Learning with Limited Labels via Momentum Damped & Differentially Weighted Optimization

Rishabh Mehrotra
Spotify Research
rishabhm@spotify.com

Ashish Gupta
Walmart Labs
ashish.gupta@walmartlabs.com

ABSTRACT

As deep learning-based models are deployed more widely in search & recommender systems, system designers often face the issue of gathering large amounts of well-annotated data to train such neural models. While most user-centric systems rely on interaction signals as implicit feedback to train models, such signals are often weak proxies of user satisfaction, as compared to (say) explicit judgments from users, which are prohibitively expensive to collect. In this paper, we consider the task of learning from limited labeled data, wherein we aim at jointly leveraging strong supervision data (e.g. explicit judgments) along with weak supervision data (e.g. implicit feedback or labels from the related task) to train neural models.

We present data mixing strategies based on submodular subset selection, and additionally, propose adaptive optimization techniques to enable the model to differentiate between a strong label data point and a weak supervision data point. Finally, we present two different case-studies (i) user satisfaction prediction with music recommendation and (ii) question-based video comprehension and demonstrate that the proposed adaptive learning strategies are better at learning from limited labels. Our techniques and findings provide practitioners with ways of leveraging external labeled data.

ACM Reference Format:
Rishabh Mehrotra and Ashish Gupta. 2020. Learning with Limited Labels via Momentum Damped & Differentially Weighted Optimization. In *Proceedings of the 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '20), August 23–27, 2020, Virtual Event, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3394486.3403394

1 INTRODUCTION

While neural models often provide state-of-the-art performances in search and recommendation tasks, they rely heavily on large tagged corpora. Manual collection and curation of such datasets at scale is often infeasible due to time and expense constraints. Moreover, given the dynamicity of real world data and modeling goals, one often needs to relabel static hand-labeled training sets. A number of popular approaches for addressing this labeled data scarcity have recently come up, including leveraging *weak supervision*[24], which are higher-level approaches for (i) labeling training data and (ii) using them in training models at scale. Examples of such data labeling approaches include distant or heuristic supervision[18], or constraints; data augmentation strategies to express class invariances; and introduction to other forms of prior knowledge [25]. Techniques for leveraging such weak supervision signals include fidelity weighted optimization [3] and pre-training models. The overarching goal of both lines of work is to enable large scale training of neural models using weak supervision signals.

Considering the specific case of search and recommender systems, models are trained on implicit signals of user satisfaction, which serve as proxies of true user satisfaction [14, 17]. However, it is not uncommon to collect labeled data via crowd-sourcing or via in-app surveys, wherein users provide explicit judgments about their experience and satisfaction [15]. Both these forms of data: large scale implicit interaction data and relatively small-scaled explicit user judged data have complimentary characteristics. While implicit data is abundantly available, it is often noisy and heavily dependent on interpretation by system designers – it may not accurately reflect true user satisfaction. On the other hand, explicit data from users is reliable and a stronger signal but is hard to obtain, at scale.

In this paper, we consider the problem of learning from limited data. Specifically, we assume access to a small strongly labeled dataset (e.g. via explicit human judgments) and a larger weak supervision dataset (e.g. from implicit feedback or data from other tasks) and consider the task of training models by jointly leveraging both these datasets. We propose a number of data mixing strategies wherein a new dataset is created by mixing the strong and weak supervision data via different techniques including similarity-based and interleaving approaches. Additionally, we cast the problem of leveraging weak supervision data in the form of subset selection, and propose a submodular subset selection strategy [12, 16] which jointly leverages notions of *representativeness* and *informativeness* of the weak supervision instances.

Going beyond simple data mixing strategies, we then investigate adaptive optimization methods to train neural models that allow us to control the extent to which we make use of weak supervision signals. The most common way to optimize deep neural models is to employ moment based stochastic gradient descent optimization algorithms (e.g. Adam, Adagrad) to perform optimization of stochastic objective functions[6, 9]. We identify a key issue with such momentum-based gradient descent approaches when learning on weak supervision data. The unrelated training examples might steer the gradient in the wrong direction, leading to suboptimal training. To this end, we propose two adaptive learning strategies which consider the importance of each data point and adapt the learning strategy based on the training sample. Our first adaptive learning strategy is a differential weighting scheme wherein we

ACM ISBN 978-1-4503-7998-4/20/08…$15.00
https://doi.org/10.1145/3394486.3403394

weight the gradient update adaptively based on the importance of weak supervision datapoint. The second proposed strategy is a momentum damped variant of Adam, a first-order stochastic optimization technique, which dampens momentum updates of the optimizer based on label data fidelity, and is thus adept at handling irrelevant data while training on weak supervision signals.

We compare and contrast the different strategies to learn from limited labeled data on two different case studies: (i) user satisfaction prediction for music recommendation and (ii) question-based video comprehension. The first case study leverages implicit feedback signals as weak supervision data, and in-app survey responses by users as ground-truth data, whereas the second video QA case study works with distant supervision signals as weak supervision data, and human judgments for questions and answers from videos as strong data. Our findings suggest that mixing weak and strong supervision data is often helpful, with significant improvements in several metrics across both use-cases. Furthermore, adaptive training strategies based on momentum damping outperform all data mixing strategies.

2 RELATED WORK

Weak Supervision: The problem of handling noise has been the focus of much attention in machine learning and most inductive learning algorithms have a mechanism for handling noise in the training labels. Removing noise from the data before hypothesis formation has the advantage that noisy examples do not influence hypothesis construction. Learning from weak data sometimes aims at encoding various forms of domain expertise or cheaper supervision from lay annotators. Alternatively, some noise cleansing methods have been proposed to remove or correct mislabeled samples[2]. Some studies are showing that weak or noisy labels can be leveraged by modifying the loss function[21, 28] or changing the update rule to avoid imperfections of the noisy data[4, 13]. Further, a growing number of tools enable labeling of weak supervision data at scale [25].

Momentum based learning: The convergence of momentum methods has been studied extensively, both theoretically and empirically [8, 29, 31]. But there exist some flaws in existing methods hence, the following works motivate successful momentum schemes. [26] explored the effect of momentum on the optimization of neural networks and introduced the momentum view of Nesterov's accelerated gradient. They focused on producing good momentum schedules during optimization to adapt to ill-conditioned curvature. Adaptive gradient methods have been introduced to deal with the ill-conditioned curvature that we often observe in deep learning [9, 27]. These methods typically approximate the local curvature of the objective to adapt to the geometry of the data. Our work extends the use of momentum updating schemes conditioned on weak and strong supervision.

