BanditMTL: Bandit-based Multi-task Learning for Text Classification

Yuren Mao[1], **Zekai Wang**[2], **Weiwei Liu**[2*], **Xuemin Lin**[1], **Wenbin Hu**[2]
[1]School of Computer Science and Engineering, University of New South Wales
[2]School of Computer Science, Wuhan University
yuren.mao@unsw.edu.au, {wzekai99,liuweiwei863}@gmail.com
lxue@cse.unsw.edu.au, hwb@whu.edu.cn

Abstract

Task variance regularization, which can be used to improve the generalization of Multi-task Learning (MTL) models, remains unexplored in multi-task text classification. Accordingly, to fill this gap, this paper investigates how the task might be effectively regularized, and consequently proposes a multi-task learning method based on adversarial multi-armed bandit. The proposed method, named BanditMTL, regularizes the task variance by means of a mirror gradient ascent-descent algorithm. Adopting BanditMTL in the multi-task text classification context is found to achieve state-of-the-art performance. The results of extensive experiments back up our theoretical analysis and validate the superiority of our proposals.

1 Introduction

Multi-task Learning (MTL), which involves the simultaneous learning of multiple tasks, can achieve better performance than learning each task independently (Caruana, 1993; Ando and Zhang, 2005). It has achieved great success in various applications, ranging from summary quality estimation (Kriz et al., 2020) to text classification (Liu et al., 2017).

In the multi-task text classification context, MTL simultaneously learns the tasks by minimizing their empirical losses together; for example, by minimizing the mean of the empirical losses for the included tasks. However, it is common for these tasks to be competing. Minimizing the losses of some tasks increases the losses of others, which accordingly increases the task variance (variance between the task-specific loss). Large task variance can lead to over-fitting in some tasks and under-fitting in others, which degenerates the generalization performance of an MTL model. To illustrate this issue, it is instructive to consider a case of two-task learning, where task 1 and task 2 are conflicting binary classification tasks. When the task variance is uncontrolled, it is possible that the empirical loss of task 1 will converge to 0, while the empirical loss of task 2 will converge to 0.5. In such a case, although the mean of the empirical losses is decreasing, task 1 overfits and task 2 underfits, which leads to poor generalization performance.

To address the problem caused by uncontrolled task variance, it is necessary to implement task variance regularization, which regularizes the variance between the task-specific losses during training. However, existing deep MTL methods, including both adaptive weighting sum methods (Kendall et al., 2018; Chen et al., 2018; Liu et al., 2017) and multi-objective optimization-based methods (Sener and Koltun, 2018; Mao et al., 2020b), ignore the task variance. Overlooking task variance degenerates an MTL model's generalization ability.

To fill this gap and further improve the generalization ability of MTL models, this paper proposes a novel MTL method, dubbed BanditMTL, which jointly minimizes the empirical losses and regularizes the task variance. BanditMTL is proposed based on linear adversarial multi-armed bandit and implemented with a mirror gradient ascent-descent algorithm. Our proposed approach can improve the performance of multi-task text classification.

Moreover, to verify our theoretical analysis and validate the superiority of BanditMTL in the text classification context, we conduct experiments on two classical text classification problems: sentiment analysis (on reviews) and topic classification (on news). The results demonstrate that applying variance regularization can improve the performance of a MTL model; moreover, BanditMTL is found to outperform several state-of-the-art multi-task text classification methods.

[*]Corresponding author.

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics
and the 11th International Joint Conference on Natural Language Processing, pages 5506–5516
August 1–6, 2021. ©2021 Association for Computational Linguistics

2 Related Works

Multi-task Learning methods jointly minimize task-specific empirical loss based on multi-objective optimization (Sener and Koltun, 2018; Lin et al., 2019; Mao et al., 2020a) or optimizing the weighted sum of the task-specific loss (Liu et al., 2017; Kendall et al., 2018; Chen et al., 2018). The multi-objective optimization based MTL can converge to an arbitrary Pareto stationary point, the task variance of which is also arbitrary. While the weighted sum methods focus on minimizing the weighted average of the task-specific empirical loss, they do not consider the task variance. To fill the gap in existing methods, this paper proposes to regularize the task variance, which will significantly impact the generalization performance of MTL models.

Variance-based regularization has been used previously in Single-task Learning to balance the trade-off between approximation and estimation error (Bartlett et al., 2006; Koltchinskii et al., 2006; Namkoong and Duchi, 2017). In the Single-task Learning setting, the goal of variance-based regularization is to regularize the variance between the loss of training samples (Namkoong and Duchi, 2016; Duchi and Namkoong, 2019). While these variance-based regularization methods can improve the generalization ability of Single-task Learning models, they do not fit the Multi-task Learning setting. This paper thus first proposes a novel variance-based regularization method for Multi-task Learning to improve MTL models' generalization ability by regularizing the between-task loss variance.

3 Preliminaries

Consider a multi-task learning problem with T tasks over an input space $\mathcal{X}$ and a collection of task spaces $\{\mathcal{Y}^t\}_{t=1}^T$. For each task, we have a set of i.i.d. training samples $D_t = (\overline{X}_t, \overline{Y}_t)$ and $(\overline{X}_t, \overline{Y}_t) = \{x_i^t, y_i^t\}_{i=1}^{n_t}$, where n_t is the number of training samples of task t. In this paper, we focus on the neural network-based multi-task learning setting, in which the tasks are jointly learned by sharing some parameters (hidden layers).

Let $h(\cdot, \theta) : \{\mathcal{X}\}_{t=1}^T \rightarrow \{\mathcal{Y}^t\}_{t=1}^T$ be the multi-task learning model, where $\theta \in \Theta$ is the vector of the model parameters. $\theta = (\theta^{sh}, \theta^1, ..., \theta^T)$ consists of θ^{sh} (the parameters shared between tasks) and θ^t (the task-specific parameters). We denote $h^t(\cdot, \theta^{sh}, \theta^t) : \mathcal{X} \rightarrow \mathcal{Y}^t$ as the task-specific map. The task-specific loss function is denoted as $l^t(\cdot, \cdot) : \mathcal{Y}^t \times \mathcal{Y}^t \rightarrow [0,1]^T$. The empirical loss of the task t is defined as $\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t) = \frac{1}{n_t} \sum_{i=1}^{n_t} l^t(h(x_i^t, \theta^{sh}, \theta^t), y_i^t)$.

The transpose of the vector/matrix is represented by the superscript $\top$, and the logarithms to base e are denoted by log.

