A CNN-based Unified Framework Utilizing Projection Loss in Unison with Label Noise Handling for Multiple Myeloma Cancer Diagnosis

Shiv Gehlot[a], Anubha Gupta[a,*], Ritu Gupta[b,*]

[a]*SBILab, Department of ECE, IIIT-Delhi, New Delhi, 110020, India*
[b]*Laboratory Oncology Unit, Dr. B.R.A.IRCH, AIIMS, New Delhi 110029, India*

Abstract

Multiple Myeloma (MM) is a malignancy of plasma cells. Similar to other forms of cancer, it demands prompt diagnosis for reducing the risk of mortality. The conventional diagnostic tools are resource-intense and hence, these solutions are not easily scalable for extending their reach to the masses. Advancements in deep learning have led to rapid developments in affordable, resource optimized, easily deployable computer-assisted solutions. This work proposes a unified framework for MM diagnosis using microscopic blood cell imaging data that addresses the key challenges of inter-class visual similarity of healthy versus cancer cells and that of the label noise of the dataset. To extract class distinctive features, we propose projection loss to maximize the projection of a sample's activation on the respective class vector besides imposing orthogonality constraints on the class vectors. This projection loss is used along with the cross-entropy loss to design a dual branch architecture that helps achieve improved performance and provides scope for targeting the label noise problem. Based on this architecture, two methodologies have been proposed to correct the noisy labels. A coupling classifier has also been proposed to resolve the conflicts in the dual-branch architecture's predictions. We have utilized a large dataset of 72 subjects (26 healthy and 46 MM cancer) containing a total of 74996 images (including 34555 training cell images and 40441 test cell images). This is so far the most extensive dataset on Multiple Myeloma cancer ever reported in the literature. An ablation study has also been carried out. The proposed architecture performs best with a balanced accuracy of 94.17% on binary cell classification of healthy versus cancer in the comparative performance with ten

[*] Corresponding author
Email addresses: anubha@iiitd.ac.in (Anubha Gupta), drritugupta@gmail.com (Ritu Gupta)

This is the accepted version of the paper submitted to Medical Image Analysis. Please cite it as:
Shiv Gehlot, Anubha Gupta, Ritu Gupta. A CNN-based Unified Framework Utilizing Projection Loss in Unison with Label Noise Handling for Multiple Myeloma Cancer Diagnosis. Medical Image Analysis. 72, 102099. 2021. https://doi.org/10.1016/j.media.2021.102099.

state-of-the-art architectures. Extensive experiments on two additional publicly available datasets of two different modalities have also been utilized for analyzing the label noise handling capability of the proposed methodology.

Keywords: Multiple Myeloma, AI in Cancer diagnosis, Affordable AI in healthcare, Cell Classification, Deep Learning, Projection Loss, Label Noise

1. Introduction

Cancer occurs due to unconstrained cell division and can cause organs' dyscrasias. In 2018, there were 18.1 million estimated new cancer cases and 9.6 million deaths (Bray et al., 2018; The Global Cancer Observatory, 2020). The cancer death numbers are projected to be approx. 13 million by 2030 (Cancer Tomorrow, 2020). The cancer mortality rate is reported to be higher in low- and middle-income countries (LMICs). These countries shared 65% of global cancer deaths in 2012 that is estimated to increase to 75% by 2030 (Shah et al., 2019b). These statistics can be improved by expanding medical facilities including expeditious diagnosis that is crucial for moderating the mortality rate.

Lymphocytes are a type of white blood cells that are further categorized as T lymphocytes (T cells) and B lymphocytes (B cells). B cells counter the invaders' antigens and are transformed into the plasma cells in the process. These plasma cells usually reside in the bone marrow. Multiple myeloma (MM), a type of white blood cancer, occurs due to the unrestricted growth of these plasma cells (Multiple Myeloma, 2020). Some conventional and reliable tests for MM diagnosis include quantitative immunoglobulins, electrophoresis, and bone marrow biopsy (immunohistochemistry, flow cytometry, cytogenetics, and fluorescent in situ hybridization) (Multiple Myeloma, 2020). Such tests require costly medical infrastructure and a trained workforce, limiting the expansion of diagnostic facilities at the required scale in rural and urban areas.

