

Learning Real-World Label Noise from Multiple Labellings: A Multi-Task Learning Approach

JIAMING (JAZLYN) LIN

SID: 470345744

Supervisor: Dr. Tongliang Liu

This thesis is submitted in partial fulfillment of
the requirements for the degree of
Bachelor of Advanced Studies (Honours)

School of Computer Science
The University of Sydney
Australia

25 June 2022



THE UNIVERSITY OF
SYDNEY

Student Plagiarism: Compliance Statement

I certify that:

I have read and understood the University of Sydney Student Plagiarism: Coursework Policy and Procedure;

I understand that failure to comply with the Student Plagiarism: Coursework Policy and Procedure can lead to the University commencing proceedings against me for potential student misconduct under Chapter 8 of the University of Sydney By-Law 1999 (as amended);

This Work is substantially my own, and to the extent that any part of this Work is not my own I have indicated that it is not my own by Acknowledging the Source of that part or those parts of the Work.

Name: Jiaming (Jazlyn) Lin

Signature:



Date: June 25, 2022

Abstract

The performance of deep neural networks relies heavily on accurately labeled datasets. However, in practice, datasets are noisy, resulting in degraded performance of classifiers. In the crowdsourcing literature, the most commonly applied technique to reduce noise is to collect multiple labelings of the same dataset, aggregate them into a single, less noisy labelling and finally discard all the original labellings. Meanwhile, a standard approach in the label-noise learning community is to explicitly model the label noise generation process as a transition matrix, through which a statistically-consistent clean classifier can be inferred. Recent work takes advantage of both approaches by learning a transition matrix from an aggregated set of labels, resulting in improved clean classifier performance compared to each approach applied in isolation. However, since aggregation discards the original labellings, potentially valuable information from the original labels fails to be exploited when learning the transition matrix.

In this thesis, we propose the Multi-Task Learning Transition Matrix (MTL-T) method, a novel multi-task learning approach aiming to estimate each transition matrix more effectively. By identifying the potential task relatedness between transition matrices, we adapt appropriate multi-task learning techniques to the label noise learning setting and exploit such relationship through knowledge transfer. Our experimental results demonstrate that in CIFAR-10N, a real-world noisy dataset with multiple labellings, MTL-T successfully improves the estimation of each transition matrix, leading to better downstream clean classifiers compared to those learnt without knowledge transfer.

Acknowledgements

I would like to thank my supervisor, Dr Tongliang Liu, for accepting me as an honours student in the first place. Through this project, I have gained a greater appreciation towards research and am extremely grateful for the learning opportunity I have been given. I would also like to thank PhD candidate Songhua Wu for all his feedback along the way.

CONTENTS

Student Plagiarism: Compliance Statement	ii
Abstract	iii
Acknowledgements	iv
List of Figures	vii
List of Tables	ix
Chapter 1 Introduction	1
1.1 Contributions	6
1.2 Thesis Organisation	7
Chapter 2 Literature Review	8
2.1 Crowdsourcing	8
2.1.1 Background	8
2.1.2 Problem Definition	8
2.1.3 Existing Work	9
2.2 Label-Noise Learning	13
2.2.1 Background	13
2.2.2 Problem Definition	14
2.2.3 Existing Work	16
2.3 Multi-task Learning	24
2.3.1 Background	24
2.3.2 Problem Definition	25
2.3.3 Existing Work	26
Chapter 3 Multi-Task Learning for Label Noise Learning	34
3.1 Proposed Solution	34
3.2 Method Selection	35

3.3	Method Formulation	36
	Stage 1: Estimate k transition matrices with knowledge transfer	36
	Stage 2: Learn k clean classifiers	38
Chapter 4	Experiments and Results	39
4.1	Method Overview	39
4.2	Experimental Questions	41
4.3	Experimental Setup	41
4.3.1	Dataset	41
4.3.2	Evaluation Metrics	42
4.3.3	Implementation Details	43
4.4	Ablation Study	44
4.5	Results and Discussion	45
4.5.1	Occurrence of Knowledge Transfer (Q1)	45
4.5.2	Impact of Knowledge Transfer in Transition Matrix Estimation (Q2)	46
4.5.3	Compare AGG-Classifer and MTL-T (Q3)	47
4.5.4	Compare AGG-Label and MTL-T (Q4)	47
Chapter 5	Conclusion	49
Chapter 6	Limitations and Future Work	50
	Bibliography	52

List of Figures

1.1	Typical crowdsourcing approach in handling label noise	2
1.2	Typical label-noise learning approach in handling label noise	3
1.3	Recent work (Wei et al., 2021)’s approach in handling label noise	4
1.4	A naïve approach in handling label noise	5
1.5	The proposed MTL-T approach in handling label noise	6
2.1	The general procedure of most existing truth inference algorithms	9
2.2	Comparison of different error flows: MentorNet, Co-teaching and Co-teaching+	17
2.3	Approximate the instance-dependent transition matrix by a weighted combination of part-dependent transition matrices (Xia et al., 2020)	21
2.4	Comparison of clean, noisy and Bayes class posteriors (Yang et al., 2022)	22
2.5	Estimate Bayes label transition matrix using collected distilled examples	23
2.6	Training a Bayes label classification network by forward corrected	24
2.7	The multi-layer feedforward neural network (Caruana, 1997; Yang et al., 2020)	26
2.8	The architecture of the cross-stitch network (Misra et al., 2016)	28
2.9	The architecture of Multilinear relationship network (MRN) (Long et al., 2017)	32
3.1	MTL-T, the proposed solution	34
3.2	adapted Bayes-Label transition network	36
4.1	AGG-Label, the first group of baselines for the MTL-T approach	39
4.2	AGG-Classifier, the second baseline for the MTL-T approach	40
4.3	MTL-T, the proposed multi-task learning approach	40
4.4	Prediction similarity computation	42
4.5	Sensitivity of $f_1(X)$, $f_2(X)$, $f_3(X)$ ’s classification accuracy towards number of linear layers	44

4.6	Prediction similarity without and with knowledge transfer
-----	---

45

List of Tables

2.1	Representative truth inference algorithms for single-choice tasks	10
4.1	Test classification accuracy of the individual clean classifiers $f_1(X)$, $f_2(X)$, $f_3(X)$ with and without knowledge transfer	46
4.2	Test classification accuracy of the aggregated clean classifiers $f_{agg}(X)$ with and without knowledge transfer	47
4.3	Test classification accuracy of the clean classifiers from AGG-Label and MTL-T	47

Introduction

Since the success of AlexNet (Krizhevsky et al., 2012) in the 2012 ImageNet challenge, deep learning models have become a pervasive choice for classification problems in domains such as image recognition (He et al., 2016a), speech recognition (Neumann and Vu, 2017) and sentiment analysis (Zhang and Wallace, 2015). The performance of these deep neural networks heavily relies on large-scale labelled datasets to learn from. Two widely adopted approaches are used to obtain labels for these datasets. The first approach is to distribute the labelling task to non-expert annotators in crowdsourcing platforms such as Amazon Mechanical Turk (Deng et al., 2009). However, instances can be mislabelled due to the limited knowledge of amateur annotations. The second approach relies on web-crawling to automatically assign labels extracted from the internet (Xiao et al., 2015). However, mislabelling also occurs due to the intrinsic unreliability of information gathered online. The presence of these mislabelled instances is known as *label noise*. Learning from these noisy datasets is challenging because deep neural networks are prone to overfitting the noisy data during training, producing biased predictions over unseen data (Arpit et al., 2017) and harming downstream applications. Unfortunately, obtaining large-scale, accurately labelled datasets is considered to be both too expensive and too time-consuming to be feasible in practice (Song et al., 2022).

To obtain datasets with less noise, a common practice in the crowdsourcing field is to hire annotators to produce multiple labellings of the same dataset, *aggregate* them into a single, less noisy labelling, and finally discard the original labellings, illustrated in Figure 1.1.

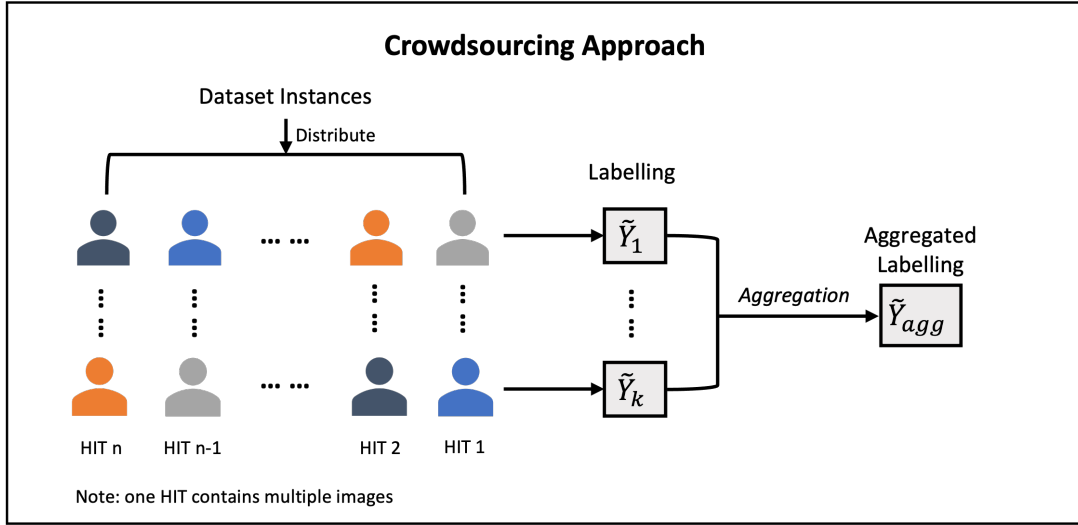


FIGURE 1.1. Typical crowdsourcing approach in handling label noise

The job of producing k labellings of the dataset is typically achieved by distributing the work in Human Intelligence Task (HIT) units (Zheng et al., 2017). Each HIT unit is a batch of multiple instances to be labelled with each batch assigned to k randomly selected annotators. Once all HIT units are labelled, k noisy labellings are produced, which are then aggregated into a single labelling with less label noise. The entire process is shown in Figure 1.1.

The most representative label aggregation algorithms employed in the literature include majority voting (MV), ZenCrowd (ZC) (Demartini et al., 2012), GLAD (Whitehill et al., 2009), CATD (Li et al., 2014a), PM (Li et al., 2014b), David & Skene (D&S) (Dawid and Skene, 1979), BCC (Kim and Ghahramani, 2012) and LFC (Raykar et al., 2010). These algorithms differ in their specific modelling of labelling task difficulty and annotator quality, but ultimately share the goal of producing an aggregated labelling that is less noisy. Even simple aggregation techniques such as majority voting (MV) are highly effective at reducing label noise. For instance, given 3 labellings of a real-world dataset, majority voting (MV) manages to reduce the label noise rate from 17% to 9% (Wei et al., 2021).

Unlike the crowdsourcing community which relies on aggregation techniques, the label noise learning community has adopted a different philosophy toward dealing with label noise. Instead of reducing the label noise in the dataset, techniques seek to learn *with the noise*. The goal is to learn a classifier with only noisy data that is able to accurately predict clean labels for unseen instances. Such a classifier is known as a *clean classifier*.

Techniques to learn a clean classifier can be broadly categorised into two main groups, *heuristic* methods and *statistically consistent* methods (Han et al., 2020). Heuristic methods yield classifiers with strong empirical performance yet lack strong theoretical guarantees. By comparison, statistically consistent techniques guarantee that given enough noisy data, the learned clean classifier over noisy data can converge to the optimal clean classifier over clean data.

All statistically consistent methods rely on the concept of a *transition matrix* which models the label noise generation process, i.e. the mapping from clean labels to noisy labels for any given instance. When accurately estimated, the transition matrix creates a firm bridge between the clean and noisy label spaces, ensuring convergence of the learned clean classifier to the optimal clean classifier. Hence, the task of the transition matrix is to map clean labels to noisy labels for a given instance.

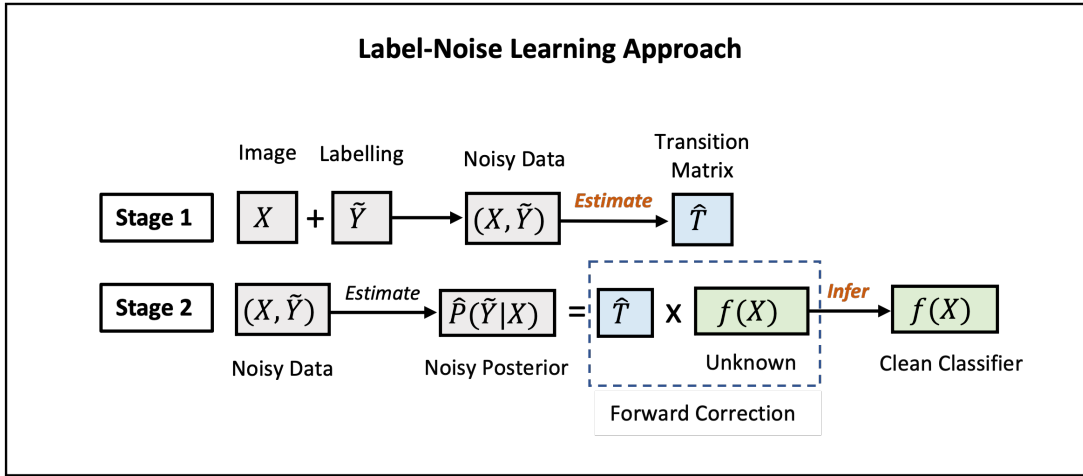


FIGURE 1.2. Typical label-noise learning approach in handling label noise

To learn the clean classifier using the transition matrix, statistically consistent methods commonly rely on a two stage training procedure, shown in Figure 1.2. Stage 1 estimates the transition matrix from the noisy label data. Stage 2 then applies a procedure called forward correction (Patrini et al., 2017) to learn the clean classifier using the estimated transition matrix.

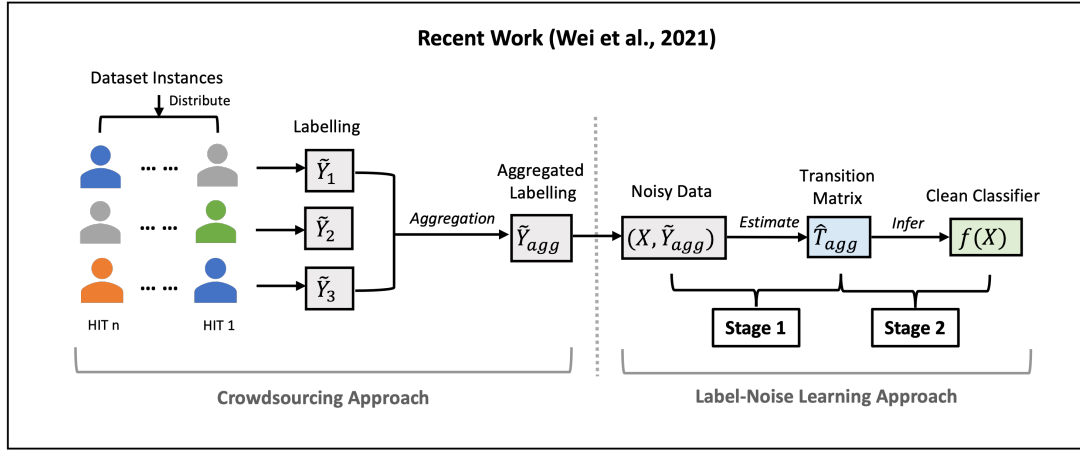


FIGURE 1.3. Recent work (Wei et al., 2021)’s approach in handling label noise

Recent work applies techniques from the crowdsourcing community in tandem with techniques from the label noise learning community to achieve improved classification accuracy than each one applied in isolation (Wei et al., 2021). The entire approach is showcased in Figure 1.3. First, three labellings of the dataset are collected from amateur annotators for the CIFAR-10 dataset. Next, the crowdsourced labellings are aggregated into a single set of labels of the dataset with less label noise, discarding the original three labellings of the dataset. Finally, label noise learning is performed using the aggregated labels to learn a clean classifier.

The results of the combined approach represent a significant improvement in accuracy. For example, when applying Forward-T as the label noise learning approach, learning from the aggregated set of labels results in a 1.2% accuracy uplift (from 87.04% to 88.24%) compared to the best result achieved when learning from any of the individual sets of labels. However, there is a fundamental limitation to this combined approach. Specifically, since aggregation discards the original set of labels, potentially valuable information from the original labels fails to be exploited when learning the transition matrix.