3.1 The Learning Objective of MTL

Under the Empirical Risk Minimization paradigm, multi-task learning aims to optimize the vector of task-specific empirical losses. The learning objective of multi-task learning is formulated as a vector optimization objective, as in equation (1).

$$\min_{\theta} \ (\hat{\mathcal{L}}^1(\theta^{sh}, \theta^1), ..., \hat{\mathcal{L}}^T(\theta^{sh}, \theta^T))^\top, \quad (1)$$

In order to optimize the learning objective, existing multi-task learning methods tend to adopt either global criterion optimization strategies (Liu et al., 2017; Kendall et al., 2018; Chen et al., 2018; Mao et al., 2020b) or multiple gradient descent strategies (Sener and Koltun, 2018; Lin et al., 2019; Debabrata Mahapatra, 2020). In this paper, we choose to adopt the typical linear-combination strategy, which can achieve proper Pareto Optimality (Miettinen, 2012) and is widely used in the multi-task text classification context (Liu et al., 2017; Yadav et al., 2018; Xiao et al., 2018). The linear-combination strategy is defined in (2):

$$\min_{\theta} \frac{1}{T} \sum_{t=1}^{T} \hat{\mathcal{L}}^t(\theta^{sh}, \theta^t), \quad (2)$$

3.2 Adversarial Multi-armed Bandit

Adversarial multi-armed bandit, a case in which a player and an adversary simultaneously address the trade-off between exploration and exploitation, is one of the fundamental multi-armed bandit problems (Bubeck and Cesa-Bianchi, 2012). In this paper, we consider the linear multi-armed bandit, which is a generalized adversarial multi-armed bandit. In our linear multi-armed bandit setting, the set of arms is a compact set $\mathcal{A} \in \mathbb{R}^T$. At each time step $k = 1, 2, ..., K$ the player chooses an arm from $\mathcal{A}$ while; simultaneously, the adversary chooses a loss vector from $[0,1]^T$. For linear multi-armed bandit, the Online Mirror Descent (OMD) algorithm is a powerful technology that can be used to achieve proper regret (Srebro et al., 2011).

3.3 Online Mirror Descent

The Online Mirror Descent (OMD) algorithm is a generalization of gradient descent for sequential de-

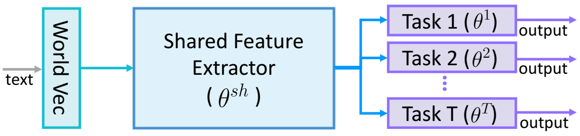

Figure 1: Illustration of the framework of hard parameter-sharing MTL models.

cision problems. Rather than taking gradient steps in the primal space, the mirror descent approach involves taking gradient steps in the dual space. The bijection $\nabla\Phi$ and its inverse $\nabla\Phi^*$ are used to map back and forth between primal and dual points. To obtain a good regret bound, Φ must be a Legendre function (Definition 1).

Assume that we update u^k with gradient g^k using OMD. The OMD algorithm consists of three steps: (1) select a Legendre function Φ; (2) perform a gradient descent step in the dual space $v^{k+1} = \nabla\Phi^*(\nabla\Phi(u^k) - \eta g^k)$, where Φ^* and $\nabla\Phi^*$ are as defined in Definition 2 and η is the step length; (3) project back to the primal space according to the Bregman divergence (Definition 3): $u^{k+1} = \arg\min_u D_\Phi(u, v^{k+1})$.

Definition 1 (Legendre Function)**.** *Let $\mathcal{O} \subset \mathbb{R}^T$ be an open convex set, and let $\overline{\mathcal{O}}$ be the closure of $\mathcal{O}$. A continuous function $\Phi : \overline{\mathcal{O}} \to \mathbb{R}$ is Legendre if:*

(i) Φ is strictly convex and admits continuous first partial derivatives on $\mathcal{O}$;

(ii) $\lim_{u \to \overline{\mathcal{O}}/\mathcal{O}} \parallel \nabla\Phi(u) \parallel = +\infty$.

Definition 2 (Fenchel Conjugate)**.** *The Fenchel conjugate Φ^* of Φ is $\Phi^*(u) = \sup_v\{\langle u, v\rangle + \Phi(v)\}$, and $\nabla\Phi^*(u) = \arg\max_v\{\langle u, v\rangle + \Phi(v)\}$.*

Definition 3 (Bregman Divergence)**.** *The Bregman divergence $D_\Phi : \overline{\mathcal{O}} \times \mathcal{O} \to \mathbb{R}$ associated with a Legendre function Φ is defined by $D_\Phi(u, v) = \Phi(u) - \Phi(v) - (u - v)^\top \nabla\Phi(v)$.*

3.4 Hard Parameter-sharing MTL Model

This paper adopts the most prevalent and efficient hard parameter-sharing MTL model (Kendall et al., 2018; Chen et al., 2018; Sener and Koltun, 2018; Mao et al., 2020b) to perform multi-task text classification. As shown in Figure 1, the hard parameter-sharing MTL model learns multiple related tasks simultaneously by sharing the hidden layers (feature extractor) across all tasks while retaining task-specific output layers for each task. In multi-task text classification, the feature extractor can be LSTM (Hochreiter and Schmidhuber, 1997), TextCNN (Kim, 2014), and so on. The task-specific layers are typically formulated by fully connected layers, ending with a softmax function.

4 Bandit-based Multi-task Learning

To avoid uncontrolled task variance, we need to develop a learning method that regularizes the task variance during training. Regularized Loss Minimization (RLM) is a learning method that jointly minimizes the empirical risk and a regularization function, and is thus a natural choice. While RLM is widely used in Single-task Learning, it cannot be directly used in Multi-task Learning to regularize the task variance. In this section, we propose a surrogate for RLM in MTL and accordingly develop a novel MTL method, namely BanditMTL.

4.1 Regularizing the Task Variance

RLM is a natural choice for regularizing the task variance. RLM for task-variance-regularized MTL can be formulated as in equation (3):

$$\min_\theta \frac{1}{T}\sum_{t=1}^{T}\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t) + \sqrt{\rho Var(\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t))}, \tag{3}$$

where $Var(\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t)) = \frac{1}{T}\sum_{t=1}^{T}(\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t) - \frac{1}{T}\sum_{t=1}^{T}\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t))^2$ is the empirical variance between the task-specific losses.

However, formulation (3) is generally non-convex and associated NP-hardness. To handle the non-convexity, we select a convex surrogate for (3) based on its equivalent formulation (4) (Ben-Tal et al., 2013; Bertsimas et al., 2018).

$$\sup_{p\in\mathcal{P}_{\rho,\mathbf{T}}} \frac{1}{T}\sum_{t=1}^{T} p_t\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t) = \frac{1}{T}\sum_{t=1}^{T}\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t) + \sqrt{\rho Var(\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t))} + o(T^{-\frac{1}{2}}), \tag{4}$$

where $\mathcal{P}_{\rho,T} := \{p \in \mathbb{R}^T : \sum_{t=1}^{T} p_t = 1, p_t \geq 0, \sum_{t=1}^{T} p_t \log(Tp_t) \leq \rho\}$.