3 LEARNING FROM LIMITED LABELS

The success of most deep neural models depends strongly on the availability of large amounts of labeled and annotated data, which is prohibitively expensive to obtain, especially for relatively new tasks. However, a large amount of labeled data may be available for a different, yet similar task. In this section, we consider the scenario where in addition to a small set of good quality labeled data for this new task, a large amount of labeled data for a different task is available. We propose a differentially weighted learning procedure which builds on top of existing gradient descent based learning algorithms and learns to leverage existing labeled data.

3.1 Definitions

We define key concepts of the dataset and supervision signals, which are used throughout the paper.

Weak Supervision: Often we have supervision signals which are derived via "weak annotators", or rules based on heuristics, patterns or "weaker", biased classifiers trained on e.g. non-expert crowd-sourced data or data from different, related domains. All such weaker forms of supervision, constitute weak supervision. The training data whose labels are obtained via any such weak supervision techniques are referred to as weak supervision dataset.

Ground Truth Supervision: On-task supervision signals derived from human annotators, or explicitly obtained via user feedback (e.g. surveys) constitute strong supervision signals. The training data whose labels are derived from such strong, trusted sources of supervision are referred to as ground-truth supervision dataset.

Note that we can generate a large amount of weak training data at almost no cost using the weak annotator or weaker forms of supervision. On the other hand, obtaining ground truth data is a much harder and cost-intensive process. In this paper, we focus on leveraging weak supervision data along with ground truth data to efficiently train neural models for recommendations.

3.2 Notation

Formally, we denote an instance by a feature vector $x \in R^d$, and its corresponding label by $y \in \{-1, 1\}$ for the binary classification case. A dataset $D = \{x_i, y_i\}_{i=1}^N$ consists of $\|N\|$ training instances. We work with two types of datasets: (i) ground truth data and (ii) weak supervision data. Ground truth data is denoted by $D_g = \{x_i^g, y_i^g\}_{i=1}^N$, wherein the label y_i^g acts as ground truth label obtained via explicit human judgments. The weak supervision dataset is denoted by $D_w = \{x_i^w, y_i^w\}_{i=1}^N$, with the weak supervision label obtained through one of the strategies described in Section 3.1. Assume that we receive a small set of ground truth data (D_g) consisting of examples which are annotated with explicit labels, and an additional relatively larger dataset of weak supervision signals (D_w), our goal is to leverage these examples to learn an accurate machine learning model $\phi(\theta, x) : R^d \longrightarrow \{0, 1\}$ where θ is the model parameters.

3.3 Problem Formulation

We hypothesize that the weak supervision data could be leveraged with the ground truth dataset in different ways, that would give improved performance over the case when only ground truth supervision is used. Given access to a small amount of ground truth data (D_g) and a large amount of weak supervision data (D_w), the goal of this work is to devise training strategies which leverage D_w with D_g to train the learning module $\phi(\theta, x) : R^d \longrightarrow \{0, 1\}$.

Formally, we wish to devise training strategies $D_t = f(D_w, D_g)$, where D_t is the final dataset the model will be trained on and $f(.,.)$ represents different strategies to leverage datasets.

In contrast, we have only a limited amount of observations from the true function, i.e. $\|D_w\| \gg \|D_g\|$.

We start with considering different ways of combining these datasets to train the model. We then proceed to devise special training methodologies which rely more on ground truth data while giving special attention to the weak supervision data. Section 4 gives detailed description of these approaches.

4 EXPLOITING WEAK SUPERVISION

In this section, we consider several ways of jointly leveraging D_w and D_g to train learning models. We begin by describing few dataset construction strategies which create a new dataset composed of training instances from both the weak and ground truth data. We then proceed to propose alternative training strategies which pay special attention to the strength of the supervision signal and exploit it to train the model accordingly.

4.1 Data Mixing Strategies

4.1.1 Training on All/No External Data. We begin by simply using all or none of the weak supervision data. In the case where none of the weak supervision data instances are used, the model is entirely trained on ground truth data. Specifically, the training dataset is constructed as:

$$D_t = \{(x_i^g, y_i^g)\} \ \ \forall (x, y) \in D_g \tag{1}$$

On the other hand, when a model is trained using all weak supervision data points, the training data size increases by a big margin, with most of the training data dominated by the weak supervision data.

$$D_t = \{(x_i^g, y_i^g)\} \cup \{(x_j^w, y_j^w)\} \ \ \forall (x_i, y_i) \in D_g \ \& \ (x_j, y_j) \in D_w \tag{2}$$

Finally, a third way of combining data could be to train only on the weak supervision signal. Such a training strategy is useful in cases where the ground truth data is extremely limited and one wants to use them only for evaluation purposes. The three data mixing strategies could be summarized as follows:

$$D_t = \{(x_j^w, y_j^w)\} \ \ \forall (x_j, y_j) \in D_w \tag{3}$$

4.1.2 Similarity Thresholded Subset. A major drawback of including all data points from weak supervision datasets is that when some data is not relevant to the core task at hand or the label is noisy, it leads the model astray and could harm the training process. This is especially true in the case of neural models where noisy labels could mislead the gradient convergence procedure giving a false optimum. A way to alleviate this problem is to consider only the training samples which are closely related to the instances present in the ground truth sample.

Specifically, we select a subset of data instances from D_w which are closest to the data points in the ground truth dataset in the feature space. Let $\psi(x_i, x_j)$ denote the cosine similarity between two feature vectors x_i and x_j, the training dataset is created as follows:

$$D_t = \{(x_i^g, y_i^g)\} \cup \left\{(x_j^w, y_j^w) \ \ s.t. \ \frac{\sum_{i=1}^{\|D_g\|} \psi(x_j, x_i)}{\|D_g\|} > \tau\right\} \tag{4}$$

where τ is a pre-defined similarity threshold. The above strategy could easily be adjusted to give the top-k most similar data points closest to the ground truth dataset.

4.1.3 Interleaved Learning. An alternate similarity based technique would be to find the most similar weak supervision data point for each datapoint in the ground truth data and include it in the training dataset. Specifically,

$$D_i = \bigcup_{i=1}^{\|D_g\|} \bigcup_{j=1}^{\|D_w\|} \{(x_i^g, y_i^g)\} \cup \{(x_j^w, y_j^w) \ \ s.t. \ \frac{\sum_{j=1}^{\|D_w\|} \psi(x_j, x_i)}{\|D_w\|} > \tau\} \tag{5}$$

A key difference between the interleaved and similarity thresholded data mixing techniques is that the former considers similarity per ground truth datapoint when making inclusion/exclusion decision, whereas the latter computes the average similarity of a weak supervision datapoint with the entire ground truth data when deciding to include it in the training dataset.

4.2 Submodular Subset of Weak Supervision

In addition to the above mentioned data mixing strategies, we cast the problem of leveraging weak supervision data as that of subset selection, wherein we wish to select a subset of most useful weak supervision data, based on the available strong supervision data to train our models. Specifically, we propose a submodular subset selection to find the optimal subset of weak data to be included in training.

