambda$s fixed to 1 (and not trainable) corresponds to SPEAR [26].

Our approach outperforms both rule-based and re-weighing approaches on all datasets. L2R and SPEAR perform worse than the baseline method (only-L) on SMS and IMDB dataset whereas Imply-Loss is marginally better on SMS. All approaches achieve better performance over the baseline method on YouTube dataset. However, **Amal** consistently reports highest gains. Recall that SPEAR has the same objective as **Amal** but without trainable $\lambda$s and all $\lambda$s fixed to 1. **Amal** tries to identify instance-wise weighted combination of loss components so that the trained feature classification model performs better. Instance wise mixing is useful to identify the loss component from which a data point could be learned better and use of fixed weights prevents from understanding nuance of each data point.

### Table 2: Performance of our **Amal** approach with rule-based approaches (Imply Loss [4] and SPEAR [26]) and point-wise re-weighing method (L2R [34]). **Amal** with fixed $\lambda = 1$ corresponds to SPEAR [26]. All numbers reported are gains over the baseline method (Only-L). All results are averaged over 5 random seed runs having different $L$ and $U$ set in each run. Numbers in brackets ‘()’ represent standard deviation of the original score.

|                | SMS      | IMDB     | YouTube |
|----------------|----------|----------|---------|
| Only-L         | 91.45 (1.3) | 77.35 (1.5) | 89.60 (2.9) |
| Imply Loss     | +0.25 (1) | -1.47 (1.8) | +2.70 (0.8) |
| L2R            | -0.20 (1.3) | -2.18 (1.4) | +3.40 (1.2) |
| SPEAR          | -0.76 (1.4) | -0.04 (1) | +4 (1) |
| **Amal**       | +1.53 (0.9) | +1.67 (1.6) | +4.70 (0.8) |

### 6 Conclusion

In this paper we present two setting viz. rule-denoising setting with limited supervision and knowledge distillation (KD), where Adaptive Loss Mixing is useful. We present **Amal** which via adaptive loss mixing extracts useful information from the limited supervision to prevent degradation of model learnt due to the presence of noisy rule. In knowledge distillation (KD) setting it titrates the teacher knowledge and ground truth label information through an instance-specific combination of teacher-matching and ground supervision objectives to learn student models that are more accurate. Our iterative approach is pivoted on solving a bi-level optimization problem in which the instance weights are learnt to minimize the CE loss on a held-out validation set whereas the model parameters are themselves estimated to minimize the weight-combined loss on the training dataset. Through extensive experiments on real-world datasets, we present how **Amal** yields accuracy improvement and better generalization on a range of datasets in both the settings.
References

[1] Túlio C Alberto, Johannes V Lochter, and Tiago A Almeida. Tubespam: Comment spam filtering on youtube. In 2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA), pages 138–143. IEEE, 2015.

[2] Görkem Algan and Ilkay Ulusoy. Meta soft label generation for noisy labels. In 2020 25th International Conference on Pattern Recognition (ICPR), pages 7142–7148. IEEE, 2021.

[3] Tiago A Almeida, José María G Hidalgo, and Akebo Yamakami. Contributions to the study of sms spam filtering: new collection and results. In Proceedings of the 11th ACM symposium on Document engineering, pages 259–262, 2011.

[4] Abhijeet Awasthi, Sabyasachi Ghosh, Rasna Goyal, and Sunita Sarawagi. Learning from rules generalizing labeled exemplars. In 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26–30, 2020. OpenReview.net, 2020.

[5] Yoshua Bengio. Gradient-based optimization of hyperparameters. Neural computation, 12(8):1889–1900, 2000.

[6] Thibault Castells, Philippe Weinzaepfel, and Jerome Revaud. Superloss: A generic loss for robust curriculum learning. Advances in Neural Information Processing Systems, 33:4308–4319, 2020.

[7] Oishik Chatterjee, Ganesh Ramakrishnan, and Sunita Sarawagi. Robust data programming with precision-guided labeling functions. In AAAI, 2020.

[8] Jang Hyun Cho and Bharath Hariharan. On the efficacy of knowledge distillation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 4794–4802, 2019.

[9] Justin Domke. Generic methods for optimization-based modeling. In Artificial Intelligence and Statistics, pages 318–326. PMLR, 2012.

[10] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In International conference on machine learning, pages 1126–1135. PMLR, 2017.

[11] Tommaso Furlanello, Zachary Lipton, Michael Tschannen, Laurent Itti, and Anima Anandkumar. Born again neural networks. In International Conference on Machine Learning, pages 1607–1616. PMLR, 2018.