$\sup_{p\in\mathcal{P}_{\rho,\mathbf{T}}} \frac{1}{T}\sum_{t=1}^{T} p_t\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t)$ is convex and can be used as a convex surrogate for (3). This paper proposes to perform task-variance-regularized multi-task-learning with the following learning objective:

$$\min_\theta \sup_{p\in\mathcal{P}_{\rho,\mathbf{T}}} \frac{1}{T}\sum_{t=1}^{T} p_t\hat{\mathcal{L}}^t(\theta^{sh}, \theta^t) \tag{5}$$

Optimizing (5) is equivalent to optimizing (3).

In the proposed learning objective (5), ρ is the regularization parameter that controls the trade-off between the mean empirical loss and the task variance. Experimental analysis on the influence of ρ is presented in Section 5.6. To learn an MTL model via learning objective (5), we formulate the learning problem as an adversarial multi-armed bandit problem in Section 4.2 and further propose the BanditMTL algorithm in Section 4.3.

4.2 Task-Variance-Regularized MTL as Adversarial Multi-armed Bandit

In deep multi-task learning, an MTL model is typically learnt by iteratively optimizing the learning objective. To iteratively optimize the proposed learning objective (5), we formulate it as an adversarial multi-armed bandit problem in which the player chooses an arm from $\mathcal{P}_{\rho,\mathbf{T}}$ and the adversary assigns a loss vector $\mathbf{L}(\theta) = (\hat{\mathcal{L}}^1(\theta^{sh}, \theta^1), ..., \hat{\mathcal{L}}^T(\theta^{sh}, \theta^T))^\top$ to each arm. In each learning iteration, the player chooses an arm from $\mathcal{P}_{\rho,\mathbf{T}}$ to increase the weighted sum loss, while the adversary aims to decrease the loss by updating the learning model. Moreover, both the player and the adversary aim to find a trade-off between exploration and exploitation to achieve proper regret.

When $l^t(\cdot,\cdot)$ is convex and Θ is compact, the adversarial multi-armed bandit problem can achieve a saddle point (θ^*, p^*) (Boyd and Vandenberghe, 2014). The saddle point satisfies $L_{\text{sup}}^p \leq p^{*\top}\mathbf{L}(\theta^*) \leq L_{\text{inf}}^\theta$, where $L_{\text{sup}}^p = \sup\{p^\top \mathbf{L}(\theta^*) | p \in \mathcal{P}_{\rho,\mathbf{T}}\}$ and $L_{\text{inf}}^\theta = \inf\{p^{*\top}\mathbf{L}(\theta) | \theta \in \Theta\}$.

To achieve a proper regret and saddle point, we adopts mirror gradient ascent for the player and mirror gradient descent for the adversary. The mirror gradient ascent-descent algorithm for MTL, namely BanditMTL, is proposed in the next section.

4.3 BanditMTL

In this paper, the task-variance-regularized multi-task learning is formulated as a linear adversarial multi-armed bandit problem. For a problem of this kind, mirror gradient descent (ascent) is a powerful technique for the adversary and the player to achieve proper regret (Bubeck and Cesa-Bianchi, 2012; Namkoong and Duchi, 2016). Moreover, based on the mirror gradient ascent-descent, we can reach the saddle point of the minimax optimization problem when the task-specific loss functions are convex and the parameter space Θ is compact (Boyd and Vandenberghe, 2014).

Algorithm 1: BanditMTL

```
Input: data $\{D_t\}_{t=1}^T$, the learning rate $\eta_p$ and $\eta_a$, the approximation parameter $\epsilon$.
Initialization: $p^1 = (\frac{1}{T}, \frac{1}{T}, ..., \frac{1}{T})^\top$, randomly initialize $\theta^1$.
for $k = 1$ to $K$ do
    Compute $\lambda$ with Algorithm 2.
    Update $p$: :
    $p_t^{k+1} = \frac{e^{\frac{1}{1+\lambda}(\log p_t^k + \eta_p \hat{\mathcal{L}}_t(\theta_{sh}^k, \theta_t^k))}}{\sum_{t=1}^T e^{\frac{1}{1+\lambda}(\log p_t^k + \eta_p \hat{\mathcal{L}}_t(\theta_{sh}^k, \theta_t^k))}}$
    Update $\theta$:
    $\theta^{k+1} = \theta^k - \eta_a \nabla_\theta \frac{1}{T} \sum_{t=1}^T p_t^k \hat{\mathcal{L}}^t(\theta^{sh}, \theta^t)$
end for
return $\theta^k$ with best validation performance.
```

Algorithm 2: Compute λ

```
Input: $p^k$, $\theta^k$, $\epsilon$, $\beta$.
Initialization: $\lambda_l = 0$, $\lambda_r = 0$.
if $f(0) \leq 0$ then
    return 0.
end if
while $f(\lambda_r) \geq 0$ do
    $\lambda_l = \lambda_r$.
    $\lambda_r = \lambda_l + \beta$.
end while
while $|f(\hat{\lambda})| > \epsilon$ do
    $\hat{\lambda} = \frac{\lambda_l + \lambda_r}{2}$.
    if $f(\hat{\lambda}) > 0$ then
        $\lambda_l = \hat{\lambda}$.
    else
        $\lambda_r = \hat{\lambda}$.
    end if
end while
return $\hat{\lambda}$.
```

In this paper, we propose a task-variance-regularized multi-task learning algorithm based on mirror gradient ascent-descent, dubbed BanditMTL. The proposed method is presented in algorithmic form in Algorithm 1. We assume that the training procedure has K learning iterations. In each learning iteration $1 \leq k < K$, the player and the adversary update via mirror gradient ascent and descent.

4.3.1 Mirror Gradient Ascent for the Player

For the player, considering the constraint in $\mathcal{P}_{\rho,T}$, we choose the Legendre function $\Phi_p(p) = \sum_{t=1}^T p_t \log p_t$. Based on the Legendre function, we propose the update rule of p in (6) (see the

Appendix for derivations of the update rule).

$$p_t^{k+1} = \frac{e^{\frac{1}{1+\lambda}(\log p_t^k + \eta_p \hat{\mathcal{L}}_t(\theta_{sh}^k, \theta_t^k))}}{\sum_{t=1}^{T} e^{\frac{1}{1+\lambda}(\log p_t^k + \eta_p \hat{\mathcal{L}}_t(\theta_{sh}^k, \theta_t^k))}} \qquad (6)$$

where η_p is the step size for the player. Moreover,λ is the solution of equation, where $f(\lambda)$ is defined in (7). $f(\lambda)$ is non-increasing and $\lambda \geq 0$.

$$\begin{aligned} f(\lambda) = & \frac{\sum_{t=1}^{T} (\log q_t) q_t^{\frac{1}{1+\lambda}}}{\sum_{t=1}^{T} (1+\lambda) q_t^{\frac{1}{1+\lambda}}} - \log \sum_{t=1}^{T} q_t^{\frac{1}{1+\lambda}} \\ & + \log T - \rho, \end{aligned} \qquad (7)$$

where $q_t = e^{(\log p_t^k + \eta_p \hat{\mathcal{L}}_t(\theta_{sh}^k, \theta_t^k))}$. To solve $f(\lambda) = 0$, we propose a bisection search-based algorithm, as outlined in Algorithm 2.