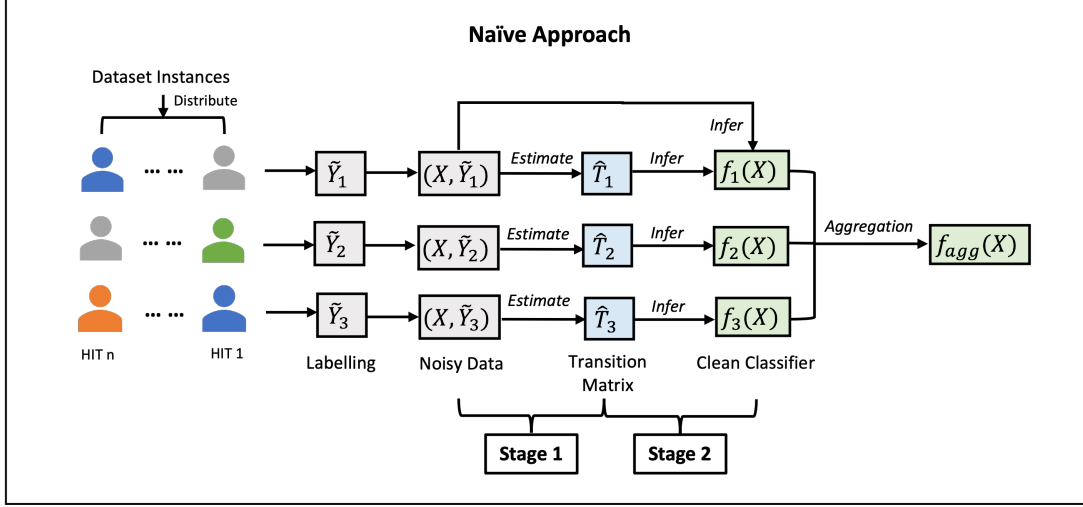


FIGURE 1.4. A naïve approach in handling label noise

A naïve approach to mitigate this information loss is illustrated in Figure 1.4. Instead of aggregating the individual labels, a transition matrix is learned for each labelling in Stage 1. Then in Stage 2, a clean classifier is learned for each transition matrix. Finally, the predicted labels for each clean classifier are then aggregated to produce a single predicted label for a given instance.

The approach illustrated above is known as *single-task learning (STL)* in the multi-task learning community since each transition matrix, i.e., task to map from clean labels to noisy labels, is learned in isolation. A well-known technique that improves upon single-task learning is called *multi-task learning (MTL)*, where related tasks are learned simultaneously, improving the performance of each task by exploiting the shared knowledge from other related tasks. However, MTL only facilitates improved task performance when a subset of tasks are *related but not identical* (Yang et al., 2020).

Given a labelling of the dataset, the transition matrix that models it can also be interpreted as the modelling of annotator’s labelling pattern (Chen et al., 2020). Under this interpretation, the task of each transition matrix, i.e. to map clean labels to the corresponding noisy labels, should be similar for two reasons. Firstly, annotators are humans and they tend to have common human labelling bias based on human intuition. For example, the labelling pattern of one CIFAR-10N labelling, visualised as a class-conditional transition matrix, reveals that mislabelling commonly occurs within one label category such as different animals, or between confusing labels such as bird and plane. Secondly, each labelling is produced by many randomly selected annotators rotating between different labellings, resulting in similar annotator patterns to emerge in each labelling. However, labelling patterns are similar but *not identical*

as annotators for each labelling are not exactly the same. Therefore, according to the definition of MTL, we can model each transition matrix as a task and exploit the shared knowledge through some MTL technique, potentially improve the estimation of transition matrices and clean classifier accuracy.

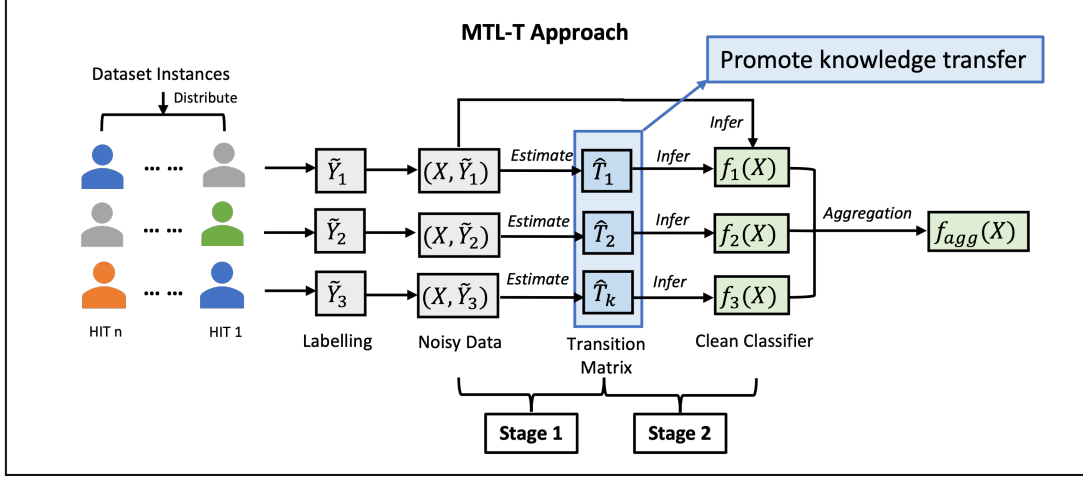


FIGURE 1.5. The proposed MTL-T approach in handling label noise

In this thesis, we propose the Multi-Task Learning Transition Matrix (MTL-T) method, a novel multi-task learning approach aiming to estimate each transition matrix more effectively. The high-level overview of MTL-T is showcased in Figure 1.5. For Stage 1 specifically, we learn all transition matrices simultaneously in a single Bayes Label transition network (Yang et al., 2022) with multiple task-specific linear layers modelling each transition matrix. For each linear layer, a tensor normal prior is imposed on the layer's parameter tensor with an associated task covariance matrix learning the layer-specific task relatedness. Knowledge transfer is then conducted accordingly based on the learned relationship. For each linear layer, if tasks are found to be related, knowledge transfer will be promoted across the tasks to improve the learning. Meanwhile, if tasks are learned to be unrelated, knowledge transfer will be disabled and layers will remain independent.

1.1 Contributions

This thesis presents the following contributions:

- We present MTL-T, a multi-task learning approach that aims to estimate each transition matrix more effectively in the presence of multiple labellings. The key idea of MTL-T is to model

each transition matrix as a task and enable knowledge transfer between transition matrices. To achieve this, we model all transition matrices in a Bayes transition matrix network (Yang et al., 2022) and impose tensor normal priors over the corresponding parameter tensor to promote knowledge transfer.

- We conduct a thorough evaluation of MTL-T over the CIFAR-10N dataset. Firstly, we confirm the occurrence of knowledge transfer through the change in prediction similarity. Next, we demonstrate improved estimation of each transition matrix through knowledge transfer, resulting in improved accuracy of each individual clean classifier (an accuracy uplift between 0.53% and 0.64%). Finally, we compare MTL-T with eight representative label-aggregation baselines in the crowdsourcing literature and find that MTL-T is at least 1.03% more accurate than any baseline.

1.2 Thesis Organisation

The rest of the thesis is structured as follows. In Chapter 2 we provide the background, problem definition and existing work of three relevant fields. Namely, label-noise learning, crowdsourcing and multi-task learning. In Chapter 3 we introduce the proposed multi-task approach for label-noise learning in the presence of multiple labellings. In Chapter 4 we describe the experiment setup, relevant experiment questions, and finally present the experiment result and discussion. In Chapter 5 we form our conclusions, discuss limitations and explore directions of future work.

Literature Review

2.1 Crowdsourcing

2.1.1 Background

The performance of deep learning models heavily relies on large yet accurately labelled datasets. Thanks to the wide deployment of crowdsourcing platforms such as Amazon Mechanical Turk (AMT) and Crowdflower, collecting large-scale annotated datasets through the crowd has become possible. ImageNet (Deng et al., 2009), a well-known image classification benchmark in the field of computer vision, outsources its million-level images to AMT distributed workers and enables the labelling work to be done in an inexpensive and time-efficient manner.

However, the openness of crowdsourcing leads to crowdsourcing workers with various backgrounds and motivations. Due to the labelling workers' limited labelling knowledge and some malicious behaviour of spammer workers, the labels collected through crowdsourcing are inevitably noisy. As mentioned in the background of label-noise learning (Section 2.2), deep learning models are known to be vulnerable to label noise, causing harm to downstream applications, particularly those in sensitive domains. To tackle this, most existing work in crowdsourcing adopt a very different philosophy to address these problems compared to those used in label-noise learning.

2.1.2 Problem Definition

Let $\mathcal{T} = \{t_1, \dots, t_n\}$ be a single-choice task set with n tasks. To cope with label noise, existing work often first employs *the redundancy strategy* by assigning each task t_i to multiple workers $\mathcal{W}^i = \{w_1, w_2, \dots, w_k\}$ and in the end obtains a collection of candidate answers $\mathcal{V}^i = \{v_i^{w_1}, \dots, v_i^{w_k}\}$. Next,

some *truth inference algorithm* can be applied to infer the true answer v_i^* (also known as truth) from all of the answers collected \mathcal{V}^i .

One of the most naive truth inference algorithms is Majority Voting (MV) (Zheng et al., 2017), which takes the majority answer from \mathcal{V}^i as the truth of task t_i . The biggest strength of majority voting is its simplicity. However, MV can suffer from noisy answers, given that it treats all workers equally regarding answer quality.

To overcome the weakness of MV, many truth inference algorithms explicitly model the worker quality. The worker quality is often estimated together with the truth in a solution framework illustrated as Figure 2.1. The motivation is that the answers produced by high-quality workers are more likely to be the truth. Conversely, workers with answers that align well with the truth are more likely to have high quality and should be assigned higher quality. Then by following this logic, the worker’s quality and task truth can be jointly and iteratively estimated until convergence.

Algorithm 1: Solution Framework

Input: workers’ answers V
Output: inferred truth v_i^* ($1 \leq i \leq n$), worker quality q^w ($w \in \mathcal{W}$)

```

1 Initialize all workers’ qualities ( $q^w$  for  $w \in \mathcal{W}$ );
2 while true do
3   // Step 1: Inferring the Truth
4   for  $1 \leq i \leq n$  do
5      $\perp$  Inferring the truth  $v_i^*$  based on  $V$  and  $\{q^w \mid w \in \mathcal{W}\}$ ;
6   // Step 2: Estimating Worker Quality
7   for  $w \in \mathcal{W}$  do
8      $\perp$  Estimating the quality  $q^w$  based on  $V$  and  $\{v_i^* \mid 1 \leq i \leq n\}$ ;
9   // Check for Convergence
10  if Converged then
11     $\perp$  break;
12 return  $v_i^*$  for  $1 \leq i \leq n$  and  $q^w$  for  $w \in \mathcal{W}$ ;

```

FIGURE 2.1. The general procedure of most existing truth inference algorithms

2.1.3 Existing Work

According to a recent survey in the crowdsourcing literature (Zheng et al., 2017), there are eight representative truth inference algorithms for the single-choice tasks we are interested in exploring. These algorithms can be further categorised based on whether and to what degree task difficulty and worker quality are modelled, according to Table 2.1.

Algorithms	Whether Model Task Difficulty	Whether Model Worker Quality
MV	\times	\times
ZC (Demartini et al., 2012)	\times	Worker Probability
GLAD (Whitehill et al., 2009)	\checkmark	Worker Probability
CATD (Li et al., 2014a)	\times	Worker Probability
PM (Li et al., 2014b)	\times	Worker Probability
D&S (Dawid and Skene, 1979)	\times	Confusion Matrix
BCC (Kim and Ghahramani, 2012)	\times	Confusion Matrix
LFC (Raykar et al., 2010)	\times	Confusion Matrix

TABLE 2.1. Representative truth inference algorithms for single-choice tasks

2.1.3.1 Algorithms with No Modelling

Majority voting (MV) is the field’s most straightforward yet most commonly adopted truth inference algorithm. The truth v_i^* is directly taken by the answer of the majority of workers or a random answer if there is no majority.

The biggest strength of MV is simplicity. Without any modelling but only direct computation, the MV algorithm is very efficient to implement in terms of time. However, such simplicity also contributes to its weakness. Given that no modelling of worker quality is involved, MV can suffer from low-quality answers, leading it to be less performant compared to those modelling counterparts.

2.1.3.2 Algorithms that Model Worker Quality as *Worker Probability*

In the literature, many algorithms model a worker’s answering quality q^w as a single value termed *worker probability*, which quantifies the worker’s capability to answer a task correctly. Some algorithms define q^w to be between 0 and 1, while others set it in a broader range such as $(-\infty, +\infty)$. Regardless of the range, a higher worker probability q^w always corresponds to a higher answer quality.

ZC ZenCrowd (ZC) (Demartini et al., 2012) adopts the general probabilistic graphic model (PGM) framework except for the priors and models the worker quality as a worker probability $q_w \in [0, 1]$. Then the probability that each worker w correctly answers a task t_i is formulated as

$$P(v_i^w | q^w, v_i^*) = q^{w \mathbb{1}_{v_i^w = v_i^*}} (1 - q^w)^{\mathbb{1}_{v_i^w \neq v_i^*}}. \quad (2.1)$$

Next, the maximum likelihood estimation (MLE) is applied to maximize for the the probability of worker answer co-occurrence, such as

$$\max_{q^w} P(V|q^w) = \frac{1}{2} \prod_{i=1}^n \sum_{z \in \{1, \dots, c\}} \prod_{w \in \mathcal{W}^i} P(v_i^w | q^w, v_i^* = z). \quad (2.2)$$

To optimise the non-convex objective function above, ZC utilises the Expectation-Minimisation (EM) algorithm to iteratively update the worker probability q^w and truth v_i^* until convergence.

GLAD The Generative model of Labels, Abilities, and Difficulties (GLAD) (Whitehill et al., 2009) can be considered an extension of ZC. It not only models the worker's quality as a worker probability but also models each task's difficulty as $d_i \in (0, \infty)$. Then the worker's answer can be modelled with worker quality and task difficulty as the following equation:

$$P(v_i^w = v_i^* | q^w, d_i), \quad (2.3)$$

which is then integrated into a similar MLE formulation shown in Equation 2.2.

CATD The Confidence-Aware Truth Discovery (CATD) method (Li et al., 2014a) was proposed to deal with the situation where the answers have a long-tail distribution. That is, some workers answer the majority of tasks while other workers answer only a few. Long-tail distribution of answers is problematic as it can degrade the overall estimation accuracy of worker quality. To address this, CATD introduces another metric called confidence which helps scale the worker probability according to the number of tasks answered. In this way, workers who answer correctly and widely will have higher quality score q^w .

PM PM (Li et al., 2014b) models each worker quality as worker probability and the objective function is formulated as

$$\min_{\{q^w\}, \{v_i^*\}} f(\{q^w\}, \{v_i^*\}) = \sum_{w \in \mathcal{W}} q^w \sum_{t_i \in \mathcal{T}^w} d(v_i^w, v_i^*) \quad (2.4)$$

where $\{q^w\}$ and $\{v_i^*\}$ denote the set of all worker quality and the set of all truth, respectively, and where d is a distance metric capturing worker's answer correctness.

2.1.3.3 Algorithms that Model Worker Quality as *Confusion Matrix*

Another common way is to model worker quality q^w as a matrix termed *confusion matrix*. Assuming that a single-choice task has C candidate choices, the worker quality confusion matrix is a $C \times C$ probability matrix. For any given task $t_i \in \mathcal{T}$, the entry $q_{j,k}^w$, formulated as below, denotes the probability worker w selects the k -th choice while the truth being the j -th choice.

$$q_{j,k}^w = P(v_i^w = k | v_i^* = j) \quad (2.5)$$

DS and its extensions The David & Skene algorithm (Dawid and Skene, 1979), also referred to as the D&S, was the first work proposed to model the worker’s quality explicitly (Jin et al., 2020). Specifically, the worker quality is captured by a confusion matrix. Similar to ZC’s optimisation process mentioned previously, D&S adopts the EM algorithms to iteratively update the confusion matrix as well as the truth.

There are many extensions to the D&S algorithm. For example, BCC (Kim and Ghahramani, 2012) tries to improve the parameter estimation method by replacing the MLE estimation with Maximum A Posteriori (MAP) estimation in the EM algorithm. LFC (Raykar et al., 2010) extends D&S by incorporating prior knowledge of each worker.

2.1.3.4 Algorithms Performance Comparison

Among all algorithms mentioned in Table 2.1, MV tends to have the least impressive performance given the same redundancy. As discussed previously, this is expected since MV does not perform any modelling of worker quality or task difficulty.

For algorithms that model worker quality, those that model worker quality as a confusion matrix are generally considered to be higher performing compared to those relying on worker probability (Zheng et al., 2017). The intuition is that a confusion matrix encodes more information about worker quality than worker probability as a single value. Empirical results agree with this hypothesis, as demonstrated through extensive experiments Zheng et al. (2017).

For algorithms that model task difficulty, i.e. GLAD, it has been shown that there is no extra performance uplift compared to those without modelling task difficulty (Zheng et al., 2017).

2.2 Label-Noise Learning

2.2.1 Background

Deep learning models have become a pervasive choice for classification problems in various domains, including image recognition (He et al., 2016a), speech recognition (Neumann and Vu, 2017) and sentiment analysis (Zhang and Wallace, 2015). The performance of these deep neural networks depends heavily on large yet accurate labelling of the datasets. However, according to a recent survey (Han et al., 2020), many popular datasets for deep learning tasks have incorrectly labelled instances. Informally, the presence of this mislabelling is known as *label noise*.