Submodular functions are discrete functions that model laws of diminishing returns and can be defined as follows [20]: Given a finite set of objects (samples) $X = \{x_1, ..., x_n\}$ and a function $f : 2^S \rightarrow \Re^+$ that returns a real value for any subset $S \subseteq X$, f is submodular if given $S \subseteq S'$, and $x \notin S'$

$$f(S + x) - f(S) \geq f(S' + x) - f(S') \tag{6}$$

That is, the incremental "value" of x decreases when the set in which x is considered grows from S to S'. A function is *monotone submodular* if $\forall S \subseteq S'$, $f(S) \leq f(S')$. Powerful guarantees exist for such subtypes of monotone submodular function maximization. Though NP-hard, the problem of maximizing a monotone submodular function subject to a cardinality constraint can be approximately solved by a simple greedy algorithm [20] with a worst-case approximation factor $(1 - e^{-1})$. This is also the best solution obtainable in polynomial time unless P=NP [7].

Submodularity is a natural model for weak data subset selection in our setting. Indeed, an important characteristic of any data-subset selection technique would be to decrease the value-addition of a new data instance $x^w \in X^w$ based on how much of that query has in common with the subset of queries already selected (S). The value $f(w^w|S)$ of an instance in the context of previously selected subset of instances S further diminishes as the subset grows $S' \supseteq S$.

Intuitively, an optimal subset of weak supervision instances selected for training with strong supervision data should have two characteristics: (i) informativeness, which measures the ability of an instance in reducing the uncertainty of the model and (ii) representativeness, which measures if an instance well represents the possible input patterns of data. Mathematically, the weak data instance subset selection problem can be formulated as selecting the subset of instances S which maximizes the value of $f(S)$ where $f(S)$ captures both the representativeness aspect as well as the informativeness aspects of the weak supervision instance.

To capture both these traits, we model the quality of the subset as: $F(S) = \beta\Phi(S) + (1-\beta)\Psi(S)$ where $\Phi(S)$ captures the representativeness aspect of the instance subset (S) with respect to the entire set of weak supervision data X^w while $\Psi(S)$ rewards selecting informative queries. The parameter β controls the trade-off between the importance of representativeness & informativeness while selecting queries.

4.2.1 Representativeness: $\Phi(S)$. $\Phi(S)$ can be interpreted either as a set function that measures the similarity of instance subset S to the overall query set Xw, or as a function representing some form of "representation" of X^w by S. Most naturally, $\Phi(S)$ should be monotone, as representativeness improves with a larger subset. $\Phi(S)$ should also be submodular: consider adding a new instance to two instances subsets, one a subset of the other.

We employ the same functional form of $\Phi(S)$ as was adopted by Lin *et al.*[11, 12] and Mehrotra *et al.*[16]. Specifically, a saturated coverage function is defined as follows

$$\Phi(S) = \sum_{x \in X^w} min\left\{C_x(S), \alpha C_x(X^w)\right\} \tag{7}$$

where $C_x(S)$ is a set based function defined as $C_x(S) : 2^S \rightarrow \Re$ and $0 \leq \alpha \leq 1$ is a threshold co-efficient. Intuitively, $C_x(S)$ measures how similar S is to instance x or how much of the instance x is covered by the subset S. We define the coverage function $C_x(S)$ in terms of the *coverage* of instances. More specifically, $C_x(S) = \sum_{x' \in S} \psi(x, x')$ where $\psi(x, x) \geq 0$ measures the cosine similarity between instances as in the previous section.

4.2.2 Informativeness: $\Psi(S)$. The $\Phi(S)$ function described above intuitively captures the notion of coverage or representativeness by selecting subset of instances S which are most representative of the entire set of weak supervision instances X^w. While representativeness is an important trait, we also wish to capture the informativeness aspect of queries and select queries which are most informative to the current version of the model. We formulate the functional form of $\Psi(S)$ based on the ability of the instance to perform model update. As a precursor, it is worth mentioning that to define the function $\Phi(S)$ we cluster weak supervision instances into k-clusters and we associate each weak supervision instance to one of these k-clusters. Formally, we define the $\Psi(S)$ function as follows:

$$\Psi(S) = \sum_{i=1}^{K} \sqrt{\sum_{x \in P_i \cap S} \Upsilon_x} \tag{8}$$

where $P_i, i = 1, ..., K$ is the cluster-partition of the set of instances X^w into K-clusters and Υ_x captures the informativeness carried by the instance x based on the current model. The function $\Psi(S)$ rewards cluster-diversity along with valuing informativeness since there is usually more benefit to selecting a instance from a cluster not yet having one of its instances already chosen. As soon as an instance is selected from a cluster, other instances from the same cluster start having diminishing gain owing to the square root function ($\sqrt{2} + \sqrt{1} > \sqrt{3} + \sqrt{0}$).

Many different ways exist to quantify the informativeness of a weak supervision instance Υ_x, including the average or maximum similarity of the instance to ground truth data, or the noise level associated with the weak supervision instance, or the gradient update this weak supervision instance would have resulted had we used it to update parameters of the trained neural model. The specific use-case and availability of appropriate information guide the selection of the specific way to quantify such informativeness measure.

4.2.3 Greedy Optimization. Having defined the individual functions based on the different paradigms, we formulate the overall weak supervision instance subset selection problem as the selection of the subset S of instances which maximizes the following function:

$$\begin{aligned} F(S) \quad = \quad & \beta \sum_{x \in X^w} min\left\{\sum_{x' \in S} \psi(x, x'), \alpha \sum_{x' \in X^w} \psi(x, x')\right\} \\ & + \quad (1-\beta) \sum_{i=1}^{K} \sqrt{\sum_{x \in P_i \cap S} \Upsilon_x} \end{aligned} \tag{9}$$

Modelling the instance selection problem in such an objective provides many advantages. Firstly, the submodular formulation provides a natural way of coupling the different aspects of instance selection. Secondly, the above formulation can be optimized efficiently and scalably given the monotone submodular form of the function $F(S)$. Assuming we wish to select a subset of N weak supervision instances from the total set of X^w instances, the problem reduces to solving the following optimization problem:

$$S^* =_{S \subseteq X^w, |S| \leq N} F(S) \tag{10}$$

While solving this problem exactly is NP-complete [7], techniques like ILP [19] can be used but scaling it to bigger datasets becomes prohibitive. Since the function $F(S)$ is submodular, it can be shown that a simple greedy algorithm will have a worst-case guarantee of $f(S^*) \geq (1 - \frac{1}{e})F(S_{opt}) \approx 0.63F(S_{opt})$ where S_{opt} is the optimal and S^* is the greedy solution [7]. The greedy solution works by starting with an empty set and repeatedly augmenting the set as

$$S \leftarrow S \cup_{x \in X^w \setminus S} F(q|S) \tag{11}$$

until we select the N number of instances of weak supervision in the subset we intended. Finally, we use the selected subset of weak supervision instances to enrich the training data:

$$D_t = \{(x_i^g, y_i^g)\} \cup \left\{(x_j^w, y_j^w) \;\; s.t. \; x_j^w \in S\right\} \tag{12}$$

where S is the subset of weak supervision instances selected using the submodular subset selection strategy.