[12] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. International Journal of Computer Vision, 129(6):1789–1819, 2021.

[13] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Doina Precup and Yee Whye Teh, editors, Proceedings of the 34th International Conference on Machine Learning, volume 70 of Proceedings of Machine Learning Research, pages 1321–1330. PMLR, 06–11 Aug 2017.

[14] Sangchul Hahn and Heeyoul Choi. Self-knowledge distillation in natural language processing. CoRR, abs/1908.01851, 2019.
[15] 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.

[16] Byeongho Heo, Minsik Lee, Sangdoo Yun, and Jin Young Choi. Knowledge distillation with adversarial samples supporting decision boundary. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 3771–3778, 2019.

[17] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network, 2015.

[18] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. arXiv preprint arXiv:2004.05439, 2020.

[19] Gao Huang, Zhuang Liu, and Kilian Q. Weinberger. Densely connected convolutional networks. CoRR, abs/1608.06993, 2016.

[20] Simon Jenni and Paolo Favaro. Deep bilevel learning. In Proceedings of the European conference on computer vision (ECCV), pages 618–633, 2018.

[21] Giannis Karamanolakis, Subhabrata Mukherjee, Guoqing Zheng, and Ahmed Hassan. Self-training with weak supervision. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 845–863, 2021.

[22] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In 4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13), Sydney, Australia, 2013.

[23] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[24] Yuang Liu, Wei Zhang, and Jun Wang. Adaptive multi-teacher multi-level knowledge distillation. Neurocomputing, 415:106–113, 2020.

[25] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In International conference on machine learning, pages 2113–2122. PMLR, 2015.

[26] Ayush Maheshwari, Oishik Chatterjee, Krishnateja Killamsetty, Ganesh Ramakrishnan, and Rishabh Iyer. Semi-supervised data programming with subset selection. In Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021, pages 4640–4651, Online, August 2021. Association for Computational Linguistics.

[27] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[28] Aditya K Menon, Ankit Singh Rawat, Sashank Reddi, Seungyeon Kim, and Sanjiv Kumar. A statistical perspective on distillation. In Marina Meila and Tong Zhang, editors, Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, pages 7632–7642. PMLR, 18–24 Jul 2021.

[29] Seyed-Iman Mirzadeh, Mehrdad Farajtabar, Ang Li, Nir Levine, Akihiro Matsukawa, and Hassan Ghasemzadeh. Improved knowledge distillation via teacher assistant, 2019.
[30] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

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

[32] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access, 2017.

[33] Alexander J Ratner, Christopher M De Sa, Sen Wu, Daniel Selsam, and Christopher Ré. Data programming: Creating large training sets, quickly. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[34] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In *International Conference on Machine Learning*, pages 4334–4343, 2018.

[35] Wendi Ren, Yinghao Li, Hanting Su, David Kartchner, Cassie Mitchell, and Chao Zhang. Denoising multi-source weak supervision for neural text classification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3739–3754, 2020.

[36] Shreyas Saxena, Oncel Tuzel, and Dennis DeCoste. Data parameters: A new family of parameters for learning a differentiable curriculum. *NeurIPS*, 2019.

[37] Wonchul Son, Jaemin Na, Junyong Choi, and Wonjun Hwang. Densely guided knowledge distillation using multiple teacher assistants, 2021.

[38] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[39] Nidhi Vyas, Shreyas Saxena, and Thomas Voice. Learning soft labels via meta learning, 2020.

[40] Xiaojie Wang, Rui Zhang, Yu Sun, and Jianzhong Qi. Kdgan: Knowledge distillation with generative adversarial networks. In *NeurIPS*, pages 783–794, 2018.

[41] Yunhe Wang, Chang Xu, Chao Xu, and Dacheng Tao. Adversarial learning of portable student networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[42] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *European Conference on Computer Vision*, pages 588–604. Springer, 2020.

[43] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016.

[44] Haoran Zhao, Xin Sun, Junyu Dong, Zihe Dong, and Qiong Li. Knowledge distillation via instance-level sequence learning. *Knowledge-Based Systems*, 233, 2021.

[45] Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. Robust curriculum learning: From clean label detection to noisy label self-correction. In *International Conference on Learning Representations*, 2020.
[46] Tianyi Zhou, Shengjie Wang, and Jeff A Bilmes. Curriculum learning by dynamic instance hardness. *Advances in Neural Information Processing Systems, 33*, 2020.
A Synthetic experiments

We explore the performance and characteristics of our approach in synthetic data settings, to derive insight into the mechanisms by which AMAL.