Due to the ways labelling is performed in practice, the existence of label noise appears to be natural and inevitable. There are two main labelling approaches. The first approach is to distribute the labelling task to non-expert annotators in crowdsourcing platforms such as Amazon Mechanical Turk (AMT) (Deng et al., 2009). For example, ImageNet (Deng et al., 2009) outsourced the labelling of millions of images to AMT distributed workers (Han et al., 2020), enabling the labelling job to be completed in an inexpensive and time-efficient manner. The second approach is to rely on machines to crawl labels from the internet. A good example is Clothing1M (Xiao et al., 2015), whose labels were extracted automatically using web-crawling to alleviate both the time and cost associated with manual annotation. However, due to the limited labelling knowledge of amateur AMT annotators as well as the intrinsic unreliability of online information, collected labellings tend to be inaccurate and label noise is inevitable. ImageNet is well-known to be noisy (Northcutt et al., 2021) and Clothing1M (Xiao et al., 2015) has noise rate as high as 38.5% (Song et al., 2022).

Label noise is problematic as deep neural networks are known to be vulnerable to it during training (Arpit et al., 2017). By producing biased predictions over unseen data, many downstream applications have degraded performance, leading to undesirable consequences, particularly for those in sensitive domains. Motivated by this, many learning algorithms that are robust to label noise have been developed, which will be discussed in Section 2.2.3.

2.2.2 Problem Definition

2.2.2.1 Empirical Risk Minimisation

Let \mathcal{D} be the data distribution of $(X, Y) \in \mathcal{X} \times \mathcal{Y}$ where X and $Y \in \{1, \dots, C\}$ are random variables of the data instance and the associated label respectively. A training set $S = \{(x_i, y_i)\}_{i=1}^N$ with N samples is generated from such distribution \mathcal{D} . In the ideal setting, the ultimate goal of a learning algorithm is to learn the Bayes classifier h^* , which can be obtained by minimising the expected risk $R_{\mathcal{D},1}$ measured using 0-1 loss function, such as

$$\begin{aligned} h^* &= \arg \min_h R_{\mathcal{D},1}(h) \\ &= \arg \min_h \mathbb{E}_{(X,Y) \sim \mathcal{D}} [1_{\{Y \neq h(X)\}}]. \end{aligned} \quad (2.6)$$

However, we cannot directly find h^* by optimising the above equation. The unknown distribution \mathcal{D} prevents the expectation calculation, and the non-convex 0-1 loss makes the objective function hard to optimise. A standard solution is to approximate the 0-1 loss with a convex surrogate loss ℓ (e.g., Cross-Entropy loss) that upper bounds it, and then optimise for the following empirical risk $\hat{R}_{\mathcal{D},\ell}$

$$\hat{R}_{\mathcal{D},\ell}(h) = \frac{1}{N} \sum_{i=1}^N \ell(h(x_i), y_i) \quad (2.7)$$

that is obtainable using the training samples. Given enough training samples, by the law of large numbers, we can ensure that such empirical risk will converge to the expected risk.

$$\frac{1}{N} \sum_{i=1}^N \ell(h(x_i), y_i) \xrightarrow{N \rightarrow \infty} \mathbb{E}_{(X,Y) \sim \mathcal{D}} [\ell(h(X), Y)]. \quad (2.8)$$

However, in a real-world setting, some labels of S are corrupted before observation and we only have access to a noisy version of the training set $\tilde{S} = \{(x_i, \tilde{y}_i)\}_{i=1}^N$, drawn from a noisy distribution $(X, \tilde{Y}) \sim \tilde{\mathcal{D}}$. To ensure the same convergence of Equation 2.8 under the noisy distribution $\tilde{\mathcal{D}}$, many robust learning algorithms develop a modified loss function $\tilde{\ell}$ corrected by a transition matrix (discussed in the following section).

$$\frac{1}{N} \sum_{i=1}^N \tilde{\ell}(h(x_i), \tilde{y}_i) \xrightarrow{N \rightarrow \infty} \mathbb{E}_{(X,Y) \sim \mathcal{D}} [\ell(h(X), Y)]. \quad (2.9)$$

This is often referred to as *statistical consistency*. By utilising this property and minimising $\hat{R}_{\tilde{\mathcal{D}},\tilde{\ell}}(h)$ the LHS of the above equation, we can then approximate h^* .

2.2.2.2 Transition Matrix

The random variables of clean and noisy label Y and \tilde{Y} can be related by a transition matrix $T(X)$, which explicitly models the noise generation process corrupting Y as \tilde{Y} . Each entry $T_{ij}(X = x)$, formulated as

$$T_{i,j}(x) = P(\tilde{Y} = j | Y = i, X = x), \quad (2.10)$$

represents the probability of the label for a given instance x to be flipped from the clean label i to the noisy label j .

As shown in Equation 2.11, the transition matrix bridges the gap between clean and noisy class posteriors, denoted as $P(\mathbf{Y}|X = x)$ and $P(\tilde{\mathbf{Y}}|X = x)$ respectively. If such a matrix is known, we can infer the clean class posterior from the noisy class posterior through the inverse transition matrix. A clean class posterior can also be called a clean classifier, which is used to infer the unknown clean labels of instances.

$$\begin{aligned} P(\tilde{\mathbf{Y}}|X = x) &= T(X = x)^\top P(\mathbf{Y}|X = x) \\ P(\mathbf{Y}|X = x) &= [T(X = x)^\top]^{-1} P(\tilde{\mathbf{Y}}|X = x) \end{aligned} \quad (2.11)$$

In reality, such a transition matrix is often unknown and needs to be estimated. An accurate estimation of the transition matrix plays a crucial role in inferring a clean classifier with statistical consistency, as described by Equation 2.9. The transition matrix can be intuitively interpreted as a bridge connecting the noisy and clean label space. With an accurately estimated transition matrix, i.e. a firm bridge, statistical consistency can then ensure the convergence of the classifier learnt over noisy data to the target optimal classifier over clean data. However, a poorly estimated transition matrix can break convergence, and the accuracy of learned clean classifiers will significantly degenerate.

Given the significance of an accurately estimated transition matrix, many label-noise learning algorithms in the literature focus on improving the transition matrix estimation accuracy.

Instance Dependence The formulation of transition matrix in Equation 2.10 is dependent on the instance x , which represents the most realistic modelling of label noise. Real-world label noise is naturally instance-dependent, as the mislabelling is supposed to be influenced by instance-specific issues such as poor image quality or confusing subjects. However, by introducing the dependency on the high-dimensional instance itself, the instance-dependent transition matrix becomes rather complicated to model and identify. Although a handful of work focuses on conquering this direction, most existing work in the literature has simplified Equation 2.10 as Equation 2.12, where the transition matrix is instance-independent.

$$T_{i,j} = P(\tilde{Y} = j | Y = i), \quad (2.12)$$

2.2.3 Existing Work

The existing label-noise learning algorithms can be categorised into two groups depending on whether a transition matrix is used. For methods that rely on a transition matrix, we can further split them into two subgroups of methods based on the noise dependency on the instance.

2.2.3.1 Heuristics

Many label-noise learning algorithms do not model the noise generation process as a transition matrix. Instead, they employ *heuristics*, with tricks such as identifying and learning from the possibly clean training examples to achieve competitive empirical performance.

MentorNet, first introduced by Jiang et al. (2018), adopts Curriculum Learning (CL) to identify clean instances. Specifically, two neural networks, namely MentorNet g_m and StudentNet g_s are involved. MentorNet g_m learns a data-driven curriculum G to help inform the StudentNet g_s which clean samples to focus on during training. Formally, the objective function is formulated as

$$\min_{\mathbf{w} \in \mathbb{R}^d, \mathbf{v} \in [0,1]^{n \times m}} \mathbb{F}(\mathbf{w}, \mathbf{v}) = \frac{1}{n} \sum_{i=1}^n \mathbf{v}_i^T \mathbf{L}(\mathbf{y}_i, g_s(\mathbf{x}_i, \mathbf{w})) + G(\mathbf{v}, \lambda) + \theta \|\mathbf{w}\|_2^2, \quad (2.13)$$

where the first term denotes the loss weighted by the latent weight vector \mathbf{v} , the second term is the MentorNet curriculum parameterised by \mathbf{v} and λ and the last regularisation term penalises the weight complexity of StudentNet g_s . The objective function is jointly optimised with an alternating update rule, where one of \mathbf{w} and \mathbf{v} is updated while the other is held fixed.

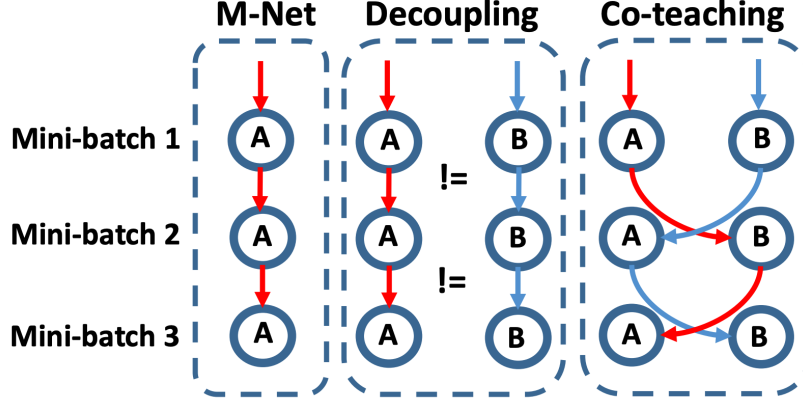


FIGURE 2.2. Comparison of different error flows: MentorNet, Co-teaching and Co-teaching+

However, as illustrated in Figure 2.2 (Yu et al., 2019), MentorNet’s network error from a previous mini-batch can directly be transferred back to itself in a later mini-batch, causing the error to be accumulated over time. To address this, Han et al. (2018) proposes a new learning paradigm called Co-teaching, where each of two networks samples its small-loss instances for the peer network to train over each mini-batch. Since both networks have different learning abilities, they can filter different error types for each other, mitigating the accumulated error issue to some extent.

Regardless, with the increase in training epochs, the two networks of Co-teaching may converge to a consensus and reduce to the self-training MentorNet. To cope with this issue, Co-teaching+ (Yu et al., 2019) was developed. Inspired by Decoupling (Malach and Shalev-Shwartz, 2017), Co-teaching+ adopts a similar disagreement strategy, consisting of the disagreement-update step and the cross-update step. In the disagreement-update step, the same training set \mathcal{D} is passed to both networks to produce a filtered training set \mathcal{D}' containing instances where the two networks’ predictions disagree. The cross-update step is similar to the Co-teaching where each network conducts the same sampling for its peer network to train but only from the new filtered training set \mathcal{D}' . As a result, this disagreement strategy manages to keep the two networks diverged with superior performance compared to Co-teaching.

Heuristic algorithms often demonstrate very competitive empirical performance. However, one significant weakness of these algorithms is that the learned clean classifiers do not guarantee statistical consistency (Equation 2.9) since no noise modelling is conducted. In contrast, another group of methods gives priority to this consistency property and directly involves the transition matrix in the process of learning clean classifiers. Depending on the noise dependency assumption on the instance (discussed

in Section 2.2.2.2), these methods can be further categorised, which are introduced in more detail in the following two sections.

2.2.3.2 Algorithms using Instance-Independent Transition Matrices

The instance-independent transition matrix generally refers to Equation 2.12, where the noise generation process is instance-independent but class-dependent. The concept of *anchor points* was proposed to learn such a matrix.

Anchor points Liu and Tao (2015) are a set of special instances defined over the clean data space. The general definition of anchor points (Xia et al., 2019) can be formulated as Equation 2.14

$$P(Y = i|X = x^i) \rightarrow 1 \text{ or } P(Y = i|X = x^i) = 1, \quad (2.14)$$

meaning that anchor points x^i are instances that belong to label class i surely or almost surely. This also implies that $P(Y = k|X = x^i) \approx 0$ for all $k \neq i$. Assuming anchor points x^i exist for each class i and the noisy class posterior is accurately learned, we can then identify instance-independent transition matrix T_{ij} row by row, as follows:

$$\begin{aligned} P(\tilde{Y} = j|x^i) &= \sum_{k=1}^C P(\tilde{Y} = j|Y = k)P(Y = k|x^i) \\ &= T_{ij}P(Y = i|x^i) \\ &= T_{ij} \\ \Rightarrow \hat{T}_{ij} &= \hat{P}(\tilde{Y} = j|x^i). \end{aligned} \quad (2.15)$$

For a given dataset, anchor points are commonly assumed to exist but are unknown (Xia et al., 2019), and need to be identified. Liu and Tao (2015) propose to take the most extreme examples of noisy class posterior for each class and identify them as anchor points, such as i.e., $x^i = \arg \max_x P(\tilde{Y} = i|x)$. However, this only holds in the context of binary classification.

Patrini et al. (2017) also adopt anchor points as the transition matrix estimator but in a multi-class classification context, demonstrating good empirical performance. Beyond that, the authors propose two modified loss functions using the estimated transition matrix, leading to desirable consistency described in Equation 2.9.

The first modified loss function is called *backward corrected loss*, and can be formulated as

$$\begin{aligned}\ell^{\leftarrow}(\hat{p}(\mathbf{y}|\mathbf{x})) &= T^{-1}\ell(\hat{p}(\mathbf{y}|\mathbf{x})) \\ \mathbb{E}_{\tilde{\mathbf{y}}|\mathbf{x}}[\ell^{\leftarrow}(\hat{p}(\mathbf{y}|\mathbf{x}))] &= \mathbb{E}_{\tilde{\mathbf{y}}|\mathbf{x}}[\ell(\hat{p}(\mathbf{y}|\mathbf{x}))] \quad \forall \mathbf{x}.\end{aligned}\tag{2.16}$$

The second equation implies that backward corrected loss ℓ^{\leftarrow} is a risk-consistent estimator, meaning that the original loss ℓ can produce the same expected risk as to the original loss over the clean distribution.

The second modified loss is *forward corrected loss* and can be defined as

$$\begin{aligned}\ell_{\psi}^{\rightarrow}(\mathbf{h}(\mathbf{x})) &= \ell(T^T \psi^{-1}(\mathbf{h}(\mathbf{x}))) \\ \arg \min_{\mathbf{h}} \mathbb{E}_{\mathbf{x}, \tilde{\mathbf{y}}}[\ell_{\psi}^{\rightarrow}(\mathbf{h}(\mathbf{x}))] &= \arg \min_{\mathbf{h}} \mathbb{E}_{\mathbf{x}, \mathbf{y}}[\ell_{\psi}(\mathbf{h}(\mathbf{x}))] \quad \forall \mathbf{x},\end{aligned}\tag{2.17}$$

where ψ^{-1} denotes the Softmax function. Different from the backward correlated loss, Forward corrected loss is not risk-consistent but instead classifier-consistent, meaning that it can produce the same minimiser as the original loss over the clean distribution.

Forward corrected loss is generally considered to be more preferable to the backward counterpart (Patrini et al., 2017). This is because forward corrected loss does not involve the computationally expensive matrix inverse, and it also has a better empirical performance. Given these benefits, many later works choose to incorporate the forward correction procedure into their learning algorithms. Specifically, a transition matrix is first estimated using their proposed transition estimation methods and then utilised to correct the loss function as a forward corrected loss $\ell_{\psi}^{\rightarrow}$ in Equation 2.17. By minimising the forward corrected loss, a statistically-consistent clean classifier $\psi^{-1}(h(x))$ can be eventually learned over noisy data, which is asymptotically the same as the optimal target classifier learned over clean data.

So far, all methods mentioned above rely on the existence of anchor points. However, anchor points may not always exist in a given dataset. In such a setting, the transition matrix may be poorly learned, leading to degenerated performance of learned classifiers.

To address this, another transition matrix estimation method, the T-revision method (Xia et al., 2019), was proposed. This transition estimation method does not employ anchor points and uses a slack variable to refine the transition matrix estimate. The learning follows a similar two-stage procedure as the previously mentioned anchor point-based methods. In the first stage, a noisy class posterior $\hat{P}(\tilde{\mathbf{Y}}|X = x)$ is learned, and \hat{T} is initialised using instances that are similar to anchor points, i.e., those with highest estimated noisy class posterior for each class. In the second stage, the initialised \hat{T} is modified by a

slack variable ΔT , which is learned and validated with a clean classifier f . In the original paper, ΔT and f can be learned by minimising the following weighted loss

$$\begin{aligned}\hat{R}(\hat{T} + \Delta T, f) &= \frac{1}{n} \sum_{i=1}^n \frac{g_{y_i}(x_i)}{((\hat{T} + \Delta T)^T g)_{y_i}(x_i)} \ell(f(x_i), \tilde{y}_i) \\ &= \frac{1}{n} \sum_{i=1}^n \tilde{\ell}(f(x_i), \tilde{y}_i),\end{aligned}\tag{2.18}$$

where $g(x_i) = \hat{P}(\mathbf{Y}|X = x_i)$, $g_{y_i}(x_i) = \hat{P}(Y = \tilde{y}_i|X = x_i)$, $f(x_i) = \arg \max g(x_i)$. $\tilde{\ell}$ is another modified loss function called *reweighted loss* and such reweighting philosophy is first proposed by Liu and Tao (2015). The clean classifier f learned in such a way is guaranteed to be risk-consistent.