4.3.2 Mirror Gradient Descent for the Adversary

For the adversary, to simplify calculation, we choose the Legendre function $\Phi_\theta(\theta) = \frac{1}{2} \parallel \theta \parallel_2^2$. By using $\Phi_\theta(\theta)$, the update rule of mirror gradient descent (presented in (8)) is the same as that of same with the common gradient descent. (see the Appendix for derivations of the update rule).

$$\theta^{k+1} = \theta^k - \eta_a \nabla_\theta \frac{1}{T} \sum_{t=1}^{T} p_t^k \hat{\mathcal{L}}^t(\theta^{sh}, \theta^t), \qquad (8)$$

where η_a is the learning rate for the adversary.

5 Experiments

In this section, we perform experimental studies on sentiment analysis and topic classification respectively to evaluate the performance of our proposed BanditMTL and verify our theoretical analysis. The implementation is based on PyTorch (Paszke et al., 2019). The code is attached in the supplementary materials.

5.1 Datasets

Sentiment Analysis . We evaluate our algorithm on product reviews from Amazon. The dataset (Blitzer et al., 2007) contains product reviews from 14 domains, including books, DVDs, electronics, kitchen appliances and so on. We consider each domain as a binary classification task. Reviews with rating > 3 were labeled positive, those with rating < 3 were labeled negative, reviews with rating $= 3$ are discarded as the sentiments were ambiguous and hard to predict.

Topic Classification . We select 16 newsgroups from the 20 Newsgroup dataset, which is a collection of approximately 20,000 newsgroup documents that is partitioned (nearly) evenly across 20 different newsgroups, then formulate them into four 4-class classification tasks (as shown in Table 1) to evaluate the performance of our algorithm on topic classification.

Table 1: Data Allocation for Topic Classification Tasks.

TASKS	NEWSGROUPS
COMP	OS.MS-WINDOWS.MISC, SYS.MAC.HARDWARE, GRAPHICS, WINDOWS.X
REC	SPORT.BASEBALL, SPORT.HOCKEY AUTOS, MOTORCYCLES
SCI	CRYPT, ELECTRONICS, MED, SPACE
TALK	POLITICS.MIDEAST, RELIGION.MISC, POLITICS.MISC, POLITICS.GUNS

5.2 Baselines

We compare BanditMTL with following baselines.

Single-Task Learning: learning each task independently.

Uniform Scaling: learning the MTL model with learning objective (2), the uniformly weighted sum of task-specific empirical loss.

Uncertainty: using the uncertainty weighting method proposed by (Kendall et al., 2018).

GradNorm: using the gradient normalization method proposed by (Chen et al., 2018).

MGDA: using the MGDA-UB method proposed by (Sener and Koltun, 2018).

AdvMTL: using the adversarial Multi-task Learning method proposed by (Liu et al., 2017).

Tchebycheff: using the Tchebycheff procedure proposed by (Mao et al., 2020b).

5.3 Experimental Settings

We adopt the hard parameter-sharing MTL model shown in Fig. 1. The shared feature extractor is formulated via a TextCNN which is structured with three parallel convolutional layers with kernels size of 3, 5, 7 respectively. The task-specific module is formulated by means of one fully connected layer ending with a softmax function. To ensure consistency with the state-of-the-art multi-task classification methods (Liu et al., 2017; Mao et al., 2020b) and ensure fair comparison, we adopt Pre-trained