**Synthetic data generation:** We use the standard sklearn.datasets package \[31\] to generate synthetic data with 14 features and 20 classes. The generated data has 8100 training points, 900 points in validation set and 1000 points in the test set. We randomly flip labels of 10% of the training data points to introduce noise. In Figure 4a shows a t-sne projection of a synthetically generated dataset (generation details in the main body of the paper); as can be seen, the 20 classes have some spatial cohesiveness but also significant overlap, making it a nontrivial learning task to classify instances into their respective labels.

(a) t-SNE plot of the synthetic data used.

(b) Teacher confidence on ground truth for noise vs clean points

(c) Relationship between teacher confidence, label noise, and learned $\lambda$s with mean and SEM error bars over 15 runs. As elaborated in Section A.1, the teacher assigns relatively lower probabilities to noisy labels, and the meta-learning process in AMAL assigns higher $\lambda$ to noisy data points.

(d) Learning trajectories: The figure shows generalization of test set accuracies (along with SEM bars) for vanilla KD, label-trained model, and AMAL, as a function of training epochs. All curves are averages over 50 runs.

Figure 4: Synthetic experiment details and analysis
A.1 $\lambda$s can counter label noise

Through this experimental setup, we examine the relationship between the teacher confidence, the label noise, and the learned $\lambda$ values of data points. We grouped training data points into buckets, based on the probability assigned by the teacher to its ground-truth label, and computed the average learned $\lambda$ per bucket; this averaging process was done separately for the instances with and without injected label noise. Finally, the experiment was repeated with 50 random seeds, and the averages as well as standard error of the mean (SEM) bars across those 50 runs are presented in Figure 4c. We see two emerging patterns: Firstly, that as expected, the teacher assigns overall lower probabilities to the noisy labels in comparison to the other data points. Secondly, we see a trend wherein $\lambda$s learned for noisy labels are overall higher than for clean labels. This is because the meta-learning process learns that the ground-truth labels on those data points contribute poorly to generalization on the validation data, whereas, for such instances, the teacher probability is more informative than the ground-truth label. In Figure 4b shows the distribution of converged teacher probabilities for noisy and clean labels, showing, as expected, that the clean label data have a broad distribution of teacher probabilities for ground-truth label, with a right skew, whereas the noisy labels have a sharply leftward skew (i.e., very low teacher probabilities for ground-truth labels). This nicely complements the data in the left panel, showing that for noisy labels, and in general for less-confident teacher signals, the learned $\lambda$ values are higher, indicating that the teacher has more informative content (e.g., instance hardness, label ambiguity) than the ground-truth label in those scenarios.

In a second experiment, we further examined the contributions of the meta-learning procedure to test-set generalization. In Figure 4d, we present the test set accuracy as well as standard error of the mean (SEM) bars for various models as a function of training data epoch. Each curve is the average of 50 random synthetic data simulation-based training runs. We see that compared to the student model trained on the ground-truth data (i.e. the label-trained model), as well as the student trained with fixed $\lambda = 0.9$ (i.e. vanilla KD), Amal learns faster, and converges to a higher test accuracy, driven by the adaptive loss mixing approach (c.f. Section 3).

B Additional details for Knowledge Distillation Experiments

B.1 Dataset Details

Table 3 provides details of the various real-world datasets used in our experiments, and the partitioning of these datasets into train, validation (needed in our meta-learning procedure), and test data subsets. Wherever available, the existing splits provided by the source data were used; in other cases, 10% of the training data was partitioned off for use as validation data.

| Dataset       | #Classes | #Instances | #Train | #Validation | #Test  |
|---------------|----------|------------|--------|-------------|--------|
| CIFAR100      | 100      | 60000      | 45000  | 5000        | 10000  |
| Stanford Cars | 196      | 16185      | 7330   | 814         | 8,041  |
| FGVC-Aircraft | 102      | 10200      | 6120   | 680         | 3400   |

Table 3: Number of classes, Number of instances in Train, Validation and Test splits in the different datasets
B.2 Additional Experimental Setup

We ran experiments using an SGD optimizer with an initial learning rate of 0.05, the momentum of 0.9, and a weight decay of 5e-4. We divided the learning rate by 0.1 on epochs 150, 180 and 210 and trained for a total of 240 epochs. In all our knowledge distillation experiments we use temperature $\tau = 4$ and $\lambda_a = 0.9$ (weights associated with KD loss) except in case of Amal. We update the $\lambda$s every 10 epochs, therefore $L = 10$. We ran all experiments on a single A100 GPU. The code for the experiments performed is available in https://anonymous.4open.science/r/DistillCodeGit-E68A.