By contrast, Dual-T (Yao et al., 2020) attempts to improve the transition matrix estimation by factorising the original transition matrix into two easier-to-estimate matrices, such as

$$\begin{aligned}T_{ij} &= P(\tilde{Y} = j|Y = i) \\ &= \sum_{\ell \in [C]} P(\tilde{Y} = j|Y' = \ell, Y = i) P(Y' = \ell, Y = i) \\ &\triangleq \sum_{\ell \in [C]} T_{\ell j}^{\clubsuit}(Y = i) T_{i\ell}^{\spadesuit},\end{aligned}\tag{2.19}$$

where T^{\clubsuit} models the transition from the clean class to the intermediate class, while T^{\spadesuit} models the transition from the intermediate class to the noisy class. Note that the intermediate class ℓ is constructed in a way such that their posterior distributions matches the noisy class posterior. Since T^{\clubsuit} shares the same form as the standard transition matrix, classical methods mentioned above can therefore be used to estimate it. T^{\spadesuit} is proposed to be estimated differently, such as

$$\begin{aligned}\hat{Y}_{\ell j}^{\spadesuit}(Y = i) &= P(\tilde{Y} = j|Y' = \ell) \\ &= \frac{\sum_{i=1} \mathbb{1}_{\{(\arg \max_k P(Y'=k|x_i)=\ell) \wedge (\tilde{y}_i=j)\}}}{\sum_{i=1} \mathbb{1}_{\{\arg \max_k P(Y'=k|x_i)=\ell\}}}.\end{aligned}\tag{2.20}$$

2.2.3.3 Algorithms using Instance-Dependent Transition Matrices

As mentioned in Section 2.2.2.2, instance-independent transition matrix is only an approximation of the real-world label noise generation process. Instance-dependent transition matrix models realistic noise, which is considered unidentifiable without *additional* assumptions.

Motivated by how humans perceive instances based on parts, Xia et al. (2020) proposes a novel but reasonable assumption that the instance-dependent transition matrix is part-dependent. Moreover, as illustrated in Figure 2.3, the authors further assume that the weight to combine part-dependent transition matrices as an instance-dependent transition matrix is identical to those to reconstruct an instance from parts. These assumptions make the instance-dependent transition matrix identifiable.

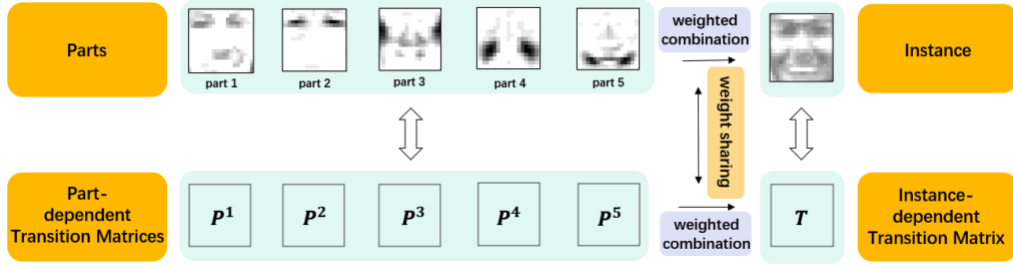


FIGURE 2.3. Approximate the instance-dependent transition matrix by a weighted combination of part-dependent transition matrices (Xia et al., 2020)

Specifically, a deep neural network acts a feature extractor, is first trained over noisy data to help obtain deep feature representations for all instances. Then by applying the philosophy of dictionary learning, the whole training data matrix X in the form of these extracted feature representations can be reconstructed as

$$\min_{W \in \mathbb{R}^{d \times r}, \mathbf{h}(\mathbf{x}_i) \in \mathbb{R}_+^r, \|\mathbf{h}(\mathbf{x}_i)\|_1 = 1, i=1, \dots, n} \sum_{i=1}^n \|\mathbf{x}_i - W\mathbf{h}(\mathbf{x}_i)\|_2^2, \quad (2.21)$$

where W is the “dictionary” matrix to learn and each column should denote as a part representation of all instances. $\mathbf{h}(\mathbf{x}_i)$ is a one-hot vector acting as a combination coefficient for instance \mathbf{x}_i , which also needs to be learned. Once W and all $\mathbf{h}(\mathbf{x}_i)$ are learned, by the previous assumptions, the instance-dependent transition matrix for each instance \mathbf{x} can then be modelled as

$$T(\mathbf{x}) \approx \sum_{j=1}^r \mathbf{h}_j(\mathbf{x}) P^j. \quad (2.22)$$

$\mathbf{h}(\mathbf{x})$ is learned as above and j denotes the j -th entry which is either 0 or 1. P^j denotes the part-dependent transition matrix of the corresponding j -th part representation. Assuming $k, k \geq r$ anchor points of each class are given, then we can learn the part-dependent transition matrices as

$$\begin{aligned} \min_{P^1, \dots, P^r \in [0, 1]^{c \times c}} & \sum_{i=1}^c \sum_{\ell=1}^k \|T_{i \cdot}(\mathbf{x}_\ell^i) - \sum_{j=1}^r \mathbf{h}_j(\mathbf{x}_\ell^i) P_{i \cdot}^j\|_2^2 \\ \text{s.t. } & \|P_{i \cdot}^j\|_1 = 1 \quad \forall i \in [c], j \in [r]. \end{aligned} \quad (2.23)$$

If anchor points are not given, they can then be identified from the noisy data using our previously mentioned methods (Liu and Tao, 2015; Patrini et al., 2017; Xia et al., 2019). Last but not least, it is worth mentioning that the authors empirically verified that the performance of such an estimation method is insensitive to hyperparameter r , which is the number of part-representation of the original data matrix X is decomposed to.

By contrast, Yang et al. (2022) takes a very different perspective to model the instance-dependent transition matrix and is considered the first work that estimates an instance-dependent transition matrix using a deep neural network. Traditionally and most commonly, the instance-dependent transition matrix models the transition from clean to noisy labels. However, Bayes-Label-T proposes to approximate such a matrix using the Bayes label transition matrix, which models the transition from Bayes labels to noisy labels instead. The Bayes label y^* of a given instance x maximises the clean class posterior, such as

$$y^* = \arg \max_y P(\mathbf{Y}|X = x). \quad (2.24)$$

The corresponding Bayes class posterior is therefore a one-hot vector rather than a probability distribution. As illustrated in Figure 2.4, the Bayes class posterior is much more sparse compared to both clean and noisy class posteriors, leading to a smaller solution space of the Bayes label transition matrix compared to the traditional counterpart. Hence it is believed that the estimation of Bayes label transition matrix can be estimated more efficiently given the same amount of training data.

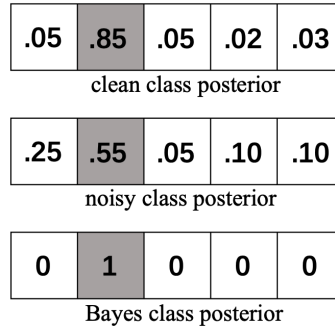


FIGURE 2.4. Comparison of clean, noisy and Bayes class posteriors (Yang et al., 2022)

The authors propose to estimate the Bayes transition matrix in two steps. The first step is to collect Bayes labels from noisy training data. In the literature, the Bayes label of a given instance x can be collected through a concept called *distilled example* (x, y^*, \tilde{y}) . y^* of a distilled example is identical to

the label assigned by the optimal clean classifier (Equation 2.24), and \tilde{y} is the associated noisy label of instance x given by the noisy training data. Note that \tilde{y} and y^* may disagree.

To identify distilled examples, Cheng et al. (2020) proposes a way with theoretical guarantee but for binary setting only. Bayes-Label-T extends its approach to a multi-classification context, such as

$$\hat{P}_{\tilde{\mathcal{D}}}(\tilde{Y} = \hat{y}^* | X = x) > \frac{1 + \rho_{max}}{2} \Rightarrow (x, \hat{y}^*) \text{ is distilled}, \quad (2.25)$$

where the LHS is the output of a noisy classifier trained over noisy data. ρ_{max} is the noise rate upper-bound, which is shown to be a relatively insensitive hyperparameter.

Once distilled examples are collected, the authors adopt a neural network (the grey box in Figure 2.5) to model the Bayes label transition matrix. First, the feature extractor extracts deep feature representations for all input instances. Then a linear layer FC models the label noise generation process, which is dependent on the features. The output of the linear layer, a $C^2 \times 1$ probability distribution vector, is finally reshaped to produce a squared transition matrix.

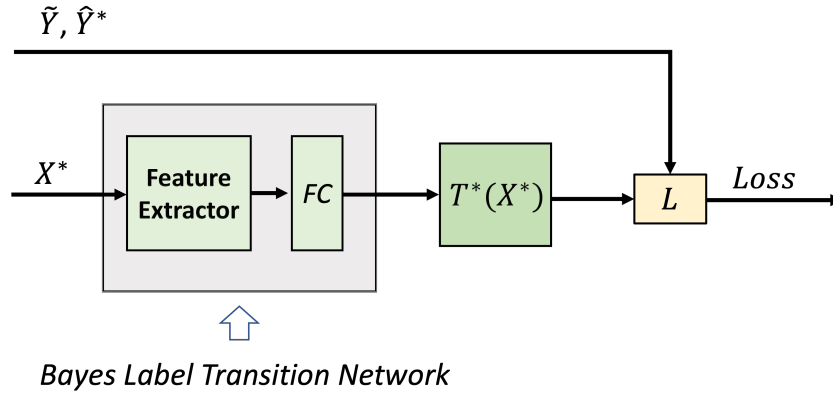


FIGURE 2.5. Estimate Bayes label transition matrix using collected distilled examples

Mathematically, to obtain such a transition matrix, the objective function can be formulated as

$$\min_{\theta} \hat{R}(\theta) = \frac{1}{N^*} \sum_{i=1}^{N^*} \tilde{y}_i \log(\hat{y}_i^* \cdot T^*(x_i^*; \theta)). \quad (2.26)$$

where \hat{y}_i^* and \tilde{y}_i denote the collected Bayes label and the associated noisy label for instance x_i , which are both one-hot vectors. θ denotes the neural network parameters and N^* denotes the number of collected distilled examples.

The estimated Bayes label transition matrix is then fixed and incorporated into a forward corrected loss (detailed in Section 2.2.3.2) to learn for a consistent Bayes label classifier modelled by a neural network. As illustrated in Figure 2.6, the Bayes label classification network shares the same architecture as the Bayes transition network but the last linear layer FC models classification instead.

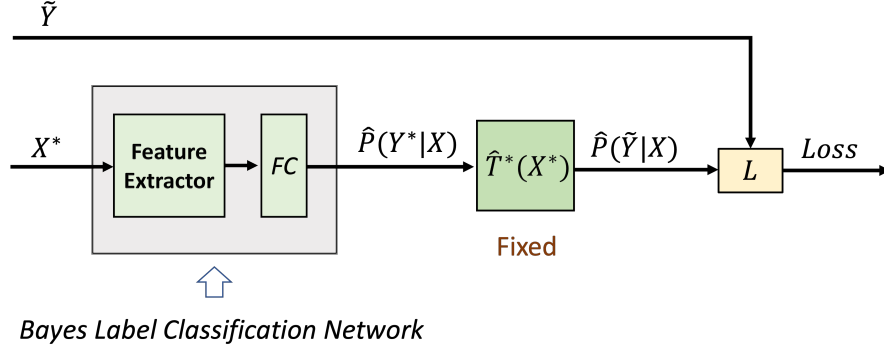


FIGURE 2.6. Training a Bayes label classification network by forward corrected

The objective function can be formulated as

$$\min_w \hat{R}(w) = \frac{1}{N^*} \sum_{i=1}^{N^*} \tilde{y}_i \log(f(x_i^*; w) \cdot T^*(x_i^*; w)), \quad (2.27)$$

where $f(x_i^*; w)$ denotes the clean class posterior, produced as the output of network softmax layer.

The Bayes label transition matrix is shown to generalise well over unseen data with a superior classification performance, demonstrating the effectiveness of its methodology.

2.3 Multi-task Learning

2.3.1 Background

When learning multiple tasks simultaneously, humans are skilled at utilising knowledge from one task and applying it to help the learning of another related task. For instance, when learning squash and tennis together, the knowledge of playing tennis may help play squash. Multi-task learning (MTL) (Caruana, 1997) is a machine learning paradigm inspired by this characteristic of human learning, which learns multiple *related* tasks simultaneously. The goal is to improve the learning performance of *every* single task by leveraging the knowledge contained in other related tasks (Zhang and Yang, 2021).

Given the MTL definition above, questions arise regarding “task relatedness”. How can we characterise when tasks are related, and how can we exploit it for learning? A recent survey (Zhang and Yang, 2021) simplifies these questions into three key points (1) When to share (2) What to share (3) How to share, and address them one by one.

The “when to share” problem is about deciding when there is relatedness between tasks, which indicates when prioritise the MTL framework compared to the traditional single-task one. Multiple contributions in the literature assume task-relatedness based on prior knowledge but without any validation process. By comparison, a more advanced solution, suggested by a recent survey (Zhang and Yang, 2021), is to adopt models that can directly *learn* the task relatedness from data. The MTRL model is a representative model which quantifies the task relatedness as a task covariance matrix Σ that is learnable from the data. If Σ becomes diagonal, meaning tasks are learned to be unrelated, then the multi-task learning model will automatically degenerate to a single-task counterpart. After the learning, MTRL re-evaluates the learned Σ and further checks if the learned relationship matches its prior knowledge of the tasks.

The “what to share” problem is addressed through what medium or form can we share the knowledge among related tasks. In the literature, most works share knowledge through features or parameters. Feature-based methods aim to learn a commonly shared feature representation across different tasks. By contrast, parameter-based methods prefer to model the task parameters and then enable knowledge sharing through regularisation.

Finally, given the form to share, we need to address the “how to share” problem and decide the specific ways of sharing the knowledge. According to a recent survey (Zhang and Yang, 2021), and a related book (Yang et al., 2020), for feature-based learning, there are three main approaches, i.e., feature transformation, feature selection and deep learning. Meanwhile, parameter-based learning has four main approaches. That is task-relation, decomposition, low-rank, and task clustering.

In the following sections, we will first formally define the MTL problem and then survey and compare some representative works for each approach mentioned above.

2.3.2 Problem Definition

Consider m supervised learning tasks $\mathcal{T} = \{\mathcal{T}_1, \dots, \mathcal{T}_m\}$ where each task \mathcal{T}_i is associated with a task-specific labelled training set $\mathcal{D}_i = \{x_j^i, y_j^i\}_{j=1}^{n_i}$. Assuming that at least a subset of tasks is related but

not identical, the goal of multi-task learning is to learn all m tasks together and improve the learning performance of each task by using the knowledge from all or some of the other tasks (Yang et al., 2020).

2.3.3 Existing Work

2.3.3.1 Feature-Based Learning

Given that tasks are related, feature-based learning methods interpret the task relatedness from the feature perspective and assume that different tasks share a common feature representation. By leveraging training data from all tasks, the learned representation is expected to be more powerful than the one under single training data, leading to improved performance of classifiers for all tasks.

In the literature, there are three main approaches to construct the shared feature representation, i.e., feature transformation, feature selection and deep learning (Yang et al., 2020; Zhang and Yang, 2021).

Feature Transformation Approach The multi-layer feed-forward neural network model (Caruana, 1997) is one of the earliest MTL models, and it follows the feature transformation approach. As illustrated in Figure 2.7, the network's hidden layers take all task-specific features as input and transform them to output a common feature representation.

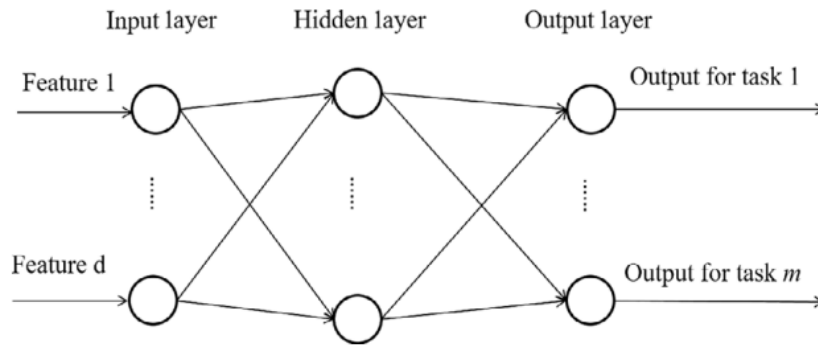


FIGURE 2.7. The multi-layer feedforward neural network (Caruana, 1997; Yang et al., 2020)

However, the multi-layer feed-forward neural network model may have redundancy in its hidden representations. To address this, the multi-task feature learning (MTFL) method (Argyriou et al., 2007)

formulates the problem under the regularisation framework with an objective function:

$$\begin{aligned} \min_{A, U, b} & \sum_{i=1}^m \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(y_j^i, (a^i)^T U^T x_j^i + b_i) + \lambda \|A\|_{2,1}^2 \\ \text{s.t. } & UU^T = I. \end{aligned} \quad (2.28)$$

U is a square transformation matrix used to linearly transform each data input x_j^i and A is the model parameters of all tasks after the transformation. The $UU^T = I$ constraint encourages the U transformation matrix to be orthogonal with less redundant information. The $\ell_{2,1}$ norm encourages A to be row-sparse, which is equivalent to feature selection.