https://www.cs.jhu.edu/~mdredze/datasets/sentiment/

http://qwone.com/~jason/20Newsgroups/

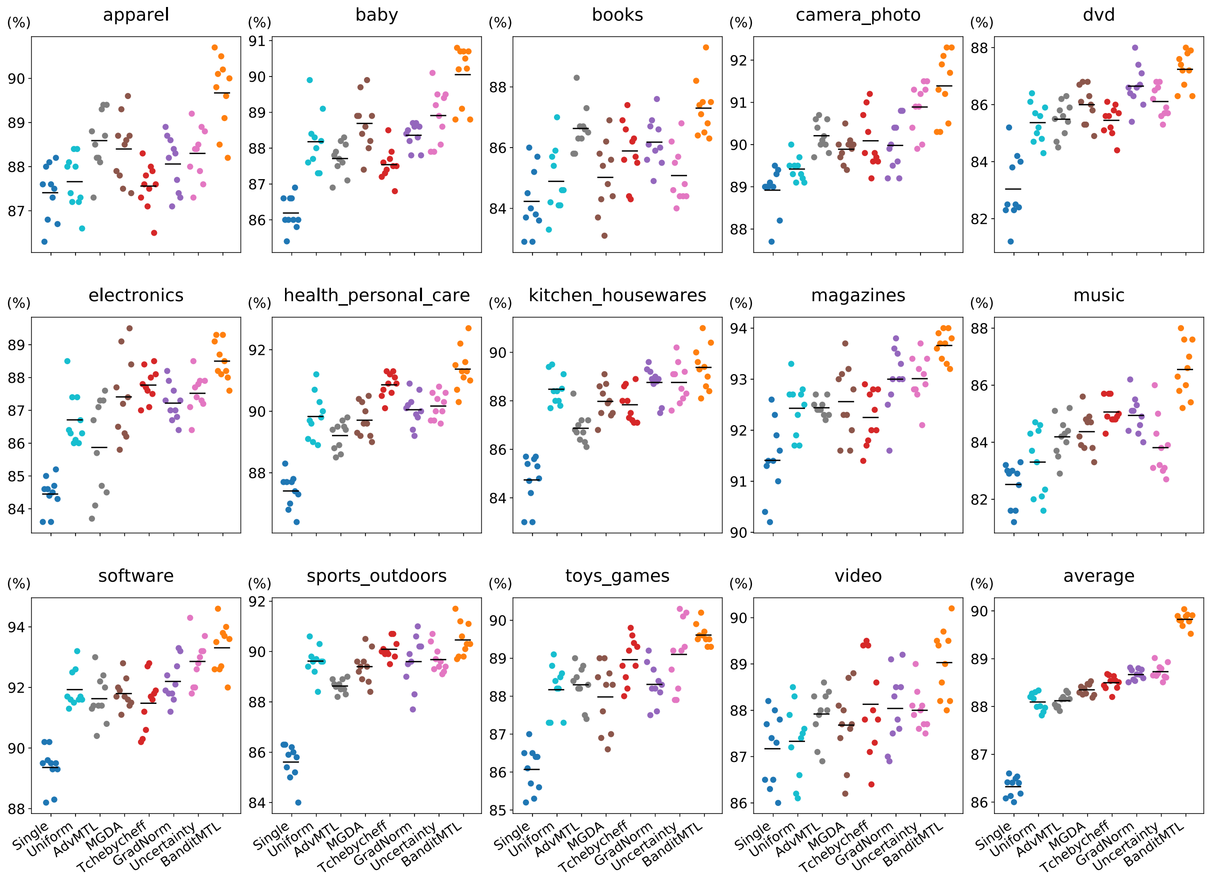

Figure 2: Classification accuracy of Single Task Learning, Uniform Scaling, AdvMTL, MGDA, Tchebycheff, GradNorm, Uncertainty, and BanditMTL on the sentiment analysis dataset. Each colored cluster illustrates the classification accuracy performance of a method over 10 runs. Our proposed BanditMTL outperforms all baselines in all tasks. ($\rho = 1.2$, $\eta_p = 0.5$)

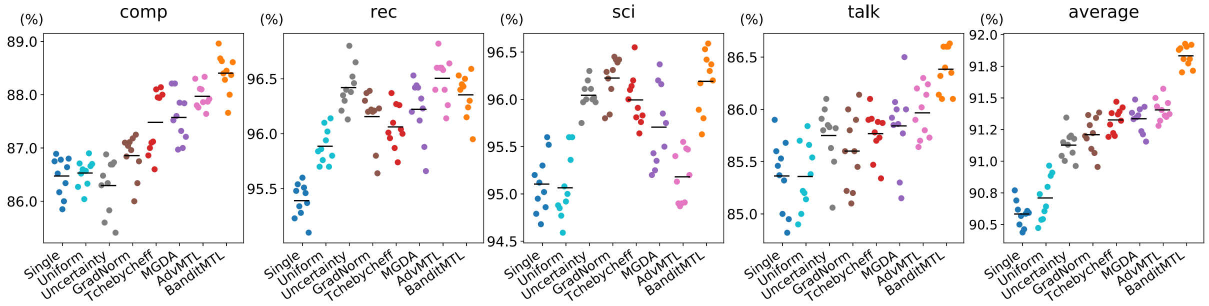

Figure 3: Classification accuracy of Single Task Learning, Uniform Scaling, Uncertainty, GradNorm, Tchebycheff, MGDA, AdvMTL, and BanditMTL on the topic classification dataset. Each colored cluster illustrates the classification accuracy performance of a method over 10 runs. Our proposed BanditMTL outperforms all baselines in all tasks except Rec. BanditMTL's average performance is also superior to that of all baselines. ($\rho = 1.2$, $\eta_p = 0.5$)

GloVe (Pennington et al., 2014) word embeddings in our experimental analysis.

We train the deep MTL network model in line with Algorithm 1. The learning rate for the adversary is $1e - 3$ for both sentiment analysis and topic classification. We use the Adam optimizer (Kingma and Ba, 2015) and train over 3000 epochs for both sentiment analysis and topic classification. The batch size is 256. We use dropout with a probability of 0.5 for all task-specific modules.

5.4 Classification Accuracy

We compare the proposed BanditMTL with the baselines and report the results over 10 runs by plotting the classification accuracy of each task for both sentiment analysis and topic classification. The results are shown in Fig. 2 and 3.

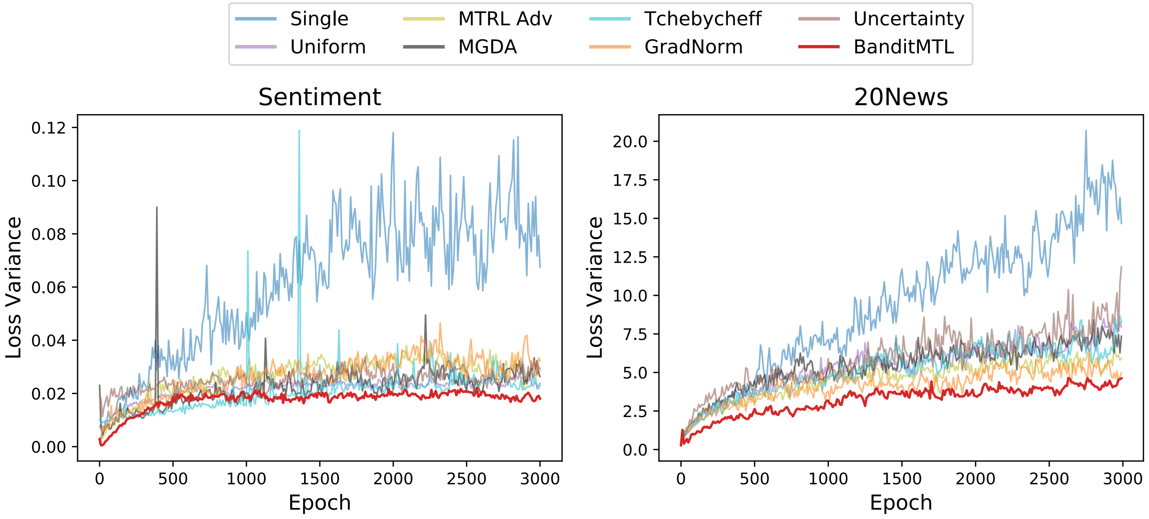

Figure 4: Evolution of task variance during training of baseline methods and BanditMTL on the sentiment analysis and topic classification datasets. $\rho = 1.2$, $\eta_p = 0.5$ for both sentiment analysis and topic classification.

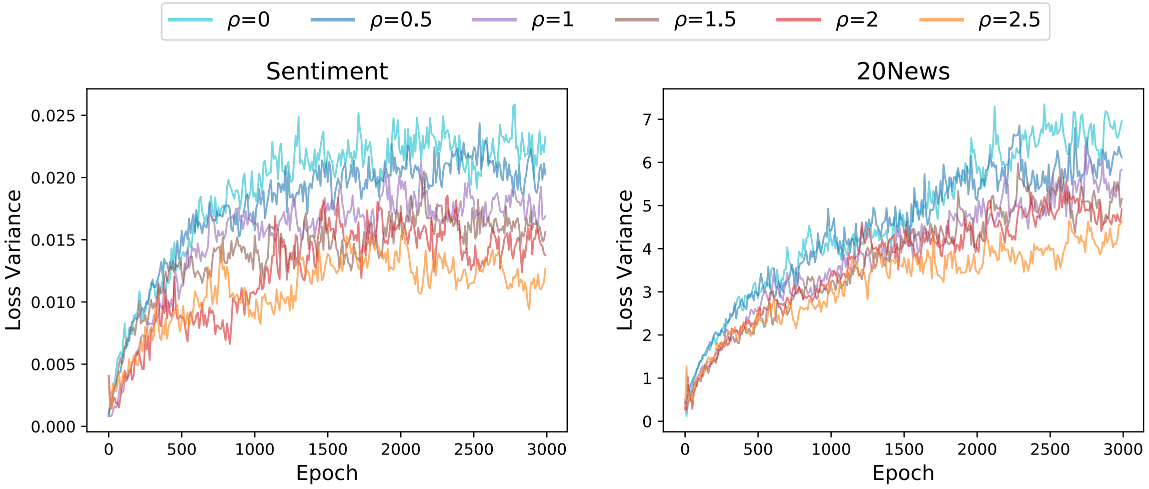

Figure 5: Evolution of task variance during training w.r.t different value of ρ on the sentiment analysis and topic classification datasets. $\eta_p = 0.5$ for both sentiment analysis and topic classification.