While these strategies present different ways of creating a training dataset, the training methodology does not differentiate between instances from the ground truth dataset and weak supervision dataset. We next propose strategies which adapt the training strategy based on the data instance.

4.3 Differentially Weighted Learning

Beyond constructing augmented datasets, an additional way to leverage weak training data is by adapting the training mechanism of the model to take into account the label quality and trust. To this end, we propose a *Differential Weighting Learning* strategy, wherein we modulate the parameter updates to the training module on a per-datapoint basis according to the quality and relevance of the weak supervision sample to the task and ground-truth dataset. A related approach was adopted by Dehghani *et al.* [3] to train a student-teacher network with access to quality for each label.

Given a training example, stochastic gradient descent (SGD) performs a parameter update for each training example $\theta = \theta - \eta.\ \triangledown_{\theta}\ J(\theta)$. Adaptive Moment Estimation (Adam) [9] computes adaptive learning rates for each parameter. In addition to storing an exponentially decaying average of past squared gradients, Adam also keeps an exponentially decaying average of past gradients:

$$m_t \leftarrow \beta_1.m_{t-1} + (1 - \beta_1).g_t \tag{13}$$

$$v_t \leftarrow \beta_2.v_{t-1} + (1 - \beta_2).g_t^2 \tag{14}$$

where m_t and v_t are estimates of the first moment (the mean) and the second moment (the uncentered variance) of the gradients respectively, hence the name of the method. These momentum estimates are then used to update the parameters:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v_t}} + \epsilon}\hat{m_t} \tag{15}$$

where $\hat{m_t} = \frac{m_t}{1-\beta_1^t}$ and $\hat{v_t} = \frac{v_t}{1-\beta_2^t}$ are bias-corrected moment estimates, m_t, v_t are exponential moving averages of the gradient and the squared gradient, β_1, β_2 control the exponential decay rates of these moving averages, $\hat{m_t}, \hat{v_t}$ are bias corrected estimates since m_t, v_t are initialized with vector of 0's hence getting biased towards zero, α is learning rate.

To enable differentially weighted learning, we build on top of Adam and attempt to make it adaptive to each training sample. Specifically, for each training instance contributed by the weak supervision dataset, we compute its similarity to the ground truth dataset and use it to determine the step size for each iteration of Adam. So, intuitively, for data points contributed by ground truth dataset, we have high confidence and a large step-size for updating the parameters. However, for data points contributed by the weak supervision dataset, we down-weight the training steps of the gradient descent method. This allows the model to *differentially weight* the importance of each training point and learn accordingly.

We propose a change in learning rate to give more weight to the more trustworthy training samples.

$$\eta_2 = exp[-\gamma * (1 - \psi(x_j, x_i))] \tag{16}$$

where γ is a scalar positive hyperparameter contributing small value whenever training example is relevant and large value whenever the training example is weak. ψ helps us encode the weight of the weak examples. Adam thus behaves like a heavy ball with friction, which prefers flat minima in the error surface compared.

To quantify ψ, we use average cosine similarity between high-quality data and the weakly labeled data. This similarity helped with weighing each training instance differentially based on how similar it is to ground truth data. Often when similarity based quantification is not preferred, one can consider alternate approaches, including marginal gain based quantification, which weights each instance by the marginal gain offered by it to the pool of strong supervision dataset based on the coverage function Φ defined in Section 4.2. Appendix A gives a step-by-step walk through of the algorithmic details.

Algorithm 1 *Momentum damped Adam*

```
Require: $\eta$ Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$:Exponential decay rates for the moment estimates
Require: $f(\theta)$:Stochastic objective function with parameters $\theta$
Require: $\theta_0$:Initial parameter vector
$m_0 \leftarrow 0$ (Initialize $1^{st}$ moment parameter)
$v_0 \leftarrow 0$ (Initialize $2^{nd}$ moment parameter)
$t \leftarrow 0$ (Initialize timestep)
while $\theta_t$ not converged do
    $t \leftarrow t + 1$
    $g_t \leftarrow \nabla\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
    $m_t \leftarrow \beta_1.m_{t-1} + (1 - \beta_1).\zeta(x_i).g_t$ (Update biased first moment)
    $v_t \leftarrow \beta_2.v_{t-1} + (1 - \beta_2).\zeta(x_i).g_t^2$ (Update biased second raw moment)
    $\hat{m_t} \leftarrow m_t/(1 - \beta_1^t)$ (Compute bias-corrected first moment)
    $\hat{v_t} \leftarrow v_t/(1 - \beta_2^t)$ (Compute bias-corrected second raw moment)
    $\theta_t \leftarrow \theta_{t-1} - \eta.\hat{m_t}/(\sqrt{\hat{v_t}} + \epsilon)$ (Update parameters)
end while
return $\theta_t$ =0
```

4.4 Momentum Damped Differentially Weighted Learning

Instead of adjusting the parameter updates resulting from weak supervision training data points, we propose an alternative definition of Adam wherein we purposefully dampen the momentum updates based on our trust in the usefulness of the training instance. Momentum is a simple and widely used trick which allows gradient-based optimizers to pick up speed along low curvature directions. Gradient descent methods SGD has trouble navigating ravines, i.e. areas where the surface curves much more steeply in one dimension than in another, which are common around local optima. Momentum [22] is a method that helps accelerate SGD in the relevant direction and dampens oscillations by adding a fraction of the update vector of the past time step to the current update vector.