C Additional experiments

C.1 $\lambda$s for better generalisation

We now explore the self-distillation setting [11, 14], where the teacher and student models have identical architectures, and the goal is to train a student with better generalization accuracy than the teacher, through the use of the distillation loss for regularization. In Table 4, we present the results of self-distillation experiment on the two datasets viz., CIFAR100 and the FGVC-Aircraft datasets on a WRN-16-8 model.

The first row presents training results based on the standard cross-entropy loss and the second row presents the results in standard knowledge distillation setup. Third row of table presents results with Platt-scaling [13], which rescales test outputs based the validation set. Here we apply Platt-scaling over the knowledge distillation setup presented in the second row. The final row shows that adaptive mixing via Amal further improves upon self-distillation, thereby making it a potentially valuable tool in the traditional supervised learning setting in addition to teacher-student transfer for training smaller, more efficient student models. Poor platt scaling results indicate that use validation data doesn’t always strengthen the baselines.

| Dataset $\rightarrow$ | CIFAR100 | FGVC-Aircraft |
|-----------------------|----------|---------------|
| Method ↓              |          |               |
| CE loss alone         | 77.52    | 63.75         |
| Self-distillation     | 79.12    | 66.93         |
| Self-distillation + Platt-Scaling | 79.01 | 66.12 |
| Self-distillation + Amal | 79.41   | 67.44         |

Table 4: Self-distillation: Compared to traditional supervised learning (CE loss alone) self-distillation show higher accuracy, as reported by earlier work. We show that Amal is easily applicable to the self-distillation setting, and provides additional gains in performance as oppose to other ways of using validation set such as Platt-scaling.
C.2 Connection to a Coreset

| Method                                      | Test Accuracy |
|---------------------------------------------|---------------|
| Random                                      | 44.92         |
| Sampled according to $\lambda_p^2 + \lambda_a^2$ | 45.5          |
| Sampled according to $|\lambda_p - \lambda_a|$ | 46.28         |
| Sampled according to $\frac{\lambda_p}{\lambda_a}$ | 46.31         |

Table 5: Test accuracies obtained after training with 20% subset obtained using various strategies on WRN-16-1 model. We perform training with only CE loss.

Since, Amal controls the contribution of each of the instance in training the model by weighting each of the points loss functions, we try to understand the significance of the weights associated with those points with a coreset based experiment. Coreset selection has become popular in recent times where a subset of training points are used to train a model from scratch. Based on the final $\lambda_p$(weighted associated with the CE loss) and $\lambda_a$(weighted associated with the KD loss) values while training WRN-16-1 model using WRN-16-8 as teacher model, we derive a probability of selection $p_i$ for each point $i$ in the training set as,

1. $p_i \propto \lambda_p^2 + \lambda_a^2$, here we pick points with maximum weights as they would contribute maximum to the model training
2. $p_i \propto |\lambda_p - \lambda_a|$, here we pick points that should be preferably learnt with only one of the losses
3. $p_i \propto \frac{\lambda_p}{\lambda_a}$, here we pick points that should be preferably learnt with only KD loss

In table 5 we present the test accuracies obtained on training WRN-16-1 with the coresets obtained when sampled using the corresponding probabilities. We also present the result of training the same model with randomly (sampled with uniform distribution) obtained subset. We train with subsets of 20% size of the original training data and train with only the CE loss. Clearly, the points that have higher weights have maximum information. More, specifically the points that require a teacher model’s assistance and cannot be learned using the ground truth seem to have the most information and therefore coreset formed using 3 performs the best.

C.3 Ablation study with temperature $\tau$ and Frequency of $\lambda$ updates $L$

Figure 5a shows change in test accuracies achieved by Amal as we vary $L$ parameter which controls how often lambdas are updated. Here we present results for $L = 5, 10, 20$. Although, there isn’t significant drop in test accuracy when we update $\lambda$s every 10 epochs instead of 5, there is a slight drop when we update $\lambda$s every 20 epoch. $L$ controls the trade-off between the time spent on updating lambdas verse improving the model performance. Here we find the $L = 10$ is best as it saves time by not updating $\lambda$s too often while also achieving comparable performance as of $L = 5$. Figure 5b we study the effect on the test accuracies achieved by Amal as we vary temperature $\tau$ parameter used to control the softening of the KD loss. Clearly, across different $\tau$ values Amal outperforms the standard KD and therefore Amal’s performance gains are not effected by change in $\tau$. 