All experimental results show that our proposed BanditMTL significantly outperforms Uniform Scaling, which demonstrates that adopting task variance regularization can boost the performance of MTL models. Moreover, BanditMTL can be seen to outperform all baselines and achieve state-of-the-art performance.

5.5 Task Variance

In this section, we experimentally investigate how BanditMTL regularizes the task variance during training and compare the task variance of BanditMTL with the baselines. The results are plotted in Fig. 4. As the figure shows, all MTL methods have lower task variance than single task learning during training. Moreover, BanditMTL has lower task variance and smoother evolution during training than other MTL methods. After considering the results obtained in Section 5.4, we conclude that task variance has a significant impact on multi-task text classification performance.

5.6 Impact of ρ

In BanditMTL, ρ is the regularization parameter. In this section, we experimentally investigate the impact of ρ on task variance and average classification accuracy over the tasks of interest.

5.6.1 Impact on Variance

Fig. 5 plots how the task variance evolves during training w.r.t different values of ρ. The task variance decreases as ρ increases. It reveals that we can control the task variance by adjusting ρ.

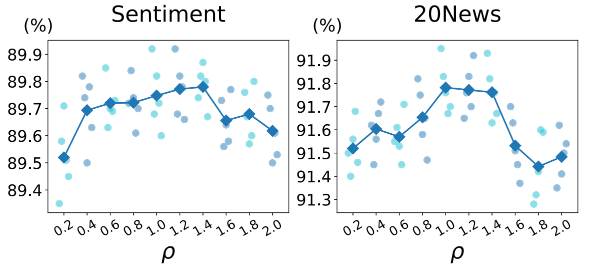

Figure 6: Task-average classification accuracy w.r.t different value of ρ. For each value of ρ, we report the results over five runs. $\eta_p = 0.5$.

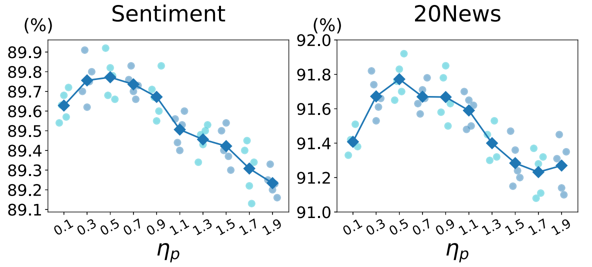

Figure 7: Changing of task-average classification accuracy w.r.t. increasing η_p. For each value of η_p, we report the results over five runs. $\rho = 1.2$ for both sentiment analysis and topic classification.

5.6.2 Impact on Average Accuracy

The changes in BanditMTL's average classification accuracy w.r.t different values of ρ is illustrated in Fig. 6. In this figure, as ρ increases, the average accuracy of BanditMTL first increases and then decreases. This reveals that ρ significantly impacts the performance of multi-task text classification. As ρ controls the trade-off between the empirical loss and the task variance, we can conclude that this trade-off significantly impacts the multi-task text classification performance. Thus, in the multi-task text classification, it is necessary for us to find a proper trade-off between the empirical loss and the task variance rather than focusing only on empirical loss. These results verify the necessary of task variance regularization.

5.7 Sensitivity Study on η_p

In BanditMTL, η_p is a hyper-parameter. To determine whether the performance of BanditMTL is sensitive to η_p, we conduct experiments on the classification performance of BanditMTL w.r.t different values of η_p. The results of these experiments are presented in Fig. 7. As the figure shows, the performance of our proposed method is not very sensitive to η_p when η_p is within the range of 0.3 to 0.9 for both sentiment analysis and topic classification. Setting η_p to between 0.3 and 0.9 can generally provide satisfactory results.

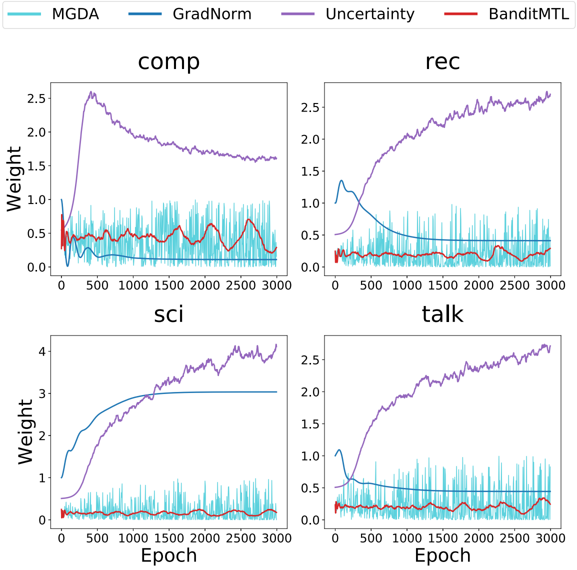

Figure 8: Comparison of task weight adaption processes between BanditMTL, Uncertainty, Gradnorm, and MGDA for topic classification. $\rho = 1.2$, $\eta_p = 0.5$.

5.8 Evolution of p_t

In this section, we observe the changes in p_t during training and compare these changes with the task weight adaption process of three weight adaptive MTL methods (i.e., Uncertainty, Gradnorm, and MGDA). The results for topic classification are reported in Fig. 9. Due to space limitations, the sentiment analysis results are presented in the appendix. From the results, we can see that the weight adaption process of BanditMTL is more stable than that of Uncertainty, Gradnorm, and MGDA.

6 Conclusion

This paper proposes a novel Multi-task Learning algorithm, dubbed BanditMTL. It fills the task variance regularization gap in the field of MTL and achieves state-of-the-art performance in real-world text classification applications. Moreover, our proposed BanditMTL is model-agnostic; thus, it could potentially be used in other natural language processing applications, such as Multi-task Named Entity Recognition.

Acknowledgements

This work is supported by the National Natural Science Foundation of China under Grants 61976161 and 61976162.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. 2006. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156.