The multi-task sparse coding method (Maurer et al., 2013) is another similar method under the regularisation framework, with the objective function as

$$\begin{aligned} \min_{A, U, b} & L(UA, b) \\ \text{s.t. } & \|a^i\|_1 \leq \lambda \forall i \in [m], \|u^j\|_2 \leq 1 \forall j \in [D]. \end{aligned} \quad (2.29)$$

The transformation matrix $U \in R^{d \times D}$, also referred to as a dictionary in the dictionary learning field, is an overcomplete matrix instead of a square matrix.

Feature Selection Approach As the name indicates, the feature selection approach constructs the shared representation by selecting features rather than transformation. The most common way to select features is to regularise the weight matrix W through $\ell_{p,q}$ norm, such as

$$\min_{W, b} \sum_{i=1}^m \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(y_j^i, (w^i)^T x_j^i + b_i) + \lambda \|W\|_{p,q}, \quad (2.30)$$

where $\|W\|_{p,q} = \|(\|w_1\|_p, \dots, \|w_d\|_p)\|_q$ and $\|w_1\|_p$ denotes the ℓ_p norm of W 's i -th row vector. In effect, the $\ell_{p,q}$ regularisation makes the weight matrix W row-sparse, ensuring only the most contributing features are selected. There are many extensions of $\ell_{p,q}$ regularisation with specific improvements. For example, the capped- $\ell_{p,1}$ regulariser for more sparse features (Gong et al., 2014).

Deep Learning Approach The deep learning approach can be considered as a special case of the feature transformation approach but with many more hidden layers. Similarly, the first several hidden layers are shared to produce a shared representation for different tasks (Liu et al., 2015; Zhang et al., 2016b; Shinohara, 2016).

However, for these models, there is no principal approach regarding how many layers to share. The best-performing architecture is commonly believed to be task-dependent and requires trial-and-error. Motivated by this inconvenience, the cross-stitch network (Misra et al., 2016) was proposed to automatically learn the best sharing scheme, i.e. the best combination of shared and task-specific layers.

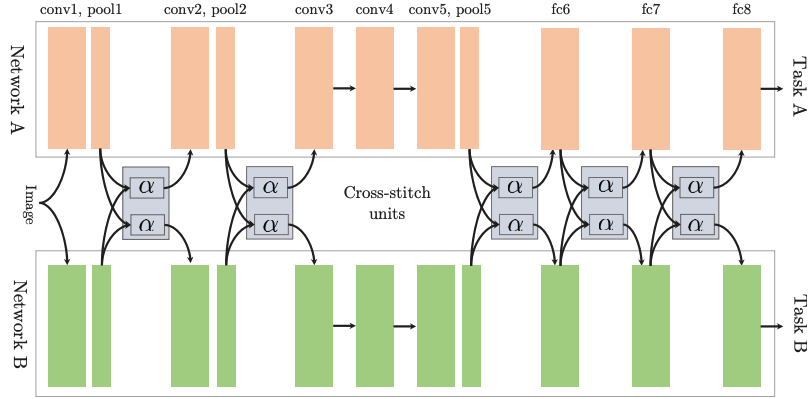


FIGURE 2.8. The architecture of the cross-stitch network (Misra et al., 2016)

The essential contribution is the design of cross-stitch units, which learns to share or isolated network layers and can be updated easily through backpropagation. Consider an example in Equation 2.31 where X_A^{ij} and X_B^{ij} denote the j -th hidden units of i -th hidden layer in both networks A and B. The cross-stitch operation creates new hidden units \tilde{X}_A^{ij} and \tilde{X}_B^{ij} by linearly combining the original ones, where the networks can learn each parameter in the parameter matrix. Higher values of α_{AB} and α_{BA} denote more shared representation, while zeros denote no sharing. Thanks to it, the cross-stitch network is much more generalisable for various tasks compared to other deep learning MTL models.

$$\begin{bmatrix} \tilde{X}_A^{ij} \\ \tilde{X}_B^{ij} \end{bmatrix} = \begin{bmatrix} \alpha_{AA} & \alpha_{AB} \\ \alpha_{BA} & \alpha_{BB} \end{bmatrix} \begin{bmatrix} X_A^{ij} \\ X_B^{ij} \end{bmatrix} \quad (2.31)$$

2.3.3.2 Parameter-Based Learning

Parameter-based learning characterises task-relatedness via model parameters relatedness. Then, depending on the specific techniques to relate the parameter, it categorises the existing approaches into four groups. That are the low-rank approach, task clustering approach, task relation learning approach and decomposition approach.

Low-Rank Approach The low-rank approach relates the model parameter through the rank of parameter matrix W , where more task relatedness implies a lower rank of W .

Ando et al. (2005) formulates the weight matrix as $W = U + \Theta^T V$ where Θ is the low-rank subspace shared by different tasks and U is the matrix containing all task-specific parameters. Based on this formulation, the objective function can be written as

$$\begin{aligned} \min_{U, V, \Theta, b} \sum_{i=1}^m \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(y_j^i, (u^i + \Theta^T v^i)^T x_j^i + b_i) + \lambda \|U\|_F^2 \\ \text{s.t. } \Theta \Theta^T = I. \end{aligned} \quad (2.32)$$

Intuitively, the orthogonal constraint on the shared parameter matrix Θ encourages less redundancy and, therefore, more sharing, while the regularisation on task-specific parameter matrix U penalises for task-specific features.

Chen et al. (2009) generalises the above objective formulation with some convex relaxation technique such that the optimisation is easier. The new formulation of the objective function becomes:

$$\begin{aligned} \min_{W, b, M} L(W, b) + \lambda \text{tr}(W^T (M + \eta I)^{-1} W) \\ \text{s.t. } \text{tr}(M) = h, 0 \preceq M \preceq I, \end{aligned} \quad (2.33)$$

where M models the feature covariance for all the tasks.

In the field of optimisation, adopting trace norm of a matrix as a regulariser is known to have a low-rank effect. Pong et al. (2010) utilises this characteristic and reformulates the objective function above as

$$\min_{W, b} \sum_{i=1}^m \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(y_j^i, (w^i)^T x_j^i + b_i) + \lambda \|W\|_{S(1)}, \quad (2.34)$$

where $\|W\|_{S(1)}$ is the trace norm of matrix W . This formulation is further extended by Han and Zhang (2016) where each singular value of W is capped by some pre-defined threshold. Such modification only penalises small singular values, encouraging an even lower rank of W than the trace norm.

Task Clustering Approach The task clustering approach adopts the general philosophy of clustering and groups tasks into different clusters where each cluster shares similar model parameters.

The first MTL clustering algorithm is proposed by Thrun and O’Sullivan (1996). It first identifies several clusters based on the model parameters learned in the single-task learning setting, and then for every

cluster, a multi-task model is learned for tasks within the cluster. However, such an algorithm may not be optimal due to the decoupling of both clustering and learning.

In contrast, Bakker and Heskes (2003) proposed a neural network model under the Bayesian framework. The architecture is similar to the previously mentioned multi-layer feed-forward neural network, where the input-to-hidden layers are shared while the hidden-to-output layers are task-specific. The difference is that in the Bayesian neural network, the task-specific weight w^i of task \mathcal{T}_i follows a Gaussian mixture model $w^i \sim \sum_{j=1}^r \pi_j \mathcal{N}(\cdot | m_j, \Sigma_j)$ and tasks in the same cluster share a same Gaussian distribution. Xue et al. (2007) proposed another Bayesian model, which is widely used except that the weight follows a Dirichlet process.

Different from the Bayesian methods, Jacob et al. (2008) learns the task clustering under the regularisation framework and penalises the parameter size, the within-cluster variance and the inter-cluster variance of the model.

Kang et al. (2011) combines the methods of MTF (Argyriou et al., 2007) from the feature transformation approach as well as the trace norm from the low-rank approach and formulates the objective function as

$$\begin{aligned} \min_{W, b, \{Q_i\}} & \sum_{i=1}^m \frac{1}{n_i} \sum_{j=1}^{n_i} \ell(y_j^i, (w^i)^T x_j^i + b_i) + \lambda \sum_{i=1}^r \|W Q_i\|_{S(1)}^2 \\ \text{s.t. } & Q_i \in \{0, 1\}^{m \times m} \forall i \in [r], \sum_{i=1}^r Q_i = I. \end{aligned} \quad (2.35)$$

Q_i is a diagonal cluster indicator, and a 1 in the j -th diagonal entry indicates \mathcal{T}_j belongs to the i -th cluster. The sum of Q_i equals one represents that each task can only belong to one cluster, which is an assumption adopted by most methods in the task clustering approach. Kumar and Daume III (2012) relaxes this assumption and allows each task to belong to multiple clusters.

Task Relation Learning Approach The task relation learning approach often quantifies parameter relatedness through task relation measures.

In the early stages of MTL research, task relations were often assumed or given as prior information and model parameters were learned by leveraging such knowledge. For example, Evgeniou and Pontil (2004) assumes that all tasks are similar and regularises the weight parameter w_i of task i to be similar to the average weight parameter of all tasks. Kato et al. (2007) assumes that a task network is similar to

all its neighbour networks and regularises their weight to be similar. However, these methods are limited as assumptions may not hold in practice, and prior knowledge is often unavailable.

Motivated by this, many approaches have been developed. Zhang and Yeung (2012, 2014) propose a multi-task relationship learning (MTRL) model where not only the task relation can be learned directly from data but also all possible task relationships can be explicitly modelled using a task covariance matrix. Specifically, the positive task correlation ensures that similar tasks have similar model parameters. In contrast, the negative task correlation can help the model to reduce optimisation space since model parameters are likely to be dissimilar. The task unrelatedness identifies outlier tasks and prevents the performance of other tasks from being impaired by unrelated tasks.

The specific modelling is formulated as follows. A matrix-variate normal prior is placed on the weight matrix as $W \sim \mathcal{MN}(0, I, \Sigma)$ where 0 , I and Σ denote the mean, row feature covariance and column task covariance of W respectively. Since the Maximum A Posterior (MAP) estimation of W is proportional to the product of the MLE estimation and such prior, by taking a negative logarithm of the estimation formulation, we can then obtain the following objective function

$$\begin{aligned} \min_{W, b, \Sigma} L(W, b) + \lambda_1 \|W\|_F^2 + \lambda_2 \text{tr}(W \Sigma^{-1} W^T) \\ \text{s.t. } \Sigma \succ 0, \text{tr}(\Sigma) \leq 1. \end{aligned} \quad (2.36)$$

The first and second regularisation terms control the complexity of both weight matrix W and the task covariance matrix Σ . Since the objective function is jointly convex with respect to W , b and Σ , the classic alternating updating rule is conducted for optimisation.

Long et al. (2017) proposed a multi-linear relationship network (MRN) approach, which extends the MTRL method further by placing a tensor-variate normal prior on the parameter tensor stacked by parameters of task-specific layers. The core design of MRN is the tensor normal prior, enabling the modelling of the multi-linear relationship across task-specific layer parameter tensors. If tasks are related, the knowledge can be transferred accordingly in the corresponding layers; if tasks are unrelated, the layers will remain independent to mitigate negative knowledge transfer.

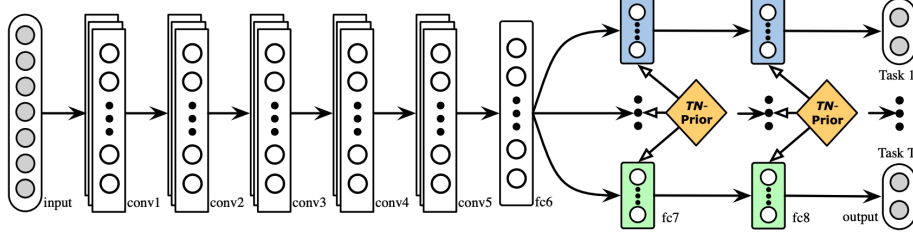


FIGURE 2.9. The architecture of Multilinear relationship network (MRN) (Long et al., 2017)

The model network of MRN is illustrated as Figure 2.9, where AlexNet is the backbone network. The first few uncolored layers are shared layers, producing a shared feature representation that is shared among all tasks. This construction is similar to the feature-based learning mentioned previously. The remaining colored layers are task-specific layers, and they remain flexible depending on the task relationships. The tensor normal prior design encourages more "sharing" across the layer parameter tensor if they are characterised as related or keeps layers independent if they are unrelated.

Similar to MTRL, the objective function of MRN is derived from the MAP estimation of task-specific parameters. Consider the model in Figure 2.9 as an example. Denote task specific layers as $\mathcal{L} = \{fc7, fc8\}$, the model parameter as \mathcal{W} and the parameter tensor of task-specific layer ℓ as $\mathcal{W}^\ell = [\mathcal{W}^{1,\ell}; \dots; \mathcal{W}^{T,\ell}] \in \mathbb{R}^{D_1 \times D_2 \times T}$ where $\ell \in \mathcal{L}$. Then the MAP estimation of \mathcal{W} given training data of multiple tasks can be formulated as

$$\begin{aligned}
 p(\mathcal{W}|\mathcal{X}, \mathcal{Y}) &\propto p(\mathcal{W}) \cdot p(\mathcal{Y}|\mathcal{X}, \mathcal{W}) \\
 &= \underbrace{\prod_{\ell \in \mathcal{L}} p(\mathcal{W}^\ell)}_{\text{Prior}} \cdot \underbrace{\prod_{t=1}^T \prod_{n=1}^{N_t} p(y_n^t | x_n^t, \mathcal{W}^\ell)}_{\text{MLE}},
 \end{aligned} \tag{2.37}$$

where \mathcal{W}^ℓ is assumed to be independent across layers, and all tasks are also assumed to be independent.

For the MLE part, it can be easily modelled by the CNN network. For the prior part, the layer-specific parameter tensor \mathcal{W}^ℓ is assumed to be sampled from the tensor normal distribution

$$vec(\mathcal{W}^\ell) \sim \mathcal{N}(vec(\mathcal{M}), \Sigma_1^\ell \otimes \Sigma_2^\ell \otimes \Sigma_3^\ell), \forall \ell \in \mathcal{L}, \tag{2.38}$$

where $\Sigma_1^\ell \in \mathbb{R}^{D_1^\ell \times D_1^\ell}$, $\Sigma_2^\ell \in \mathbb{R}^{D_2^\ell \times D_2^\ell}$, and $\Sigma_3^\ell \in \mathbb{R}^{T^\ell \times T^\ell}$ denote the covariance modelling feature covariance, class covariance and task covariance respectively. $vec(\mathcal{W}^\ell)$ denotes the vectorisation of

tensor \mathcal{W}^ℓ , with a density function written as

$$p(\text{vec}(\mathcal{W}^\ell)) = (2\pi)^{-\frac{\prod_{k=1}^K d_k}{2}} \left(\prod_{k=1}^K |\Sigma_k|^{-\frac{\prod_{k=1}^K d_k}{2d_k}} \right) \times \exp\left(-\frac{1}{2}(\text{vec}(\mathcal{W}^\ell) - \text{vec}(\mathcal{M}))^T (\Sigma_1 \otimes \dots \otimes \Sigma_K)^{-1} (\text{vec}(\mathcal{W}^\ell) - \text{vec}(\mathcal{M}))\right), \quad (2.39)$$

where \mathcal{M} denotes the mean tensor of weight parameter tensor \mathcal{W}^ℓ and the covariance matrices are assumed to be shared across layers, e.g., $\Sigma_1^\ell = \Sigma_1$.

Therefore, by taking the negative logarithm of Equation 2.37, we can obtain the objective function formulation

$$\min_{f_t |_{t=1, \dots, T}, \Sigma_k^\ell |_{k=1, \dots, K}} \sum_{t=1}^T \sum_{n=1}^{N_t} \ell(f_t(x_n^t), y_n^t) + \frac{1}{2} \sum_{\ell \in \mathcal{L}} \left(\text{vec}(\mathcal{W}^\ell)^T (\Sigma_1 \otimes \dots \otimes \Sigma_K)^{-1} \text{vec}(\mathcal{W}^\ell) + \sum_{k=1}^K \frac{D}{D_k^\ell} \ln(|\Sigma_k^\ell|) \right) \quad (2.40)$$

where f_t denotes the classifier learned by the CNN network for each task t and ℓ denotes the cross-entropy loss function.

Decomposition Approach For decomposition approach, the philosophy is to decompose the task parameter matrix W into two or more matrices such as $W = \sum_{k=1}^h W_k$ and then regularise component matrices accordingly. The objective function is generally formulated as

$$\min_{W_i \in C_W, b} L\left(\sum_{k=1}^h W_k, b\right) + \sum_{k=1}^h g_k(W_k), \quad (2.41)$$

where C_W is a set of constraints for component matrices and g_k is a regulariser of component matrix W_k .

Based on this general formulation, most existing work utilises different g_k and C_W to achieve different regularisation effects.