As shown in Algorithm 1, Adam updates exponential moving averages of the gradient (m_t) and the squared gradient (v_t) where the hyper-parameters $\beta_1, \beta_2 \in [0, 1)$ control the exponential decay rates of these moving averages.

$$m_t \leftarrow \beta_1.m_{t-1} + (1 - \beta_1).g_t \tag{17}$$

$$v_t \leftarrow \beta_2.v_{t-1} + (1 - \beta_2).g_t^2 \tag{18}$$

We modify the Adam algorithm by including a weighting factor which further dampens contribution stemming out of the current training example's gradient update if the current training sample is contributed by weak supervision dataset. At each optimization step, these velocities and momentum terms are updated and then averaged to produce the final velocity/momentum used to update the parameters. This updated iterative procedure can be written as follows:

$$m_t \leftarrow \beta_1.m_{t-1} + (1 - \beta_1).\zeta(x_i).g_t \tag{19}$$

$$v_t \leftarrow \beta_2.v_{t-1} + (1 - \beta_2).\zeta(x_i).g_t^2 \tag{20}$$

Signal	Description
session length	duration of entire session in seconds
ms played	total milli-seconds streamed
home dwell time	session duration minus downstream time spent streaming content
avg interaction time	time spent on the Homepage interacting with slates
time to success	time until first stream
songs played	number of songs played
no of interactions	total number of clicks on Homepage
nSlates	no of slates interacted with
didScroll	whether or not the user scrolled
max depth	number of slates vertically scrolled
no of exits	number of exits from Homepage to any playlist
abandoned	binary 0/1 if session was abandoned without interaction

Table 1: Description of user interaction signals used.

where the momentum damping co-efficient is defined based on relevance of training example, or based on marginal gain in relevance utility. For the case of the relevance of the training example, we compute $\zeta(x_i)$ as:

$$\zeta(x_i) = \begin{cases} \frac{\sum_{i=1}^{\|D_g\|} \psi(x_j, x_i)}{\|D_g\|} & x_i \in D_w \\ 1 & x_i \in D_g \end{cases}$$

while, for the case of marginal gain in relevance utility, we define $\zeta(x_i)$ as:

$$\zeta(x_i) = \begin{cases} \Phi(X^s \cup x) - \Phi(X^s) & x_i \in D_w \\ 1 & x_i \in D_g \end{cases}$$

with $\Phi(S)$ denoting the coverage function defined in Sec 4.2 for a set S. By taking advantage of how relevant or important a training instance is, momentum damped Adam can optimize well over ill-conditioned curvature and adapt the training based on weak supervision labels.

The different ways of leveraging weak supervision dataset via data mixing strategies and weak supervision aware training, i.e., differential weight and momentum dampening, allows us to adapt our training procedure based on ground truth and weak supervision data instances. To show the utility of such approaches, we consider two different use cases, discussed next.

5 CASE STUDIES IN MUSIC RECOMMENDATION & VIDEO QA

To compare the performance gains obtained by leveraging weak supervision signals, we consider two different use-cases. First, we consider personalized music recommendation and investigate the problem of predicting user satisfaction with the slate recommendation. Second, we consider the task of video question-answering from transcripts.

5.1 Case Study I: User Satisfaction with Slate Recommendations

Detecting and understanding the implicit measures of user satisfaction are essential for enhancing recommendation quality. Developing a better understanding of how users interact with such recommender systems and predicting user's satisfaction is important not only for improving user experience but also for effective and efficient optimization of the recommendation algorithm. Since obtaining explicit feedback from users is prohibitively expensive and challenging to implement in real-world systems, commercial systems rely on exploiting *implicit* feedback signals derived from user activity. When users interact with the recommendations served, they leave behind fine-grained traces of interaction patterns, which could be leveraged for predicting how satisfied was their experience, and for developing metrics of user satisfaction [15].

Our first case study, we focus on satisfaction prediction problem in a streaming music recommendation and obtain both explicit and implicit signals of user satisfaction. Explicit labels from users are treated as the ground-truth dataset (D_g) while implicit satisfaction signals are treated as weak supervision dataset (D_w). We next briefly describe processing to collect the two.

5.1.1 D_g = Ground Truth Data. We work with real-world user data from Spotify, a large music streaming service, and conduct a large scale in-app survey to collect judgments about intents and user satisfaction. We trigger the in-app survey to a random sample of over 3 million users and observe a response rate of 4.5%. In total, we received responses from over 116000 users, resulting in over 200K judgments about intents and satisfaction combined. For each session, we collected back-end logs of user interactions with the slates of recommendation and extracted data for all the different interaction signals mentioned in Table 1. For each session, we use the intent selected by the user as the session intent and use the satisfaction prediction label given by the user to train and evaluate our satisfaction prediction models.

5.1.2 D_w = Implicit Weak Supervision Data. The Homepage rendered for a user is rich enough to allow him or her to interact with it in a myriad of ways, including clicking on playlists, scrolling vertically to view more slates, scrolling horizontally to view more playlists in a specific slate, pausing to read and visually absorb content, clicking and consuming content via streaming, among others. While past work on understanding user interaction in mobile search setting has proposed few signals [10, 30], we additionally propose several new signals resulting from the specific *Slate Recommendation* scenario considered in this work.

We use back-end logs of user interactions and extract four different types of interaction signals for each user session: (i) temporal signals, (ii) surface-level signals, (iii) downstream signals and (iv) derivative signals. Table 1 provides a detailed description of the different interaction signals extracted for each user session.

For each user session, the corresponding features extracted (Table 1) are used as feature vectors (x_i) and a heuristic based satisfaction label is assigned to each user session based on:

$$y_1 = \begin{cases} 1 & \text{dwell time} \geq 20s \land \text{songs played}>5 \land \text{session length}>500s \\ -1 & \text{dwell time} \geq 20s \land \text{songs played}=0 \land \text{did scroll} =1 \end{cases}$$

Effectively, we consider extreme cases of user satisfaction and dissatisfaction, i.e. sessions wherein users have a long session and they stream music are tagged as +1, while sessions wherein users struggle are tagged as -1. This label serves as weak supervision label for the SAT prediction problem.

5.1.3 Predicting Satisfaction. Our main goal is to understand and predict user satisfaction (SAT) using interaction data. To this end, we extracted detailed user-interaction signals and identified different intents users might have. In this section, we leverage the extracted signals and intents and present techniques to predict user satisfaction using the signals. The neural model has input in the form of user interaction signals, intents with weights on different

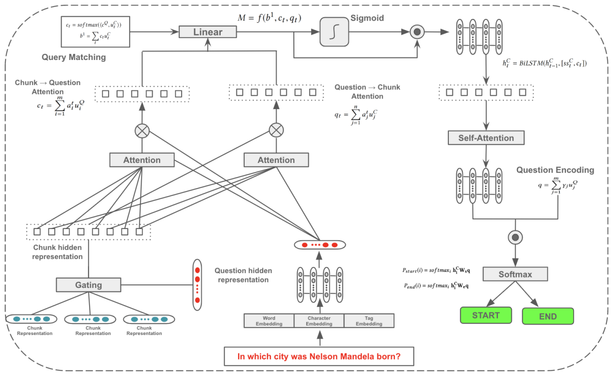
$c_t = softmax(\langle c^Q, u_t^C \rangle)$
$b^1 = \sum_t c_t u_t^C$
Query Matching
Linear
$M = f(b^1, c_t, q_t)$
Sigmoid
$h_t^C = BiLSTM(h_{t-1}^C, [ss_t^C, c_t])$
Chunk → Question Attention
$c_t = \sum_{i=1}^{m} a_i^t u_i^Q$
Question → Chunk Attention
$q_t = \sum_{j=1}^{n} a_j^t u_j^C$
Self-Attention
Question Encoding
$q = \sum_{j=1}^{m} \gamma_j u_j^Q$
Attention
Attention
Chunk hidden representation
Question hidden representation
Gating
$P_{start}(i) = softmax_i\ h_i^C W_s q$
$P_{end}(i) = softmax_i\ h_i^C W_e q$
Softmax
START
END
Chunk Representation
Chunk Representation
Chunk Representation
Word Embedding
Character Embedding
Tag Embedding
In which city was Nelson Mandela born?