Aharon Ben-Tal, Dick den Hertog, Anja De Waegenaere, Bertrand Melenberg, and Gijs Rennen. 2013. Robust solutions of optimization problems affected by uncertain probabilities. *Manag. Sci.*, 59(2):341–357.

Dimitris Bertsimas, Vishal Gupta, and Nathan Kallus. 2018. Robust sample average approximation. *Math. Program.*, 171(1-2):217–282.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*.

Stephen P. Boyd and Lieven Vandenberghe. 2014. *Convex Optimization*. Cambridge University Press.

Sébastien Bubeck and Nicolò Cesa-Bianchi. 2012. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Found. Trends Mach. Learn.*, 5(1):1–122.

Rich Caruana. 1993. Multitask learning: A knowledge-based source of inductive bias. In *ICML*.

Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. 2018. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *ICML*.

Vaibhav Rajan Debabrata Mahapatra. 2020. Multi-task learning with user preferences: Gradient descent with controlled ascent in pareto optimization. In *ICML*.

John C. Duchi and Hongseok Namkoong. 2019. Variance-based regularization with convex objectives. *J. Mach. Learn. Res.*, 20:68:1–68:55.

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

Alex Kendall, Yarin Gal, and Roberto Cipolla. 2018. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *CVPR*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Vladimir Koltchinskii et al. 2006. Local rademacher complexities and oracle inequalities in risk minimization. *The Annals of Statistics*, 34(6):2593–2656.

Reno Kriz, Marianna Apidianaki, and Chris Callison-Burch. 2020. Simple-qe: Better automatic quality estimation for text simplification. *Arxiv*.

Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qingfu Zhang, and Sam Kwong. 2019. Pareto multi-task learning. In *NIPS*.

Pengfei Liu, Xipeng Qiu, and Xuanjing Huang. 2017. Adversarial multi-task learning for text classification. In *ACL*.

Yuren Mao, Weiwei Liu, and Xuemin Lin. 2020a. Adaptive adversarial multi-task representation learning. In *ICML*.

Yuren Mao, Shuang Yun, Weiwei Liu, and Bo Du. 2020b. Tchebycheff procedure for multi-task text classification. In *ACL*.

Kaisa Miettinen. 2012. *Nonlinear multiobjective optimization*, volume 12. Springer Science & Business Media.

Hongseok Namkoong and John C. Duchi. 2016. Stochastic gradient methods for distributionally robust optimization with f-divergences. In *NeurIPS*.

Hongseok Namkoong and John C. Duchi. 2017. Variance-based regularization with convex objectives. In *NeurIPS*.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*.

Ozan Sener and Vladlen Koltun. 2018. Multi-task learning as multi-objective optimization. In *NeurIPS*.

Nati Srebro, Karthik Sridharan, and Ambuj Tewari. 2011. On the universality of online mirror descent. In *NeurIPS*.

Liqiang Xiao, Honglun Zhang, and Wenqing Chen. 2018. Gated multi-task network for text classification. In *NAACL-HLT*.

Shweta Yadav, Asif Ekbal, Sriparna Saha, Pushpak Bhattacharyya, and Amit P. Sheth. 2018. Multi-task learning framework for mining crowd intelligence towards clinical treatment. In *NAACL*.

Appendix

1 Derivations of the Update Rule for the Player

Assume the mirror gradient ascent step in the dual space is q^{k+1} w.r.t the $k+1^{th}$ learning iteration. Then, the q^{k+1} can be obtained as the follows.

According to the gradient descent step,

$$\nabla\Phi_p(q^{k+1}) = \nabla\Phi_p(p^k) + \eta_p \mathbf{L}(\theta^k). \tag{9}$$

For each task, the t-th element of $\nabla\Phi_p(q^{k+1})$,

$$\nabla\Phi_p(q_t^{k+1}) = 1 + \log q_t^{k+1}. \tag{10}$$

Combining (9) and (10), we have

$$q_t^{k+1} = e^{(\nabla\Phi_p(p_t^k)+\eta_p\hat{\mathcal{L}}_t(\theta_{sh}^k,\theta_t^k))-1)}. \tag{11}$$

To map back to the primal space, we need to solve optimization objective (12).

$$p^{k+1} = \arg\min_{p\in\mathcal{P}_{\rho,\mathbf{T}}} D_{\Phi_p}(p, q^{k+1}), \tag{12}$$

The Lagrangian for the optimization problem (12) is:

$$\begin{aligned}\mathcal{L}(p^{k+1},\alpha,\lambda) &= \sum_{t=1}^{T} p_t^{k+1}\log\frac{p_t^{k+1}}{q_t^{k+1}} \\ &- \alpha(\sum_{t=1}^{T} p_t^{k+1} - 1) - \lambda(\rho - \sum_{t=1}^{T} p_t^{k+1}\log p_t^{k+1} T).\end{aligned} \tag{13}$$

The partial derivative w.r.t p_t is:

$$\begin{aligned}\nabla_{p_t^{k+1}}\mathcal{L}(p^{k+1},\alpha,\lambda) =&(1+\lambda)\log p_t^{k+1} - \log q_t^{k+1} \\ &- \alpha + \lambda\log T + 1 + \lambda.\end{aligned} \tag{14}$$

Using the first order conditions w.r.t p_t^{k+1} ($\nabla_{p_t^{k+1}}\mathcal{L}(p^{k+1},\alpha,\lambda) = 0$), we have

$$p_t^{k+1} = (q_t^{k+1})^{\frac{1}{1+\lambda}} T^{-\frac{\lambda}{1+\lambda}} \exp(\frac{\alpha}{1+\lambda} - 1). \tag{15}$$

Combining with $\sum_{t=1}^{T} p_t^{k+1} = 1$, we have

$$p_t^{k+1} = (q_t^{k+1})^{\frac{1}{1+\lambda}} / (\sum_{t=1}^{T}(q_t^{k+1})^{\frac{1}{1+\lambda}}). \tag{16}$$

Plugging this back into the Lagrangian, we have

$$\begin{aligned}\mathcal{L}(\lambda) =& \min_{\alpha}\max_{p^{k+1}\in\mathcal{P}_{\rho,\mathbf{T}}} \mathcal{L}(p^{k+1},\alpha,\lambda) \\ =&\lambda(\log T - \rho) - (1+\lambda)\log\sum_{t=1}^{T}(q_t^{k+1})^{\frac{1}{1+\lambda}}.\end{aligned} \tag{17}$$

Taking derivatives, we have

$$\begin{aligned}\frac{d}{d\lambda}\mathcal{L}(\lambda) =& \log T - \rho - \log\sum_{t=1}^{T}(q_t^{k+1})^{\frac{1}{1+\lambda}} \\ &- \frac{\sum_{t=1}^{T} log(q_t^{k+1})(q_t^{k+1})^{\frac{1}{1+\lambda}}}{(1+\lambda)\sum_{t=1}^{T}(q_t^{k+1})^{\frac{1}{1+\lambda}}}.\end{aligned} \tag{18}$$

Combining (11) and (16), we have

$$p_t^{k+1} = \frac{e^{\frac{1}{1+\lambda}(\log p_t^k + \eta_p\hat{\mathcal{L}}_t(\theta_{sh}^k,\theta_t^k))}}{\sum_{t=1}^{T} e^{\frac{1}{1+\lambda}(\log p_t^k + \eta_p\hat{\mathcal{L}}_t(\theta_{sh}^k,\theta_t^k))}} \tag{19}$$

where λ is obtained by solving the equation $\frac{d}{d\lambda}\mathcal{L}(\lambda) = 0$, which is the necessary condition to optimize the Lagrangian function.