Multi-Task Learning for Label Noise Learning

In this chapter, we first introduce the proposed MTL-T approach for learning a clean classifier from a dataset with multiple labellings. We begin by describing the key idea of MTL-T approach. We then discuss the motivation behind our choice of methods. Finally we describe the mathematical formulation of our method.

3.1 Proposed Solution

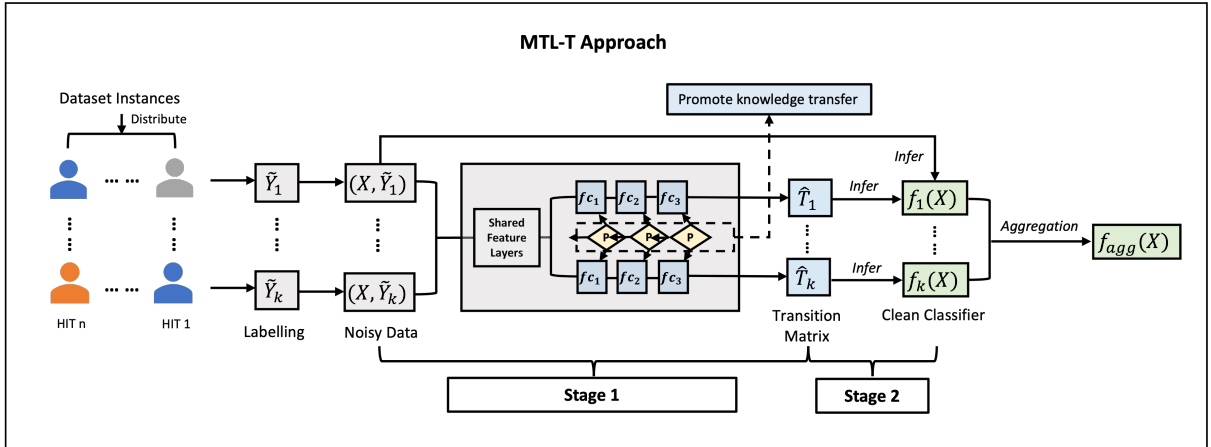


FIGURE 3.1. MTL-T, the proposed solution

The general learning procedure of MTL-T is illustrated in Figure 3.1. Given a dataset with multiple labellings, in Stage 1, we estimate all transition matrices, i.e., tasks, simultaneously in an adapted Bayes transition network. The feature layers are shared but the three linear layers are being task-specific. For each task-specific layer, layer-specific task relatedness is learned. If tasks are found to be related over that layer, knowledge transfer will be promoted; if tasks are found to be unrelated, layers will remain independent to mitigate potential harmful transfer. Once all transition matrices are estimated, in Stage

2, we apply the forward correction procedure to learn all clean classifiers using their corresponding transition matrices. In the end, we aggregate the learned clean classifiers using prediction aggregation. By exploiting the potential task relatedness in Stage 1, the estimation of each transition matrix is potentially improved, leading to more accurate clean classifiers downstream.

3.2 Method Selection

Our proposed MTL-T approach adapts Bayes-Label-T method (Yang et al., 2022) as the transition matrix estimation method and the MRN model (Long et al., 2017) as the multi-task learning technique. In the following section, we reason about our choice of methods.

Bayes-Label-T method (detailed in Section 2.2.3.3) is an instance-dependent transition matrix estimation method that models the matrix in a deep neural network called the Bayes label transition network. In such a network, the feature layers learn a feature representation of an instance, and the last linear layer models the label noise generation process, dependent on the feature, i.e., the instance itself. Compared to the instance-independent counterpart (Section 2.2.3.2), Bayes-Label-T provides more realistic modelling for the real-world label noise we are interested in (Han et al., 2020). Additionally, Bayes-Label-T is considered the first work to learn an instance-dependent transition matrix in a parameterised way through deep neural network layers (Yang et al., 2022).

Meanwhile, the MRN model (detailed in Section 2.3.3.2) is a parameter-based approach that characterises task-relatedness through parameter-relatedness. It models all the tasks in a single deep neural network, where each task has some shared and some task-specific layers. The key strength is that it learns the layer-specific task-relatedness from data rather than from prior assumptions, so knowledge transfer can be conducted based on the learned-relatedness.

Since both models use a deep neural network, it becomes convenient to incorporate the MRN model into the Bayes label transition network. Firstly, considering all transition matrices, i.e., tasks, are associated with the same dataset instances, we can create shared feature layers in the Bayes label transition network to represent them. Secondly, since all transition matrices also differ in the noise generation process, we can create several task-specific linear layers to model them separately. Finally, the tensor normal prior is imposed on the parameter tensor of the task-specific layers to exploit task relatedness through knowledge transfer. Overall, we can obtain an adapted Bayes label transition network with the capability

of exploiting task relatedness, potentially improving the estimation of each transition matrix, hence the clean learned classifiers.

3.3 Method Formulation

We detail the mathematical formulation of the MTL-T approach in two stages.

Stage 1: Estimate k transition matrices with knowledge transfer

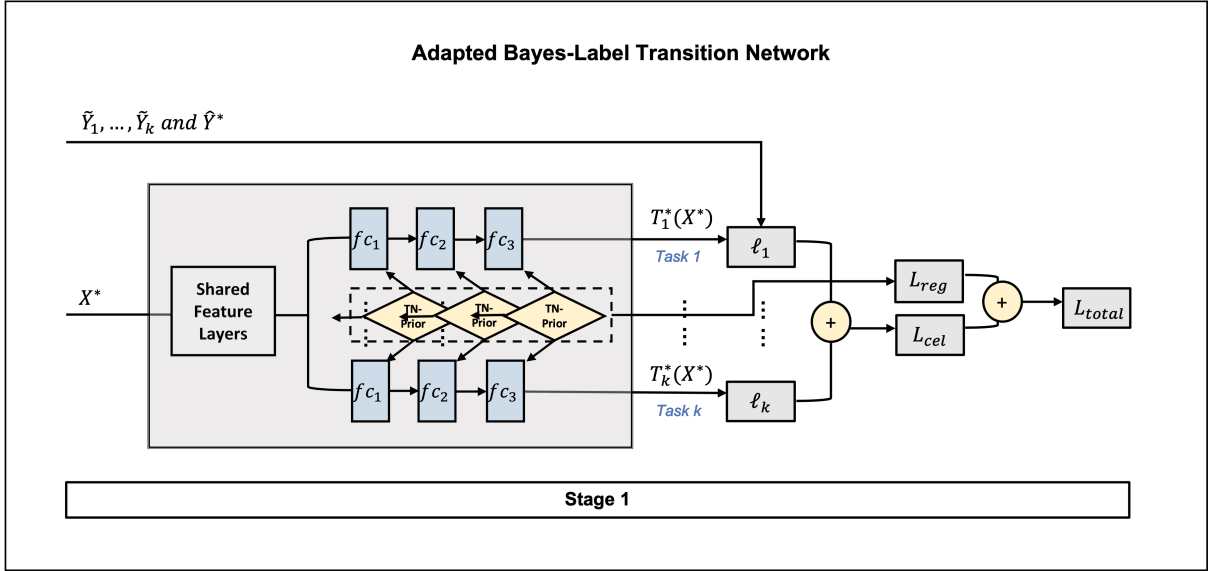


FIGURE 3.2. adapted Bayes-Label transition network

Objective Function

Denote all task-specific layers in Figure 3.2 as $\mathcal{L} = \{f_{c1}, f_{c2}, f_{c3}\}$ and all-task specific parameters from k tasks as $\mathcal{W} = [\mathcal{W}^\ell], \ell \in \mathcal{L}$. Each $\mathcal{W}^\ell = [\mathcal{W}^{1,\ell}; \dots; \mathcal{W}^{k,\ell}] \in \mathbb{R}^{D_1 \times D_2 \times k}$ represents a $D_1 \times D_2 \times k$ parameter tensor for layer ℓ , stacked by $D_1 \times D_2$ parameters from all k tasks. Additionally, we denote $\tilde{S} = \{(x_i, \tilde{y}_i^1, \dots, \tilde{y}_i^k)\}_{i=1}^N$ as the noisy dataset and $\tilde{S}^* = \{(x_i^*, \hat{y}_i^*, \tilde{y}_i^1, \dots, \tilde{y}_i^k)\}_{i=1}^{N^*}$ as a distilled dataset collected from \tilde{S} (using a similar collection process as the Bayes-Label-T method).

Similar to the MRN model, we also derive the objective function through the Maximum a Posterior (MAP) estimation of the task-specific parameters \mathcal{W} , such as

$$\begin{aligned} p(\mathcal{W}|\mathcal{X}^*, \mathcal{Y}^*, \tilde{\mathcal{Y}}) &\propto p(\mathcal{W}) \cdot p(\tilde{\mathcal{Y}}|\mathcal{X}^*, \mathcal{Y}^*, \mathcal{W}) \\ &= \underbrace{\prod_{\ell \in \mathcal{L}} p(\mathcal{W}^\ell)}_{\text{Prior}} \cdot \underbrace{\prod_{j=1}^k \prod_{i=1}^{N^*} p(\tilde{y}_i^j | \hat{y}_i^*, x_i^*, \mathcal{W})}_{\text{MLE}}. \end{aligned} \quad (3.1)$$

On the prior side, each layer-specific parameter tensor \mathcal{W}^ℓ follows a tensor normal distribution, such as

$$p(\mathcal{W}^\ell) = \mathcal{TN}_{D_1^\ell \times D_2^\ell \times T}(O, I, I, \Sigma^\ell), \forall \ell \in \mathcal{L}, \quad (3.2)$$

where both feature and class covariances are set to be identity matrices on purpose. Since all of our instances and classes are associated with the same dataset, both covariances seem to be less relevant here. Another difference is that we learn layer-specific task covariance matrix Σ^ℓ rather than assuming them to be the same. Since we have decomposed the modelling of each transition matrix, i.e., task as three linear layers, we can learn more granular task relatedness over each linear layer.

The MLE side is modelled by an adapted Bayes label transition network. The corresponding loss is represented as a sum of cross-entropy loss from all k tasks, with each task-specific loss being Equation 2.26 of the Bayes-Label-T method (Section 2.2.3.3).

Overall, by expanding the prior term using the density function and then taking a negative logarithm of Equation 3.1 we obtain the following constrained optimisation equation:

$$\min_{\mathcal{W}^\ell, \Sigma^\ell | \ell \in \mathcal{L}} \underbrace{\sum_{j=1}^k \sum_{i=1}^{N^*} \tilde{y}_i^j \log(\hat{y}_i^* T_j^*(x_i^*; \mathcal{W}))}_{L_{cel} \text{ from MLE}} + \underbrace{\sum_{\ell \in \mathcal{L}} \lambda_\ell \left[\text{vec}(\mathcal{W}^\ell)^T (\Sigma^\ell)^{-1} \text{vec}(\mathcal{W}^\ell) + D_1^\ell D_2^\ell \log(|\Sigma^\ell|) \right]}_{L_{reg} \text{ from Prior}}, \quad (3.3)$$

The equation denotes the total loss $L_{total} = L_{cel} + L_{reg}$. In the LHS, L_{cel} denotes a sum of total cross-entropy loss from all k tasks. In the RHS, L_{reg} denotes the regularisation loss. Specifically, the $\text{vec}(\mathcal{W}^\ell)$ term denotes the vectorisation of the parameter tensor, $\text{vec}(\mathcal{W}^\ell)^T (\Sigma^\ell)^{-1} \text{vec}(\mathcal{W}^\ell)$ term regularises \mathcal{W}^ℓ according to the task-relationship and the $D_1^\ell \times D_2^\ell \log(|\Sigma^\ell|)$ term penalises the covariance matrix complexity. The layer-specific regularisation parameter λ_ℓ controls the strength of regularisation and will be tuned during our experiments.

Optimisation Procedure

The objective function 3.3 is non-convex with respect to both \mathcal{W} and the covariance matrix Σ . Hence we adopt a similar alternative updating strategy to the MRN model. That is,

- (1) Update $\mathcal{W}^{j,\ell}$ while keep Σ^ℓ fixed

$$\frac{\partial L_{total}}{\partial \mathcal{W}^{j,\ell}} = \frac{\partial L_{cel}}{\partial \mathcal{W}^{j,\ell}} + \left[(\Sigma^\ell)^{-1} \text{vec}(\mathcal{W}^\ell) \right]_{..j} \quad \forall \text{ Task } j \in [k]$$

- (2) Update Σ^ℓ while keeping \mathcal{W}^ℓ fixed

$$\Sigma^\ell = \frac{1}{D_1^\ell D_2^\ell} (\mathcal{W}^\ell)_{(3)} (\mathcal{W}^\ell)_{(3)}^T + \epsilon I_k,$$

where ϵ is a small penalty added for the sake of numerical stability.

Stage 2: Learn k clean classifiers

Once k Bayes label transition matrices with knowledge transfer are estimated, we then apply a forward correction procedure similar to Figure 2.6, but to learn k Bayes label classifiers. Since our ultimate goal is to learn a single clean classifier, we aggregate all predictions of k Bayes label classifiers through majority voting.

Note that unlike from Stage 1, we do not apply the multi-task learning technique to Stage 2 to exploit potential task relatedness. The reason for this decision is that all labellings are produced against the same dataset, and intuitively, there should be only one clean classifier to learn. Hence, clean classifiers aim to be identical, which fails to meet the condition of task relatedness (tasks are related but not identical).

Experiments and Results

This chapter describes the experiments used to evaluate the effectiveness of the proposed MTL-T approach. We first present an overview of the methods used in our experiments. We then formulate the relevant experimental questions. Next, we detail our experiment setup, including the dataset, evaluation metrics, implementation details and ablation study. Finally, we analyse the empirical results and use them to answer our experimental questions.

4.1 Method Overview

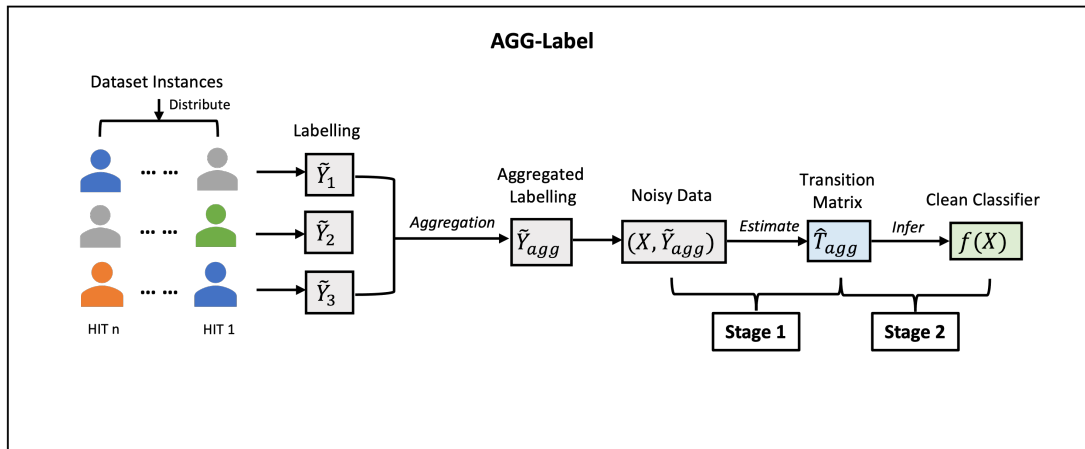


FIGURE 4.1. AGG-Label, the first group of baselines for the MTL-T approach

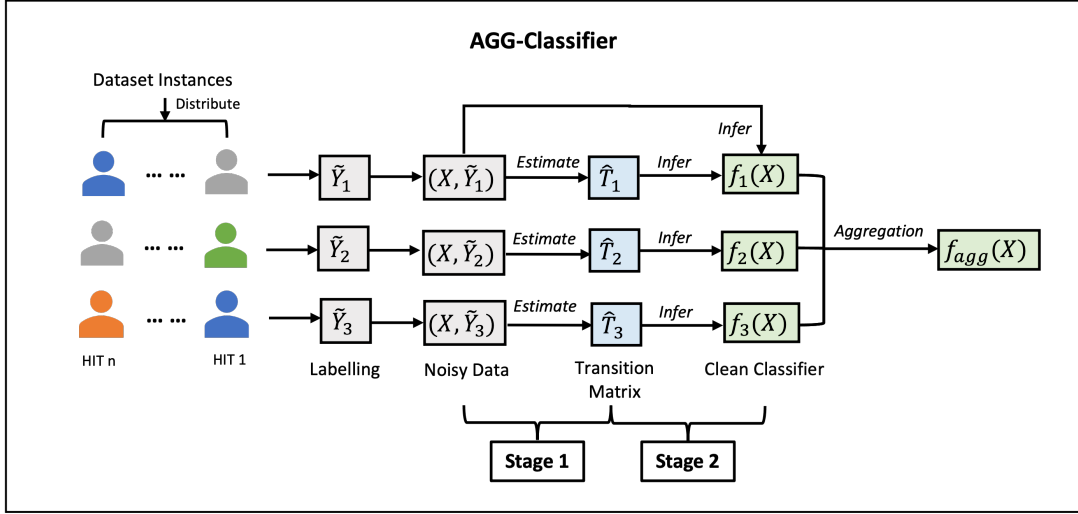


FIGURE 4.2. AGG-Classifier, the second baseline for the MTL-T approach

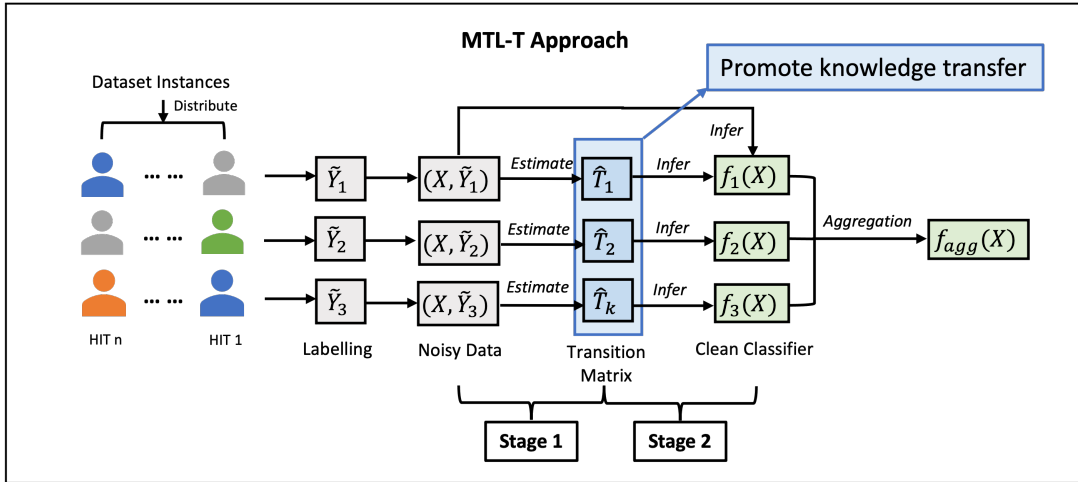


FIGURE 4.3. MTL-T, the proposed multi-task learning approach

For our experiment, we compare the proposed multi-task learning approach MTL-T against two baselines, AGG-Label and AGG-Classifier.