Figure 1: Overview of video comprehension model

user interaction signals and intent. The model was trained on binary cross-entropy. We use a neural model for SAT prediction which is composed of several convolution layers, with few dense layers. We used a minibatch size of 32, *Adam* for optimization as described in [9] with hyperparameters such as learning rate of 0.004, β_1 as 0.9 and β_2 as 0.996. Dropout with p = 0.2 applied to convolutional layers. Hence, it is capable of giving different weight to different training instances according to the relevance of the data.

5.2 Case Study II: Video Comprehension

As a second case study, we consider the task of video recommendation in a question answering setting. Specifically, given some question, we need to find the video and parts of the video which contains the best possible answer based on information from video transcripts. It is prohibitively challenging to collect explicit data on videos since it requires a judge to watch multiple videos to extract answers. This problem aligns well with the overall problem considered in the paper: we have access to a few sets of QA pairs curated from domain experts and access to a larger additional external data comprising QA training data from an unrelated domain (SQuAD[23]) which enhances our dataset. While a detailed overview is beyond the scope of this work, below we briefly describe the neural model used for the task of video recommendation.

5.2.1 D_w = Distant Supervision. We use the SQuAD dataset for pre-training our deep neural model. The Stanford Question Answering Dataset (SQuAD) Rajpurkar et al.[23] is a dataset for machine comprehension based on Wikipedia. The dataset contains 87k examples for training and 10k for development, with each example composed of a paragraph extracted from a Wikipedia article and an associated human-generated question.

5.2.2 D_g = Explicit Labels. We made use of Amazon Mechanical Turk for getting labeled data for a total of 1021 videos with answers. We followed a rigorous process for the video annotation task, wherein for each question, a set of videos were annotated by a minimum of 3 different human judges. Judges went through a brief training video describing the task wherein they were explained and shown examples of question-answering tasks. Overall, we obtained an inter-rater agreement of 0.69 (Fleiss' kappa) which implies substantial agreement among judges. This explicitly labeled data serves as the ground truth data for the video QA task.

5.2.3 Gated Attention model for Video QA. We next briefly describe the neural model used for recommending a video for a given question. Given a question and any number of candidate videos, the system must be able to come up with an appropriate answer for the given question and recommend the video which contains that answer. To extract answers from the video content, we worked with transcripts data and propose a gated attention-based neural model with content bifurcation module to break down a large piece of transcript text into small chunks and then perform QA on those chunks. To tackle this problem, we have broken it into two sub-problems:- Video retriever and answer generation.

For each question, the video retriever extracts the top-k videos and tags them with transcripts. For each retrieved video, the content bifurcation module extracts meaningful chunks which are then passed onto the gated attention module. Finally, the answer generation module considers each chunk and finds the answer boundaries and outputs the confidence score for each chunk. The chunk ranking module considers all chunks from all videos and ranks the answers based on the model's confidence scores and outputs the final answer.

Given a question and a bifurcated chunk with tokens $\{q_1, ..., q_m\}$ and $\{c_1, c_2, ..., c_n\}$ respectively, we develop a gated attention LSTM model which we apply on the chunk, to learn question representation and question-aware chunk representation, which are used to detect answer spans. For a given question, we consider all the retrieved videos, select relevant answers from each video and use the above equation to find the top videos to be recommended for any given question. Figure 1 gives a pictorial overview of the neural approach employed for question specific video recommendation.

6 EXPERIMENTS

We next discuss results which compare the different proposed strategies of learning from weak supervision data on the two case studies described above. Unless otherwise stated, all evaluation results reported in the paper are on a hold-out ground-truth dataset.

6.1 Impact of Weak Supervision

We begin by investigating the impact of including weak supervision dataset when training a model. The All- and Sub- variants of the methods denote the method trained using all data and subset of data selected via submodular subset selection approach. We report results for the best performing threshold values, and present results in Tables 2and 3.

6.1.1 Case study: Video QA. Table 2 presents result which compare the different data mixing strategies presented in Section 4.1. We observe significant gains across all metrics when weak supervision data is used, with over 40% gains in F1 score when both ground truth and weak supervision signal is used. Interestingly for this task, the ground truth data is not enough to train the model well, due to which a model trained exclusively on weak supervision signal outperforms a model trained on only ground truth data.

Furthermore, we see that intelligently picking up relevant weak supervision data points results in improving performance. Both similarity thresholded and interleaved mixing strategy perform better than other techniques. Finally, we observe that the submodular subset selection results in the best performance across all data mixing strategies. This is expected since submodular subset selection builds on top of the intuition covered by the similarity thresholded mixing strategy but also considers potential gradient update weight. The relative weight of $\beta = 0.3$ (the ratio of weak data to strong data was around 7:3 in most of our experiments) indicates the relative importance given to gradient update weight.

6.1.2 Case study: User SAT prediction. Interestingly, we observe that adding weak supervision signals in the SAT prediction task hurts the performance of the model, with a model trained only D_g only performing better than both D_w as well as $D_w \cup D_g$ training instances. This highlights that the labels provided by users themselves are much more useful than the conservative satisfaction labels generated using implicit feedback mechanism.

However, implicit feedback based weak supervision label could still be used to improve predictive accuracy. We observe that when considering only the most similar data points from the weak supervision data, the model performs better than D_g only case. This highlights that adding more training data which is similar to the explicit data is indeed helpful. Similar to the Video QA task, interleaved style data mixing strategy performs better than similarity thresholded approach. Indeed, the interleaved approach selects data points which are close to specific data points in the ground truth data. Since the similarity thresholded data points consider overall similarity, they may end up picking noisy datapoints whose average similarity is high but individually, they may not be much relevant to any ground truth data point. Finally, we observe that similar to the video QA case, submodular subset selection outperforms all other data mixing strategies.

6.2 Representativeness vs Informativeness

As demonstrated above, submodular subset selection of weak instances to add to strong supervision data performs best among the considered data mixing strategies. To better understand the performance of the submodular approach, we conducted an experiment on the video comprehension case-study of the relative importance of the two aspects of representativeness and informativeness and how they contribute to the overall model performance.