Of late, considerable research is being undertaken to develop computer-assisted diagnostic (CAD) tools for healthcare. If accurate enough, these tools can be deployed to aid medical professionals and can mitigate the requirement of expensive specialized resources. Thus, CAD tools can act as significant enablers in scaling the necessary and affordable diagnostic facilities. In literature, two approaches are frequently used for CAD tool development: 1) using the traditional machine learning classifiers such as support vector machine, naïve Bayes, decision tree, random forest, etc., and 2) using the deep learning models, say, convolutional neural networks (CNNs). As compared to the CNN-based tools, traditional classifiers require a relatively smaller dataset. However, these classifiers' performance depends on input features extracted manually using the apriori information such as the cytoplasm or nucleus structure. These predefined features may not be the best to work with, leading to sub-optimal performance.

In the context of blood cancers, Mohapatra et al. (2011); Madhukar et al. (2012); Joshi et al. (2013); Mohapatra et al. (2014); Putzu et al. (2014); Chatap & Shibu (2014); Reta et al. (2015); Neoh et al. (2015); Vincent et al. (2015);

Table 1: A brief summary of some of the methods utilizing CNNs for cell classification and cancer diagnosis. C.V.: cross validation, T.S.: training from scratch.

Reference	**Task**	**Approach**	**Architecture**
Han et al. (2016)	HEp-2 Cell Classification	CNN (T.S.)	CaffeNet
Bayramoglu et al. (2016)	Breast Cancer Classification in Histopathology Images	CNN (T.S.)	Custom CNN
Gao et al. (2017)	HEp-2 Cell Classification	CNN (T.S.)	LeNet based CNN
Sirinukunwattana et al. (2016)	Colon Cancer Histology Images Classification	CNN (T.S.)	Custom CNN
Meng et al. (2019)	Cell Classification ATOM images	CNN (T.S.)	Custom CNN
Qin et al. (2018)	Leukocyte Classification	CNN (T.S.)	Custom CNN
Chang et al. (2017)	Cancer Cell Classification in Pancreas Histological Images	CNN (T.S.)	Custom CNN
Sharma et al. (2017)	Classification of Gastric Carcinoma Histopathological Images	CNN (T.S.)	Custom CNN
Gehlot et al. (2020b)	ALL Classification in Microscopic Images	CNN (T.S.)	Custom CNN
Xu et al. (2015)	Brain Tumor Classification in Histopathology Images	Transfer learning (features extraction)	AlexNet
Phan et al. (2016)	HEp-2 Cell Classification	Transfer learning (features extraction)	AlexNet based CNN
Bayramoglu & Heikkilä (2016)	Nuclei Classification in Histopathological Images	Transfer learning (fine tuning)	AlexNet, GoogleNet, VGG-16, GenderNet
Tabibu et al. (2019)	Renal Cell Carcinoma (RCC) Histopathological Image Classification	Transfer learning (fine tuning)	Resnet-18 & Resnet-34
Han et al. (2018b)	Classification of Cutaneous Tumors	Transfer learning (fine tuning)	ResNet-152
Harangi (2018)	Skin lesion classification	Transfer learning (fine tuning)	GoogLeNet, AlexNet, ResNet-152, VGGNet
Zhang et al. (2017)	Cervical Cell Classification	Transfer learning (fine tuning)	CaffeNet
Jiang et al. (2017)	Breast Cancer Classification in Mammograms	Transfer learning (fine tuning)	GoogLeNet, AlexNet
Esteva et al. (2017)	Skin Cancer Classification	Transfer learning (fine tuning)	Inception-v3
Hekler et al. (2019)	Histopathological Melanoma Image Classification	Transfer learning (fine tuning)	ResNet50
Mazo et al. (2018)	Cardiovascular Tissue Classification in Histological Images	Transfer learning (fine tuning)	VGG16, VGG19, ResNet, Inception

Kazemi et al. (2015); Patel & Mishra (2015); Amin et al. (2016a); Singhal & Singh (2016); Amin et al. (2016b); Rawat et al. (2017a,b); Karthikeyan & Poornima (2017); Mishra et al. (2017, 2019) have utilized conventional classifiers. Besides the limitation of using hand-crafted features, these methods have used very small datasets (19-267 images) for training and evaluating the test performance. The tools designed with such datasets may not be reliable for deployment in practical scenarios due to the large-scale heterogeneity within and across subjects' data in real-life.