- **AGG-Label:** As illustrated in Figure 4.1, the key idea is to first aggregate all labellings to a single set of labels and then perform a typical two-stage training procedure to learn a clean classifier. To perform the label aggregation step, we compare eight representative algorithms in Table 2.1, yielding eight corresponding baselines.

- **AGG-Classifier:** As illustrated in Figure 4.2, the core idea is to first perform the two-stage training procedure for each labelling and in the end aggregate the clean classifiers' predictions using majority voting.
- **MTL-T:** As illustrated in Figure 4.3, the key difference between MTL-T and AGG-Classifier is that MTL-T exploits the task relatedness through knowledge transfer in Stage 1. Hence, MTL-T uses knowledge transfer to improve the estimation of each transition matrix. Further details of the MTL-T approach can be found in Chapter 3.

4.2 Experimental Questions

We aim to answer the following experimental research questions:

- (Q1) Does knowledge transfer occur between each transition matrix with MTL-T?
- (Q2) Does knowledge transfer in MTL-T improve the estimation of each transition matrix?
- (Q3) Does MTL-T produce a more accurate classifier than AGG-Classifier?
- (Q4) Does MTL-T produce a more accurate classifier than the aggregation methods in AGG-Label?

4.3 Experimental Setup

4.3.1 Dataset

We evaluate the performance of the proposed MTL-T approach over CIFAR-10N (Wei et al., 2021), a real-world noisy dataset obtained by re-annotating the CIFAR-10 training dataset three times (Krizhevsky et al., 2009).

The 50,000 training dataset is noisy, with three labellings produced by 3 randomly selected annotators from a total of 747 annotators. Each labelling has a noise rate of 18% (17.23% vs 18.12% vs 17.64%). The noise has been qualitatively and quantitatively demonstrated to be instance-dependent (Wei et al., 2021). By contrast, the 10,000 testing images are clean, which are used to evaluate the learned clean classifier's performance in predicting clean labels.

CIFAR-10N is the only large-scale (i.e. 10,000+) real-world noisy dataset of images with multiple labellings. Hence a key limitation of our experiments is that we only evaluate the performance on CIFAR-10N, which is discussed in Chapter 6.

4.3.2 Evaluation Metrics

To measure the effectiveness of proposed MTL-T approach, we focus on two metrics, prediction similarity and test classification accuracy.

Prediction Similarity. In the existing MTL literature, after applying the multi-task learning technique to exploit the relatedness, the most common practice is to directly measure the task performance, e.g., test classification accuracy. However, to the best of our knowledge, there seems to be no intuitive measure of knowledge transfer discussed in the literature. We propose prediction similarity, which is a domain-specific measure to intuitively demonstrate the before and after effect of knowledge transfer.

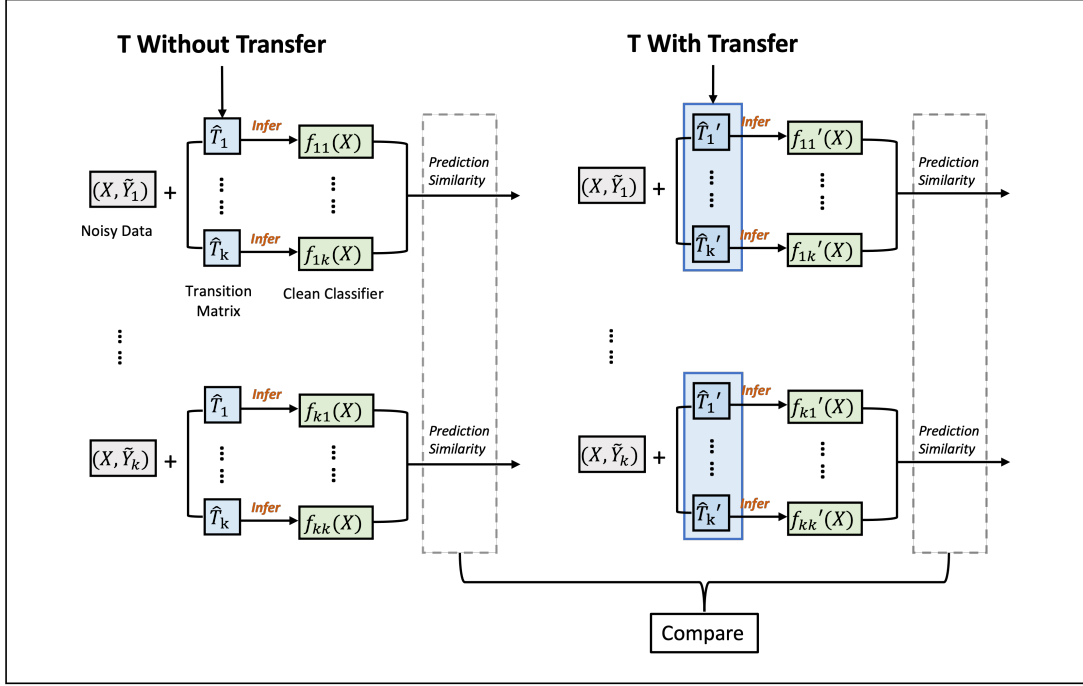


FIGURE 4.4. Prediction similarity computation

The computation process is illustrated in Figure 4.4. Assuming we have the same set of image instances X with k associated labellings $\tilde{Y}_1, \dots, \tilde{Y}_k$. To obtain prediction similarity, we first need to estimate all k transition matrices from their corresponding noisy data $(X, \tilde{Y}_1), \dots, (X, \tilde{Y}_k)$ both with and without knowledge transfer. This essentially produces two sets of transition matrices $\{\hat{T}_1, \dots, \hat{T}_k\}$ and $\{\hat{T}'_1, \dots, \hat{T}'_k\}$. Next, we fix each noisy data (X, \tilde{Y}_i) , and utilise these two sets of transition matrices learned with and without transfer to infer two corresponding sets of clean classifiers $\{f_{i1}(X), \dots, f_{ik}(X)\}$ and

$\{f'_{11}(X), \dots, f'_{1k}(X)\}$. Finally, we measure the prediction similarity of clean classifiers in each set and compare the prediction similarity for both sets.

The motivation is that if knowledge transfer does occur, all k transition matrices learned with the transfer should be more similar due to the propagation of shared knowledge. In this sense, for each fixed noisy data, the inferred k clean classifiers utilising the k more similar transition matrices should also share more prediction similarity than those without the transfer. Following such logic, we believe the prediction similarity can be an intuitive metric to measure knowledge transfer.

Test Classification Accuracy. The goal of label-noise learning algorithms is to train a clean classifier using the noisy training data, which can predict unknown clean labels accurately over the clean testing data. The accuracy of the trained clean classifier over clean testing data is known as test classification accuracy, which is a common metric to evaluate the label-noise learning algorithms performance. The higher the test classification accuracy, the more noise-tolerant the label-noise learning algorithm is.

Test classification accuracy can also be an *indirect* measure of transition matrix estimation quality. Intuitively, an accurately inferred clean classifier from the noisy data implies the effective "denoise" process, which is modelled and estimated as transition matrix.

Note that given we are learning from real-world label noise, the true transition matrix is unknown. Hence we cannot use the common transition matrix estimation error to evaluate our estimated transition matrices.

4.3.3 Implementation Details

All the experiments are implemented using PyTorch on NVIDIA Tesla V100 GPUs. Each experiment is performed 5 times and the average result is reported. For a more fair comparison, we align with existing work Wei et al. (2021) and use ResNet-34 (He et al., 2016b) as the backbone network for training.

AGG-Label. To produce the seven extra aggregated labellings besides the existing one in CIFAR-10N, we re-implement the open-sourced code by the recent crowdsourcing survey (Zheng et al., 2017). We also manually tune the hyperparameter ρ_{max} for the distilled dataset collection within the range of 0.1 and 0.5 and select the value with highest test classification accuracy of learned clean classifier. The reason is that in CIFAR-10N, the transition matrix estimation accuracy (reflected by the clean classifier accuracy) is found to be quite sensitive with regard to ρ_{max} .

AGG-Classifier. In Stage 1, the estimation of each Bayes label transition matrix, we adopt the same optimisation procedure as the original Bayes-T paper (Yang et al., 2021), i.e. optimiser SGD, with a momentum of 0.9 and a learning rate of 0.01. We also manually tune the hyperparameter ρ_{max} within the range of 0.1 and 0.5. In Stage 2, to learn each associated classification network, we adopt the Adam optimiser with an initial learning rate of 3×10^{-4} , The weight decay parameter is 5×10^{-4} , a mini-batch size is 128. Since the feature representation is learned and fixed, and we are only training a linear layer, the total number of epochs to train is set to be 20.

MTL-T. The implementation details are exactly the same as AGG-Classifier except that MTL-T tunes for extra three regularisation parameters λ_{fc_1} , λ_{fc_2} and λ_{fc_3} that control the strength of weight regularisation. We conduct grid search to tune them over candidate value range $[0, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}]$ using the noisy validation set. Given that the training time for each Bayes label transition matrix is rather fast (5 epochs only for around 200s), we tune all regularisation parameters over such range and select the hyper-parameter combination with highest noisy validation accuracy.

4.4 Ablation Study

We model the transition matrix using three linear layers. However, in order to further explore the effect of the number of linear layers, we perform an ablation study in this section.

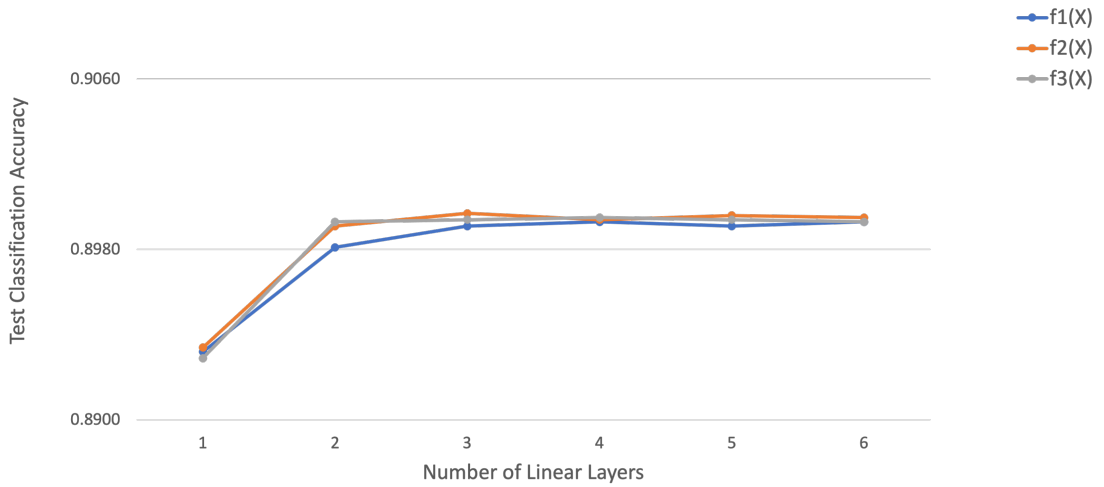


FIGURE 4.5. Sensitivity of $f_1(X)$, $f_2(X)$, $f_3(X)$'s classification accuracy towards number of linear layers

Specifically, we vary the number of linear layers from one to six, conduct the experiment 5 times and average the test classification accuracy of the three individual clean classifier $f_1(X)$, $f_2(X)$, $f_3(X)$. Each classifier is associated with the transition matrix learned with knowledge transfer (those in Figure 4.3). Judging from the accuracy illustrated in Figure 4.5, except when the number of linear layers is 1, the classification accuracy of each clean classifier remains fairly stable.

4.5 Results and Discussion

4.5.1 Occurrence of Knowledge Transfer (Q1)

We are interested in illustrating the effect of knowledge transfer in a more intuitive way and we choose prediction similarity as the evaluation metric. The details of its computation process, motivation and interpretation can be found in Section 4.3.2.

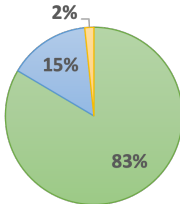
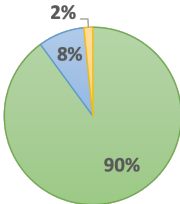
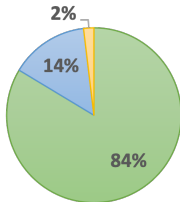
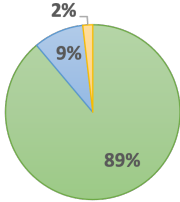
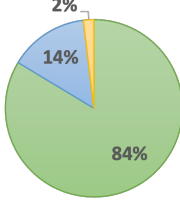
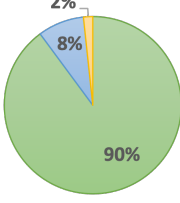
Case	Prediction Similarity		Proportion Change								
	Without Transfer	With Transfer									
Fixed (X, \tilde{Y}_1)			<table><tr><th>Category</th><th>Change (%)</th></tr><tr><td>All Same</td><td>+6.43</td></tr><tr><td>Mutually Same</td><td>−6.37</td></tr><tr><td>All Different</td><td>−0.06</td></tr></table>	Category	Change (%)	All Same	+6.43	Mutually Same	−6.37	All Different	−0.06
Category	Change (%)										
All Same	+6.43										
Mutually Same	−6.37										
All Different	−0.06										
Fixed (X, \tilde{Y}_2)			<table><tr><th>Category</th><th>Change (%)</th></tr><tr><td>All Same</td><td>+5.24</td></tr><tr><td>Mutually Same</td><td>−5.18</td></tr><tr><td>All Different</td><td>−0.06</td></tr></table>	Category	Change (%)	All Same	+5.24	Mutually Same	−5.18	All Different	−0.06
Category	Change (%)										
All Same	+5.24										
Mutually Same	−5.18										
All Different	−0.06										
Fixed (X, \tilde{Y}_3)			<table><tr><th>Category</th><th>Change (%)</th></tr><tr><td>All Same</td><td>+5.79</td></tr><tr><td>Mutually Same</td><td>−5.65</td></tr><tr><td>All Different</td><td>−0.14</td></tr></table>	Category	Change (%)	All Same	+5.79	Mutually Same	−5.65	All Different	−0.14
Category	Change (%)										
All Same	+5.79										
Mutually Same	−5.65										
All Different	−0.14										

FIGURE 4.6. Prediction similarity without and with knowledge transfer

As shown in Figure 4.6, by utilising the transition matrices learned with the knowledge transfer, the three inferred clean classifiers from the same noisy dataset become more similar in their predictions. For case of fixed noisy data, 5% to 6% of the classifier predictions turn from "Mutually Same" to "All Same" (i.e. more similar) after the knowledge transfer. Meanwhile, the proportion of "All Different" predictions generally remains the same both with and without knowledge transfer.

The experiment supports the hypothesis that due to the knowledge transfer, the propagated shared knowledge tends to make transition matrices more similar than those learned without the transfer. Hence, for fixed noisy data associated with the same labelling, the learned clean classifiers utilising those "more similar" transition matrices tend to be more similar in their predictions.