We observed that a relative weighting scheme of $\beta = 0.3$ (which weighs representativeness-vs-informativeness in 3:7 proportions) works best for weak instance subset selection, with a gradual increase in performance as more training data is available. This highlights that while representativeness is important, selecting informative instances from the different cluster indeed helps.

6.3 Impact of Adaptive Training

We next investigate the benefit obtained due to training instance-specific adaptive training. Based on the results obtained for differential weighting and momentum damped Adam training shown in Tables 2 and 3, we observe that adaptive training methods perform better than all other data mixing approaches considered. This demonstrates that equal importance should not be given to ground truth data and weak supervision data. By merely adjusting the learning rate in gradient descent style optimizers based on the relevance of weak supervision datapoint gives about 10% accuracy gain in SAT prediction task and over 15% gain in video QA task.

Approach	F1	ndcg@1	ndcg@5	prec@1	prec@5
D_g only	0.34	0.43	0.31	0.45	0.29
D_w only	0.44	0.58	0.31	0.605	0.321
$D_g \cup D_w$	0.509	0.318	0.624	0.653	0.325
Similarity thresholded	0.512	0.629	0.324	0.658	0.33
Interleaved	0.52	0.635	0.362	0.664	0.38
Submodular	0.54	0.65	0.374	0.68	0.41
Differential Weighting (All)	0.57	0.67	0.403	0.7	0.43
Differential Weighting (Sub)	0.59	0.68	0.426	0.72	0.44
Momentum Damped (All)	0.61	0.69	0.428	0.75	0.46
Momentum Damped (Sub)	**0.62**$^{*\&}$	**0.72**$^{*\&}$	**0.438**$^{*\&}$	**0.77**$^{*\&}$	**0.47**$^{*\&}$

Table 2: Comparing data mixing strategies on Video QA task. * and & signify statistically significant difference between the method and the two best performing data mixing strategies using χ^2 test with $p \leq 0.05$

Approach	Accuracy	Precision	Recall	F1Score
D_g only	0.55	0.60	0.54	0.57
D_w only	0.5	0.51	0.49	0.5
$D_g \cup D_w$	0.514	0.54	0.51	0.525
Similarity thresholded	0.59	0.62	0.58	0.6
Interleaved	0.6	0.62	0.59	0.605
Submodular	0.63	0.635	0.62	0.627
Differential Weighting (All)	0.632	0.64	0.62	0.629
Differential Weighting (Sub)	0.643	0.652	0.63	0.634
Momentum Damped (All)	0.65	0.66	0.63	0.645
Momentum Damped (Sub)	**0.66**$^{*\&}$	**0.67**$^{*\&}$	**0.65**$^{*\&}$	**0.66**$^{*\&}$

Table 3: Comparing data mixing strategies on SAT prediction task. * and & signify statistically significant difference between the method and the two best performing data mixing strategies using χ^2 test with $p \leq 0.05$

Furthermore, we observe that momentum damping for unrelated weak supervision datapoints has a positive impact on performance. Training the model via momentum damped Adam optimizer achieves the best performance overall, with 5% improvement over differential weighting approach, and a substantial and statistically significant 23% gain in accuracy over the approach of combining all data points.

6.4 Amount of Weak Supervision Used

We additionally investigated the amount of weak supervision data needed when training neural models for the video QA task. In Figure 2, we present a comparison of the data mixing strategies with the adaptive learning strategies: differential weighting and momentum damped, across different amounts of training data. We observe superior performance of the adaptive learning approaches over data mixing approaches, with momentum damped learning to perform best across all methods, metrics and amount of weak supervision data used. Further, we observe a steady increase in performance upon adding weak supervision signals. Indeed, adding more relevant data from SQuAD would expose the model to new topics and questions, and the resulting coverage would help the model perform better on the ground-truth data.

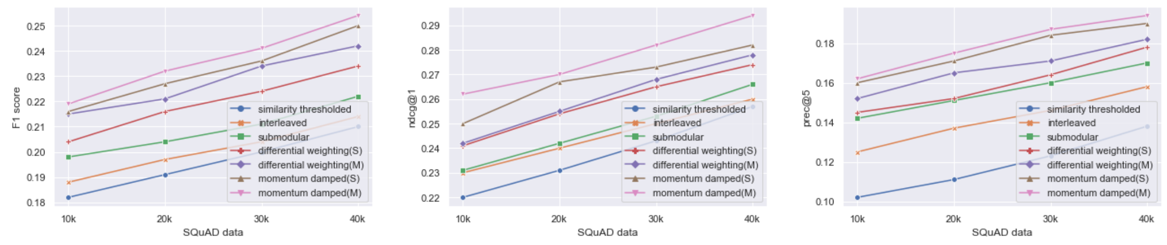

Figure 2: Comparison of approaches for Video QA across three metrics and varying amount of weak supervision data.

6.5 Generalizability across Tasks & Models

User satisfaction prediction and question-based video recommendation are two separate tasks with little conceptual overlap. Evaluating the proposed approaches on such different task highlights that the results obtained are not a characteristic of a specific task or neural model. Furthermore, the kind of weak supervision signal used in the two tasks is also different. We leveraged implicit interaction data from users to craft heuristic weak supervision labels for the SAT prediction task, while we used distant supervision for the video QA task. This highlights that the proposed approach is robust to different kinds of weak supervision signals.

Finally, we adopt a different momentum-based approach: Nadam [5] to investigate dependencies in the proposed approach on the specific type of optimizer used. We obtained similar performance gains when we replaced Adam optimizer in the neural training process by Nadam; which demonstrates that momentum damping is useful regardless of the specific optimizer used. We omit detailed results for Nadam due to space constraints.

7 CONCLUSION

We address the problem of learning from limited labeled data by proposing different data mixing strategies as well as proposing adaptive training algorithms, which enable the model to leverage weak as well as strong supervision data. Our findings demonstrate that momentum damped adaptive learning strategy is better than simply combining weak and strong labeled datasets. We contend that the proposed techniques apply to other related problems around learning with limited labels and provide practitioners with ways of leveraging external labeled data.

REFERENCES

[1] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. [n.d.]. Algorithms for hyper-parameter optimization. In *NIPS 2011*.

[2] Carla E Brodley and Mark A Friedl. 1999. Identifying mislabeled training data. *Journal of artificial intelligence research* 11 (1999), 131–167.

[3] Mostafa Dehghani, Mehrjou, Gouws, Kamps, and Schölkopf. 2017. Fidelity-weighted learning. *arXiv preprint arXiv:1711.02799* (2017).