On the other hand, CNNs eliminate the necessity and limitation of extracting manual features and facilitate task-dependent automatic feature extraction. The use of CNNs in the medical domain has seen a rapid surge in recent years (Deng et al., 2020; Litjens et al., 2017). However, training of CNNs from scratch requires a large dataset depending upon the depth of the CNN. The availability of a large annotated dataset for supervised learning is a challenge in the medical domain. An alternate solution is to use transfer learning, wherein a network trained on one dataset (pre-trained network) is used on another dataset. In one approach of transfer learning, a pre-trained network is used directly for feature extraction, while in another approach, a pre-trained network is fine-tuned on an available dataset. An overview of the works utilizing CNNs for cell classification and cancer diagnosis with all of the above three approaches is provided in Table 1. It is observed from Table 1 that CAD tools have targeted a broad class of cancer diagnosis. Also, training from scratch and transfer learning have been deployed frequently to classify different types of cancers. Although direct feature extraction eliminates the need for a training set, it is a less preferred approach, as seen from Table 1, because fine-tuning usually performs better than direct feature extraction.

For blood cancer cell classification, fine-tuning has been performed on AlexNet by Rehman et al. (2018) and Shafique & Tehsin (2018), while Vogado et al. (2017) and Vogado et al. (2018) extracted features directly from CNNs and used SVM or other classifiers later. However, very small datasets have been used in these studies, say of 108, 310, 330, and 891 images by Vogado et al. (2017), Rehman et al. (2018), Shafique & Tehsin (2018), and Vogado et al. (2018), respectively. Fine-tuning approach on a large dataset of B-ALL cancer, consisting of 12528 cell images for training and 2586 cell images for testing (Gupta & Gupta, 2019a), has been carried out on different architectures (Gupta & Gupta, 2019b) by Pan et al. (2019); Verma & Singh (2019); Prellberg & Kramer (2019); Xiao et al. (2019); Shi et al. (2019); Liu & Long (2019); Shah et al. (2019a); Ding et al. (2019); Xie et al. (2019). For example, pretrained ResNet incorporating label correction is employed by Pan et al. (2019). Similarly, fine-tuning of ResNetXt50 with a layer-dependent learning rate is used by Prellberg & Kramer (2019). Xiao et al. (2019) utilized a pseudo labeling approach with ensembling of pretrained architectures. The ensembling is also used by Shah et al. (2019a) and Ding et al. (2019). Verma & Singh (2019) used MobileNet; Liu & Long (2019) and Xie et al. (2019) used Inception ResNet, while Goswami et al. (2020) fine-tuned a pretrained Inception-v3 on the above mentioned dataset using a newly defined heterogeneity loss function. Besides using class centers, this loss function assigns a separate center to each subject and attempts to capture the inter-class and inter-subject distinguishable characteristics.

The classification networks used for transfer learning are generally pre-trained on the ImageNet, a large 1000 class non-medical images' dataset. For transfer learning, the target medical images are required to match these pre-trained networks' input image size. This requires a suitable scaling of input medical images. This scaling may change the morphology of the medical constituents of images, say cells, and hence, can hurt the classifier's overall performance. Moreover, these networks may be undesirably massive for medical applications because the medical domain, in general, does not encounter these many classes as are present in the ImageNet dataset (Wong et al., 2018).

Duggal et al. (2017) trained AlexNet and T-CNN from scratch for leukemia diagnosis. The trained architectures were then fine-tuned after including the trainable stained deconvolutional (SD) layer. However, a significant downside of this method is the aggregation of the dataset of all the subjects. This leads to the training and testing on the same subjects' data that is not the case in practical deployment, wherein a tool developed using a set of the subjects is to be tested on the prospective unseen subjects. Gehlot et al. (2020b) has addressed this issue by segregating the dataset at the subject-level such that there is no common subject between training and test datasets. The method includes utilizing a combination of SD and DCT layers pre-appended to a compact CNN architecture to aid the extraction of distinctive features from the visually similar classes. The resultant architecture is then trained from scratch. Also, an ensembling approach utilizing an auxiliary classifier has been used to boost the classifier's overall performance.