18
Figure 5: Effect on test accuracies achieved by Amal as we vary temperature \( \tau \) and \( L \) while performing knowledge distillation on CIFAR100 dataset with WRN-16-8 as the teacher and WRN-16-1 as the student model.

D Additional details for Limited supervision and rule-denoising Experiments

Here, we provide additional details about the datasets used in rule-denoising experiments. We used three dataset in our experiments, namely, YouTube, SMS and IMDB. In addition to the features, we have access to \( m \) rules of labelling functions (LFs). In Table 6, we provide statistics of these LFs as well as size of labeled and unlabeled set. We borrow LFs from the SPEAR and our experimental setting such as batch size, learning rates are same as SPEAR \cite{26} to ensure fair comparison.

| Dataset  | \( |L| \) | \( |U| \) | #Rules/LFs | Precision | %Cover | %Conflicts | \(|\text{Test}| \) |
|----------|---------|---------|-------------|------------|--------|------------|--------|
| YouTube  | 100     | 1586    | 10          | 75         | 86.6   | 30.1       | 250    |
| SMS      | 69      | 4502    | 73          | 97.3       | 39.3   | 0.67       | 500    |
| IMDB     | 284     | 852     | 25          | 80         | 48.6   | 11.1       | 500    |

Table 6: Statistics of datasets and their rules/LFs. Precision refers to micro precision of rules. \%Cover is the fraction of instances in \( U \) covered by at least one LF. \%Conflict denotes the fraction of instance covered by conflicting rules among all instances. Size of Validation set is equal to \(|L|\).

E Rule-denoising objective

Here, we describe the individual loss components borrowed from SPEAR \cite{26}. Further, we define our adaptive mixing loss which forms our overall objective function.

**First Component (L1):** Standard cross-entropy loss on \( D \) for the model \( P^f_{\theta} : L_{CE} \left( P^f_{\theta}(y|x_i), y_i \right) = -\log \left( P^f_{\theta}(y = y_i|x_i) \right) \)

**Second Component (L2):** The second component \( LL_s(\phi|D) \) is the (supervised) negative log
likelihood loss on the labeled set $\mathcal{D}$: $LL_s(\phi | \mathcal{D}) = -\sum_{i=1}^{N} \log P_{\phi}(l_i, y_i)$

**Third Component (L3):** The third component $L_{CE} \left( P_{\phi}^f (y | x_i), g(l_i) \right)$ is the cross-entropy of the classification model using the hypothesised labels from CAGE [7] on $U'$. CAGE is a generative graphical model that assigns parameter $\phi_j$ for each rule and share it across $y$. (Please refer Appendix for the complete formulation.) Using the LF-based graphical model $P_{\phi}(l_i, y)$ as: $g(l_i) = \arg\max_y P_{\phi}(l_i, y)$

**Fourth Component (L4):** The fourth component $KL( P_{\phi}^f (y | x_i), P_{\phi}(y | l_i) )$ is the Kullback-Leibler (KL) divergence between the predictions of both the models, viz., feature-based model $f_\theta$ and the rule-based graphical model $P_{\phi}$ summed over every example $x_i \in U' \cup D'$. We try and make the models agree in their predictions over the union of the labeled and unlabeled datasets.

E.1 Adaptive Loss Mixing for limited supervision and rule-denoising

In our joint objective, feature based classification model $f_\theta(x)$ while second component (L2) trains the rule-based model. L1 and L2 works on $\mathcal{D}'$, L3 works on $U'$ and L4 component works on $\mathcal{D}' \cup U'$. Therefore, joint objective can be rewritten using instance wise weights $\lambda_{p_i}$ and $\lambda_{a_{1,i}}$, introduced in Eq. 2 as,

$$
\mathcal{L} = \begin{cases} 
\lambda_{p_i} \star L_{CE} \left( P_{\phi}^f (y | x_i), y_i \right) + \lambda_{a_{1,i}} \star KL( P_{\phi}^f (y | x_i), P_{\phi}(y | l_i) ) & \text{if } (x_i, y_i, l_i) \in \mathcal{D}' \\
+ LL_s(\phi | x_i, y_i), & \\
\lambda_{p_i} \star L_{CE} \left( P_{\phi}^f (y | x_i), g(l_i) \right) + \lambda_{a_{1,i}} \star KL( P_{\phi}^f (y | x_i), P_{\phi}(y | l_i) ) & \text{if } (x_i, y_i, 1_i) \in U'
\end{cases}
$$

Thus our primary objective changes based on whether the point belongs to the labelled set $\mathcal{D}'$ or the unlabelled set $U'$. 

20