We note that, with knowledge transfer, the classifier predictions become more similar but *not* exactly the same. The result matches intuition since transition matrices model the label noise of a given labelling, each with their own unique noise generation process.

Overall, to answer Q1, the increased prediction similarity confirms that knowledge transfer occurs between transition matrices.

4.5.2 Impact of Knowledge Transfer in Transition Matrix Estimation (Q2)

We have identified the occurrence of knowledge transfer in Q1, and now we want to analyse if this knowledge transfer improves each task's performance, i.e., the estimation of each transition matrix. We measure the estimation quality using test classification accuracy of the learned clean classifier. Further details regarding evaluation metrics can be found in Section 4.3.2.

	Test Classification Accuracy		
	$f_1(X)$	$f_2(X)$	$f_3(X)$
AGG-Classifier (Without Transfer)	0.8934±0.13	0.8933±0.11	0.8941±0.15
MTL-T (With Transfer)	0.8991±0.23	0.8997±0.17	0.8994±0.19

TABLE 4.1. Test classification accuracy of the individual clean classifiers $f_1(X)$, $f_2(X)$, $f_3(X)$ with and without knowledge transfer

As shown in Table 4.1, with knowledge transfer, all individual clean classifiers learned $f_1(X)$, $f_2(X)$, $f_3(X)$ are more accurate than those without knowledge transfer. The accuracy uplift is consistent, ranging from

0.53% to 0.64%. The consistent improvement suggests that a better estimation of all underlying transition matrices is achieved, since improved estimation directly leads to more accurate inference of all clean classifiers.

Therefore, to answer Q2, the knowledge transfer does improve the estimation quality of each transition matrix.

4.5.3 Compare AGG-Classifier and MTL-T (Q3)

	Test Classification Accuracy
	$f_{agg}(X)$
AGG-Classifier (Without Transfer)	0.9074 \pm 0.18
MTL-T (With Transfer)	0.9128\pm0.21

TABLE 4.2. Test classification accuracy of the aggregated clean classifiers $f_{agg}(X)$ with and without knowledge transfer

For Q3, we compare the final aggregated clean classifier $f_{agg}(X)$ produced by aggregating individual clean classifiers $f_1(X)$, $f_2(X)$, $f_3(X)$ derived in the previous experiment. As shown in Figure 4.2, with knowledge transfer, the final aggregated clean classifier of MTL-T is more accurate than AGG-Classifier. The test classification accuracy gap (0.54%) is in line with the individual gaps measured in Q2, which are 0.57%, 0.64% and 0.53% respectively.

Therefore to answer Q3, MTL-T produces a more accurate final clean classifier than AGG-Classifier.

4.5.4 Compare AGG-Label and MTL-T (Q4)

Approach	Aggregation Algorithms	Test Classification Accuracy
AGG-Label	MV	0.8949 \pm 0.12
	ZC (Demartini et al., 2012)	0.8991 \pm 0.16
	GLAD (Whitehill et al., 2009)	0.8989 \pm 0.13
	CATD (Li et al., 2014a)	0.8992 \pm 0.11
	PM (Li et al., 2014b)	0.8973 \pm 0.13
	D&S (Dawid and Skene, 1979)	0.9014 \pm 0.09
	BCC (Kim and Ghahramani, 2012)	0.9021 \pm 0.07
	LFC (Raykar et al., 2010)	0.9025 \pm 0.10
MTL-T		0.9128\pm0.21

TABLE 4.3. Test classification accuracy of the clean classifiers from AGG-Label and MTL-T

For Q4, we want to compare if the MTL-T approach produces a more accurate classifier than any aggregation baselines in AGG-Label. Table 4.3 presents the test classification accuracy of clean classifiers produced by AGG-Label and MTL-T approaches over the CIFAR-10N dataset. Overall, MTL-T outperforms all AGG-Label baselines, with an accuracy uplift ranging from 1.03% to 1.79%.

As for individual baselines, LFC, BCC and D&S aggregation algorithms achieve an accuracy of over 90%. Meanwhile, the subsequent four aggregation algorithms, CATD, ZC, GLAD and PM, have relatively lower accuracy than the previous three, below 90% but higher than Majority Voting (MV).

The relative performance of each aggregation baseline is consistent with the literature, explained in Section 2.1.3. In particular, the techniques relying on the confusion matrix (LFC, BCC and D&S) tend to be more accurate than those that rely on worker probability modelling (GLAD, ZC and PM). Finally, MV achieves the lowest accuracy since it doesn't involve any modelling. .

Therefore to answer Q4, MTL-T produces a more accurate classifier than any of the aggregation baselines in AGG-Label.

Conclusion

In this thesis, we studied how to effectively learn a clean classifier from a real-world noisy dataset with multiple labellings. Motivated by the information loss resulting from label aggregation methods, we propose the MTL-T approach which directly learns from all labellings simultaneously. First, we model the label noise of each labelling as an instance-dependent transition matrix to more accurately model real-world label noise. Next, by identifying the potential task relatedness between transition matrices, we exploit this relationship with knowledge transfer, adapting appropriate multi-task learning techniques to the label noise setting. Our experimental results demonstrate that for CIFAR-10N, a real-world noisy dataset with multiple labellings, MTL-T successfully improves the estimation of each transition matrix with knowledge transfer, leading to better downstream clean classifiers to be learned.

The key contribution of this thesis is the insight that exploiting task relatedness between multiple transition matrices can improve the estimation of each transition matrix and ultimately, the accuracy of associated clean classifier.

Limitations and Future Work

The first key limitation of this thesis is that we have only conducted experiments over CIFAR-10N (Wei et al., 2021) without demonstrating that our methods generalise across different datasets. Unfortunately, in the field of label-noise learning, CIFAR-10N is the only real-world noisy dataset with *multiple* labellings. Further, in the field of crowdsourcing, there are also no appropriate datasets for our use-case, i.e., large real-world images datasets with multiple labellings, with enough training data to accurately estimate instance-dependent transition matrix. Relevant datasets such as the Dog dataset (Zhou et al., 2012) and the LabelMe dataset (Rodrigues et al., 2017) are too small, with the training set sizes being 804 and 1000 respectively. Hence if budget permits, future work could involve creating more datasets with multiple labellings. The creation can be done in a similar fashion as CIFAR-10N (Wei et al., 2021), where multiple crowdsourcing annotators are hired to re-annotate existing large-scale image datasets.

Another limitation is the assumption of the tensor normal prior. In the literature, there seems to be no evidence that real-world label noise follows a specific distribution. Although our empirical results are promising, the noise distribution assumption affects the confidence in our results, particularly since we only test on one dataset. Ideally, we would instead use techniques that do not rely on specific prior assumptions, such as the tensor-decomposition approach in Section 2.3.3.2. This approach could be used to decompose the same parameter tensor of all transition matrices into a shared component and an labelling-specific component. Both matrices are then regularised accordingly to enforce more sharing. However, the source code for many of these candidate methods are not publicly available, which makes experimental comparisons challenging. As future work, it may be beneficial to contact the authors and investigate more into alternative approaches.

Another key limitation concerns the aggregation baselines in AGG-Label, where they are considered to be the most representative algorithms in the literature rather than the state-of-the-art ones. One reason

we adopt this terminology is that there seems to be no universally best performing aggregation algorithms across different datasets, since each method requires different assumptions on the data, resulting in inconsistent performance (Zheng et al., 2017; Zhang et al., 2016a). Hence, selecting appropriate aggregation baselines is challenging, when seeking to conduct a fair comparison. Therefore, it would be beneficial to compare against a wider range of aggregation algorithms as future work.

Finally, as future work we may consider using transfer learning as an alternative method to exploit related knowledge and effectively estimate each transition matrix. Compared to multi-task learning, the key difference of transfer learning is that the direction of knowledge transfer flows from source tasks to the target task rather than between pairs of tasks. In the presence of multiple labellings, we may consider the transition matrix of one labelling as the target task and the remaining transition matrices as the source tasks. Then by utilising appropriate transfer learning techniques, we may improve the estimation quality of the target transition matrix.

Bibliography

- Rie Kubota Ando, Tong Zhang, and Peter Bartlett. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6(11).
- Andreas Argyriou, Massimiliano Pontil, Yiming Ying, and Charles Micchelli. 2007. A spectral regularization framework for multi-task structure learning. *Advances in neural information processing systems*, 20.
- Devansh Arpit, Stanisław Jastrzębski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, et al. 2017. A closer look at memorization in deep networks. In *International Conference on Machine Learning*, pages 233–242. PMLR.
- BJ Bakker and TM Heskes. 2003. Task clustering and gating for bayesian multitask learning.
- Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.
- Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. 2009. A convex formulation for learning shared structures from multiple tasks. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 137–144.
- Zhijun Chen, Huimin Wang, Hailong Sun, Pengpeng Chen, Tao Han, Xudong Liu, and Jie Yang. 2020. Structured probabilistic end-to-end learning from crowds. In *IJCAI*, pages 1512–1518.
- Jiacheng Cheng, Tongliang Liu, Kotagiri Ramamohanarao, and Dacheng Tao. 2020. Learning with bounded instance and label-dependent label noise. In *International Conference on Machine Learning*, pages 1789–1799. PMLR.
- Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the em algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):20–28.
- Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2012. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the 21st international conference on World Wide Web*, pages 469–478.
- Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.
- Theodoros Evgeniou and Massimiliano Pontil. 2004. Regularized multi-task learning. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 109–117.

- Pinghua Gong, Jiayu Zhou, Wei Fan, and Jieping Ye. 2014. Efficient multi-task feature learning with calibration. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 761–770.
- Bo Han, Quanming Yao, Tongliang Liu, Gang Niu, Ivor W Tsang, James T Kwok, and Masashi Sugiyama. 2020. A survey of label-noise representation learning: Past, present and future. *arXiv preprint arXiv:2011.04406*.
- Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. 2018. Co-teaching: Robust training of deep neural networks with extremely noisy labels. *arXiv preprint arXiv:1804.06872*.
- Lei Han and Yu Zhang. 2016. Multi-stage multi-task learning with reduced rank. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016a. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016b. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Laurent Jacob, Jean-philippe Vert, and Francis Bach. 2008. Clustered multi-task learning: A convex formulation. *Advances in neural information processing systems*, 21.
- Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International Conference on Machine Learning*, pages 2304–2313. PMLR.
- Yuan Jin, Mark Carman, Ye Zhu, and Yong Xiang. 2020. A technical survey on statistical modelling and design methods for crowdsourcing quality control. *Artificial Intelligence*, 287:103351.
- Zhuoliang Kang, Kristen Grauman, and Fei Sha. 2011. Learning with whom to share in multi-task feature learning. In *ICML*.
- Tsuyoshi Kato, Hisashi Kashima, Masashi Sugiyama, and Kiyoshi Asai. 2007. Multi-task learning via conic programming. *Advances in Neural Information Processing Systems*, 20.
- Hyun-Chul Kim and Zoubin Ghahramani. 2012. Bayesian classifier combination. In *Artificial Intelligence and Statistics*, pages 619–627. PMLR.
- Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25.
- Abhishek Kumar and Hal Daume III. 2012. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*.
- Qi Li, Yaliang Li, Jing Gao, Lu Su, Bo Zhao, Murat Demirbas, Wei Fan, and Jiawei Han. 2014a. A confidence-aware approach for truth discovery on long-tail data. *Proceedings of the VLDB Endowment*, 8(4):425–436.

- Qi Li, Yaliang Li, Jing Gao, Bo Zhao, Wei Fan, and Jiawei Han. 2014b. Resolving conflicts in heterogeneous data by truth discovery and source reliability estimation. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1187–1198.
- Tongliang Liu and Dacheng Tao. 2015. Classification with noisy labels by importance reweighting. *IEEE Transactions on pattern analysis and machine intelligence*, 38(3):447–461.
- Wu Liu, Tao Mei, Yongdong Zhang, Cherry Che, and Jiebo Luo. 2015. Multi-task deep visual-semantic embedding for video thumbnail selection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3707–3715.
- Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Philip S Yu. 2017. Learning multiple tasks with multilinear relationship networks. *Advances in neural information processing systems*, 30.
- Eran Malach and Shai Shalev-Shwartz. 2017. Decoupling" when to update" from" how to update". *arXiv preprint arXiv:1706.02613*.
- Andreas Maurer, Massi Pontil, and Bernardino Romera-Paredes. 2013. Sparse coding for multitask and transfer learning. In *International conference on machine learning*, pages 343–351. PMLR.
- Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. 2016. Cross-stitch networks for multi-task learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3994–4003.
- Michael Neumann and Ngoc Thang Vu. 2017. Attentive convolutional neural network based speech emotion recognition: A study on the impact of input features, signal length, and acted speech. *arXiv preprint arXiv:1706.00612*.
- Curtis G Northcutt, Anish Athalye, and Jonas Mueller. 2021. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749*.
- Giorgio Patrini, Alessandro Rozza, Aditya Krishna Menon, Richard Nock, and Lizhen Qu. 2017. Making deep neural networks robust to label noise: A loss correction approach. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1944–1952.
- Ting Kei Pong, Paul Tseng, Shuiwang Ji, and Jieping Ye. 2010. Trace norm regularization: Reformulations, algorithms, and multi-task learning. *SIAM Journal on Optimization*, 20(6):3465–3489.
- Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of machine learning research*, 11(4).
- Filipe Rodrigues, Mariana Lourenco, Bernardete Ribeiro, and Francisco C Pereira. 2017. Learning supervised topic models for classification and regression from crowds. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2409–2422.
- Yusuke Shinohara. 2016. Adversarial multi-task learning of deep neural networks for robust speech recognition. In *Interspeech*, pages 2369–2372. San Francisco, CA, USA.
- Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2022. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*.
- Sebastian Thrun and Joseph O’Sullivan. 1996. Discovering structure in multiple learning tasks: The tc algorithm. In *ICML*, volume 96, pages 489–497.

- Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. 2021. Learning with noisy labels revisited: A study using real-world human annotations. *arXiv preprint arXiv:2110.12088*.
- Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier Movellan, and Paul Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *Advances in neural information processing systems*, 22.
- Xiaobo Xia, Tongliang Liu, Bo Han, Nannan Wang, Mingming Gong, Haifeng Liu, Gang Niu, Dacheng Tao, and Masashi Sugiyama. 2020. Part-dependent label noise: Towards instance-dependent label noise. *Advances in Neural Information Processing Systems*, 33.
- Xiaobo Xia, Tongliang Liu, Nannan Wang, Bo Han, Chen Gong, Gang Niu, and Masashi Sugiyama. 2019. Are anchor points really indispensable in label-noise learning? *Advances in Neural Information Processing Systems*, 32:6838–6849.
- Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. 2015. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2691–2699.
- Ya Xue, Xuejun Liao, Lawrence Carin, and Balaji Krishnapuram. 2007. Multi-task learning for classification with dirichlet process priors. *Journal of Machine Learning Research*, 8(1).
- Qiang Yang, Yu Zhang, Wenyuan Dai, and Sinno Jialin Pan. 2020. *Multi-task Learning*, page 126–140. Cambridge University Press.
- Shuo Yang, Erkun Yang, Bo Han, Yang Liu, Min Xu, Gang Niu, and Tongliang Liu. 2021. Estimating instance-dependent label-noise transition matrix using dnns. *arXiv preprint arXiv:2105.13001*.
- Shuo Yang, Erkun Yang, Bo Han, Yang Liu, Min Xu, Gang Niu, and Tongliang Liu. 2022. Estimating instance-dependent bayes-label transition matrix using a deep neural network. In *ICML*.
- Yu Yao, Tongliang Liu, Bo Han, Mingming Gong, Jiankang Deng, Gang Niu, and Masashi Sugiyama. 2020. Dual T: reducing estimation error for transition matrix in label-noise learning. In Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.
- Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pages 7164–7173. PMLR.
- Jing Zhang, Xindong Wu, and Victor S Sheng. 2016a. Learning from crowdsourced labeled data: a survey. *Artificial Intelligence Review*, 46(4):543–576.
- Wenlu Zhang, Rongjian Li, Tao Zeng, Qian Sun, Sudhir Kumar, Jieping Ye, and Shuiwang Ji. 2016b. Deep model based transfer and multi-task learning for biological image analysis. *IEEE transactions on Big Data*, 6(2):322–333.
- Ye Zhang and Byron Wallace. 2015. A sensitivity analysis of (and practitioners’ guide to) convolutional neural networks for sentence classification. *arXiv preprint arXiv:1510.03820*.
- Yu Zhang and Qiang Yang. 2021. A survey on multi-task learning. *IEEE Transactions on Knowledge and Data Engineering*.

- Yu Zhang and Dit-Yan Yeung. 2012. A convex formulation for learning task relationships in multi-task learning. *arXiv preprint arXiv:1203.3536*.
- Yu Zhang and Dit-Yan Yeung. 2014. A regularization approach to learning task relationships in multitask learning. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 8(3):1–31.
- Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth inference in crowdsourcing: Is the problem solved? *Proceedings of the VLDB Endowment*, 10(5):541–552.
- Dengyong Zhou, Sumit Basu, Yi Mao, and John Platt. 2012. Learning from the wisdom of crowds by minimax entropy. *Advances in neural information processing systems*, 25.