In this work, we have also employed the technique of training a custom CNN architecture from scratch to target the problem of multiple myeloma (MM) cancer diagnosis. The proposed approach includes a novel loss function, a label noise handling method, and an ensembling approach. Also, we have used a large dataset for training and testing purposes. Specifically, we have used a total of 72 subjects' data (26 healthy subjects and 46 MM cancer patients) divided into 34555 training cell images of 46 subjects and 40441 test cell images of 26 subjects. To the best of our knowledge, no other work has utilized such a large dataset for any blood cancer diagnosis. The salient contributions of this works are listed as below:

1. A novel projection loss utilizing class-specific vectors has been proposed to achieve inter-class separation by maximizing the projection between the activation and the respective class vectors. The proposed loss also constrains the class vectors to be orthogonal to each other.

2. A dual-branch architecture is used to accommodate projection loss in combination with cross-entropy loss to achieve enhanced performance. The architecture also employs two different feature pooling to capture the discerning features in multiple ways.

3. Two label noise handling approaches utilizing the dual-branch architecture has been proposed to address the training samples label noise. The proposed approaches are unsupervised from the perspective that no label information is required.

4. A coupling classifier is proposed that resolves the ambiguity and predicts a unique label from the dual-branch architecture. This coupling classifier uses distinct sets of features from the two branches.

5. The test performance has been evaluated using a large multiple myeloma (MM) dataset of 40441 images. An ablation study highlighting the pro-

posed contributions along with the subject-level analysis has also been provided on the test dataset. Two other datasets (Camelyon7 and TBX11K) have also been used to validate the proposed label noise handling approach.

2. Materials

We have used three datasets for the experiments. In this section, we provide a detailed description of all the three datasets.

2.1. Multiple Myeloma (MM) Dataset

The dataset is collected using the slides prepared from the bone marrow aspirate of the healthy and cancer subjects using the standard procedure, including staining slides with the Jenner-Giemsa stain. The stain is used for highlighting the bone marrow cells, including the plasma cells, i.e., the cells of interest. Subsequently, the slides are imaged in the .bmp format using the camera mounted on the microscope. The captured microscopic images are of size 2040×1536 pixels and contain the cells of interest annotated by the expert oncologists. These cells are segmented from the images using an in-house deep learning-based segmentation tool (Gehlot et al., 2020a). Each segmented image contains only a single cell centered at the origin. The segmented cell images are also zero-padded to achieve a fixed spatial size of 350×350, ensuring the containment of cells of varying sizes. The cancer class samples are collected from the patients diagnosed with multiple myeloma, whereas samples of the healthy class are from subjects not suffering from cancer of plasma cells.

The dataset is collected from 72 subjects (26 healthy subjects and 46 MM cancer patients), out of which 46 subjects' data is used for training, and 26 subjects' data is used for testing. Out of the 46 subjects in the training set, 26 subjects belong to the MM cancer class, and the remaining 20 are healthy. These subjects are divided into five-folds, such that the entire data of one subject is present in one fold only. In total, there are 34555 images in the train set and 40441 images in the test set. The detailed description of the dataset is provided in Table 2. Also, the sample images from both classes are shown in Fig. 2. The test set size is sufficiently large for the fair observation of the classifier's performance. The dataset was collected at the Laboratory Oncology, AIIMS, New Delhi, India, after the Ethics Committee's approval. The subjects' confidentiality was maintained during the data collection process. Only one of the co-authors had access to the subject-specific information, which was entirely removed before sharing the data for experiments.

Challenges of the dataset. From Fig. 2, it is evident that the images of both the classes are visually similar. Moreover, there is a stain variation in the cell images of data of different subjects. In general, stain color variation occurs owing to multiple reasons (Gupta et al., 2020). Since this data is collected over a period of two years, the significant reason for stain variation is the use of different manufactured batches of the staining chemical to prepare the slides.