2 Derivations of the Update Rule for the Adversary

Assume the mirror gradient descent step in the dual space is γ^{k+1} w.r.t the $k+1^{th}$ learning iteration. Then, the γ^{k+1} can be obtained as the follows.

$$\nabla\Phi_\theta(\gamma^{k+1}) = \nabla\Phi_\theta(\theta^k) - \eta_a\frac{1}{T}\sum_{t=1}^{T} p_t^k\hat{\mathcal{L}}^t(\theta^{sh},\theta^t) \tag{20}$$

For $\Phi_\theta(\theta) = \frac{1}{2} \parallel \theta \parallel_2^2$, we have $\nabla\Phi_\theta(\gamma^{k+1}) = \gamma^{k+1}$ and $\nabla\Phi_\theta(\theta^k) = \theta^k$. Thus,

$$\gamma^{k+1} = \theta^k - \eta_a\frac{1}{T}\sum_{t=1}^{T} p_t^k\hat{\mathcal{L}}^t(\theta^{sh},\theta^t). \tag{21}$$

Moreover, it is obvious that

$$\arg\min D_{\Phi_\theta}(\Phi_\theta, \gamma^{k+1}) = \gamma^{k+1}. \tag{22}$$

Then,

$$\theta^{k+1} = \theta^k - \eta_a\frac{1}{T}\sum_{t=1}^{T} p_t^k\hat{\mathcal{L}}^t(\theta^{sh},\theta^t). \tag{23}$$

which means that the update rule of the mirror gradient descent is same with the vanilla gradient descent when Legendre function $\Phi_\theta(\theta) = \frac{1}{2} \parallel \theta \parallel_2^2$ is adopted.

3 Weight Adaption Process for Sentiment Analysis

The results of the change of p_t during a training banditMTL model are shown in Fig. 9. Comparing it with the task weights adaption process of three weight adaptive MTL methods (i.e., Uncertainty, Gradnorm, MGDA), we can see that the weights adaption process of banditMTL is more stable.

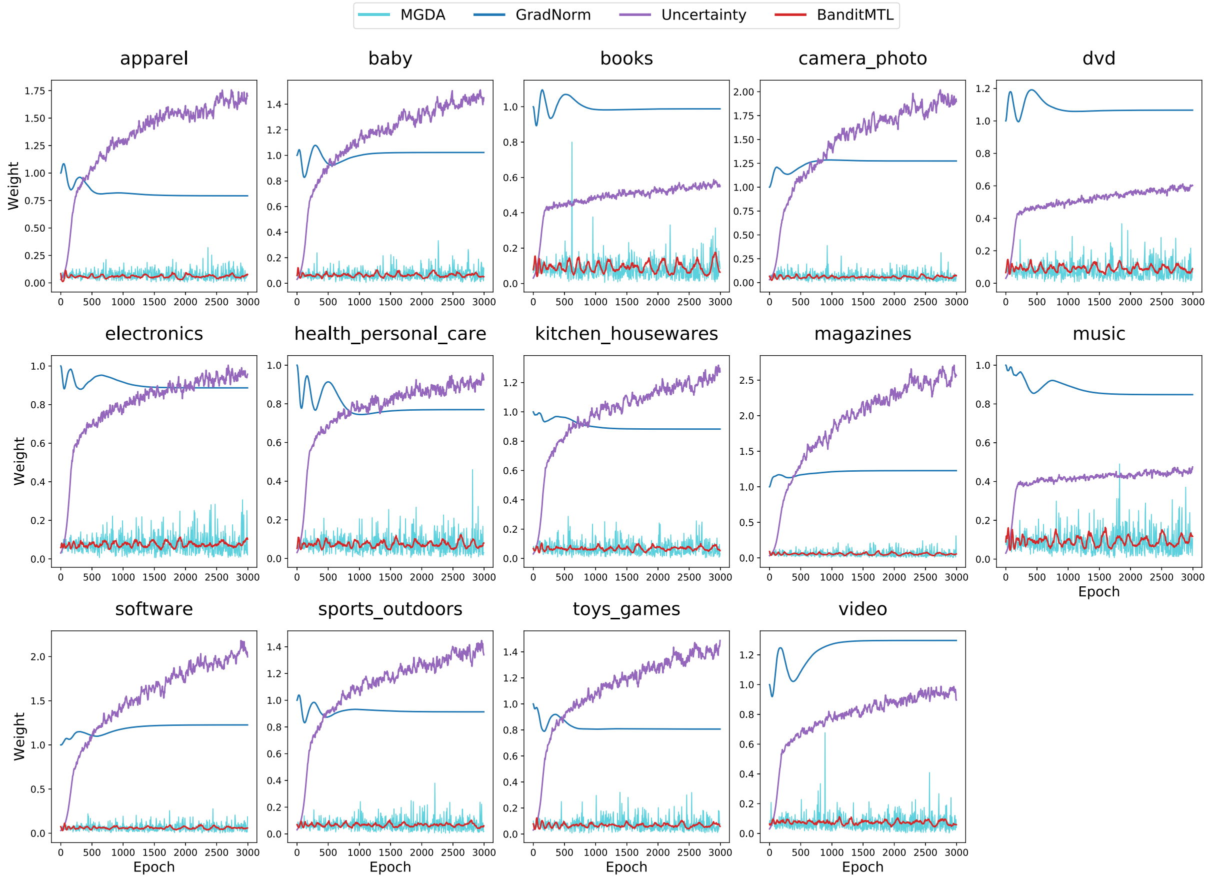

Figure 9: Comparison of task weight adaption processes between BanditMTL, Uncertainty, Gradnorm, and MGDA for sentiment analysis. $\rho = 1.2$, $\eta_p = 0.5$.

4 Hardware Specification and Environment

Our experiments are conducted on a Ubuntu 64-Bit Linux workstation, having 10-core Intel Xeon Silver CPU (2.20 GHz) and Nvidia GeForce RTX 2080 Ti GPUs with 11GB graphics memory.