[4] Mostafa Dehghani, Aliaksei Severyn, Sascha Rothe, and Jaap Kamps. 2017. Learning to learn from weak supervision by full supervision. *arXiv preprint arXiv:1711.11383* (2017).

[5] Timothy Dozat. 2016. Incorporating nesterov momentum into adam. (2016).

[6] John Duchi, Elad Hazan, and Yoram Singer. [n.d.]. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR 2011* ([n. d.]).

[7] Uriel Feige. 1998. A threshold of ln n for approximating set cover. *J. ACM* (1998).

[8] Rahul Kidambi, Praneeth Netrapalli, Prateek Jain, and Sham Kakade. 2018. On the insufficiency of existing momentum schemes for stochastic optimization. In *2018 Information Theory and Applications Workshop (ITA)*. IEEE, 1–9.

[9] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[10] Julia Kiseleva, Kyle Williams, Hassan Awadallah, Crook, Zitouni, and Anastasakos. [n.d.]. Predicting user satisfaction with intelligent assistants. In *SIGIR 2016*.

[11] Hui Lin and Jeff Bilmes. [n.d.]. Multi-document summarization via budgeted maximization of submodular functions. In *NAACL 2010*.

[12] Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. 510–520.

[13] Eran Malach and Shai Shalev-Shwartz. 2017. Decoupling" when to update" from" how to update". In *Advances in Neural Information Processing Systems*. 960–970.

[14] Rishabh Mehrotra, Ahmed Hassan Awadallah, Milad Shokouhi, Emine Yilmaz, Imed Zitouni, Ahmed El Kholy, and Madian Khabsa. 2017. Deep sequential models for task satisfaction prediction. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. 737–746.

[15] Rishabh Mehrotra, Mounia Lalmas, Doug Kenney, Thomas Lim-Meng, and Golli Hashemian. 2019. Jointly leveraging intent and interaction signals to predict user satisfaction with slate recommendations. In *The World Wide Web Conference*. 1256–1267.

[16] Rishabh Mehrotra and Emine Yilmaz. 2015. Representative & informative query selection for learning to rank using submodular functions. In *Proceedings of the 38th international ACM sigir conference on research and development in information retrieval*. 545–554.

[17] Rishabh Mehrotra, Imed Zitouni, Ahmed Hassan Awadallah, Ahmed El Kholy, and Madian Khabsa. 2017. User interaction sequences for search satisfaction prediction. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 165–174.

[18] Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. [n.d.]. Distant supervision for relation extraction without labeled data. In *ACL 2009*.

[19] George L Nemhauser and Laurence A Wolsey. 1988. *Integer and combinatorial optimization*. Vol. 18. Wiley New York.

[20] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. 1978. An analysis of approximations for maximizing submodular set functions-I. *Mathematical Programming* (1978).

[21] Giorgio Patrini, Frank Nielsen, and Carioni. [n.d.]. Loss factorization, weakly supervised learning and label noise robustness. In *ICML 2016*.

[22] Ning Qian. 1999. On the momentum term in gradient descent learning algorithms. *Neural networks* 12, 1 (1999), 145–151.

[23] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* (2016).

[24] A Ratner, S Bach, P Varma, and C Ré. [n.d.]. Weak supervision: the new programming paradigm for machine learning. Hazy Research.

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

[26] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. [n.d.]. On the importance of initialization and momentum in deep learning. In *ICML 2013*.

[27] T Tieleman and G Hinton. 2017. Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *Technical Report.* (2017).

[28] Arash Vahdat. 2017. Toward robustness against label noise in training deep discriminative neural networks. In *Advances in Neural Information Processing Systems*. 5596–5605.

[29] Andre Wibisono and Ashia C Wilson. 2015. On accelerated methods in optimization. *arXiv preprint arXiv:1509.03616* (2015).

[30] Kyle Williams, Julia Kiseleva, Aidan C Crook, Zitouni, Awadallah, and Khabsa. [n.d.]. Detecting good abandonment in mobile search. In *WWW 2016*.

[31] Ashia C Wilson, Benjamin Recht, and Michael I Jordan. 2016. A lyapunov analysis of momentum methods in optimization. *arXiv preprint arXiv:1611.02635* (2016).

Algorithm 2 *Differentially Weighted Adam*

```
Require: $\eta = \eta_1\eta_2$ Stepsize
Require: $\beta_1, \beta_2 \in [0, 1)$:Exponential decay rates
Require: $f(\theta)$:Stochastic objective function with parameters $\theta$
Require: $\theta_0$:Initial parameter vector
$m_0$ ←0 (Initialze $1^{st}$ moment parameter)
$v_0$ ←0 (Initialze $2^{nd}$ moment parameter)
$t$ ←0 (Initialze timestep)
while $\theta_t$ not converged do
    $t \leftarrow$ t + 1
    $g_t \leftarrow \nabla\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective)
    $m_t \leftarrow \beta_1.m_{t-1} + (1 - \beta_1).g_t$ (Update biased first moment)
    $v_t \leftarrow \beta_2.v_{t-1} + (1 - \beta_2).g_t^2$ (Update biased second raw moment)
    $\hat{m_t} \leftarrow m_t/(1 - \beta_1^t)$ (Compute bias-corrected first moment)
    $\hat{v_t} \leftarrow v_t/(1 - \beta_2^t)$ (Compute bias-corrected second raw moment)
    $\eta \leftarrow \eta_1 * \eta_2$ (Compute adaptive update of stepsize)
    $\theta_t \leftarrow \theta_{t-1} - \eta.\hat{m_t}/(\sqrt{\hat{v_t}} + \epsilon)$ (Update parameters)
end while
return $\theta_t$ (Resulting parameters)
```

A APPENDIX A: DIFFERENTIALLY WEIGHTED LEARNING

To help ease reproducibility, we provide a more detailed step-by-step algorithmic walk through of the method here. Algorithm 2 summarizes pseudocode for modulated Adam. Here, f(θ) is a noisy objective function which is differentiable w.r.t θ. We are interested in minimizing the expected value of this function, E[f(θ)] w.r.t. its parameters θ. With $f_1(\theta), ...,, f_T(\theta)$ we denote the realisations of the stochastic function at subsequent timesteps 1, ..., T. The stochasticity might come from the evaluation at random subsamples (minibatches) of datapoints, or arise from inherent function noise. With $g_t = \nabla f_t(\theta)$ we denote the gradient, i.e. the vector of partial derivatives of f_t, w.r.t θ evaluated at timestep t.

B APPENDIX B: HYPERPARAMETER SEARCH

To make fair comparison across different techniques, we use Sequential Model-Based Optimization[1] to search for best hyperparameters for each approach. For both case-studies, we use the validation dataset for hyper parameter search, which are then used to perform evaluation on test set. We conduct hyperparameter search for similarity thresholds, learning rates and example weights as applicable.