Table 2: Data Description.
Number of subjects and images in the training and test sets. Distribution of subjects and images in different folds of training data is mentioned. The folds have been prepared such that almost 1:1 ratio of the data of healthy and cancer class is maintained, while a subjects' data is present in only one fold.

Training set

	Folds	1	2	3	4	5
No. of subjects	Cancer Class	5	5	5	5	6
	Healthy Class	4	4	4	3	5
	Total no. of subjects in fold	9	9	9	8	11
No. of Images	Cancer Class	3456	3134	3679	3673	3471
	Healthy Class	3150	3428	3606	3870	3088
	Total no. of images in fold	6606	6562	7285	7543	6559
Total no. of subjects	46					
Total no. of images	34555					

Test set

No. of subjects	Cancer Class	20
	Healthy Class	6
	Total no. of subjects	26
No. of Images	Cancer Class	19366
	Healthy Class	21075
	Total no. of images	40441

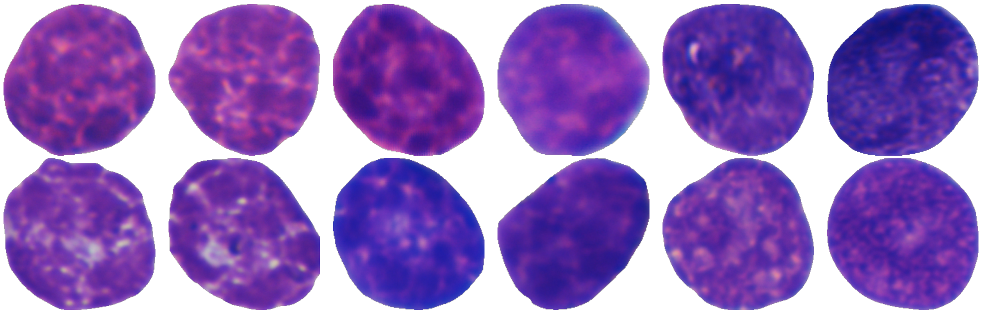


Figure 1: Sample images from the cancer class (first row), and the healthy class (second row). Samples of three subjects of each class have been shown.

Although stain normalization can be used before cell segmentation/classification (Gupta et al., 2020), we have eliminated this step to make the problem more challenging.

2.2. Camelyon17 (Bndi et al., 2019)

It is a publicly available dataset and consists of 50 annotated whole slide images (WSIs) collected from five different centers (C0-C4). The WSIs have captured H&E stained slides prepared from the lymph node sections. We have extracted 128 × 128 pixels patches from WSIs of center-3 (C3) for experiments.

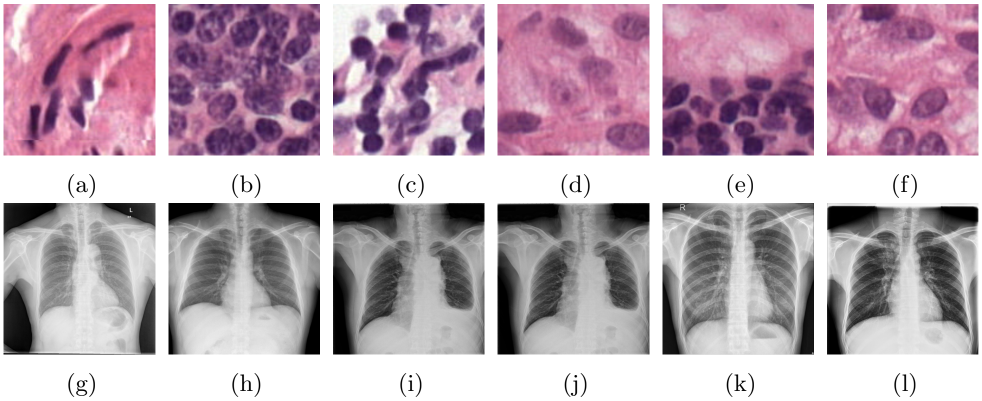


Figure 2: Sample images from the Camelyon17 (first row): (a-c) Healthy samples, and (d-f) Tumor samples. TBX11K (second row): (g-h) Healthy samples, (i-j) Sick & Non-TB samples, and (k-l) TB samples.

Table 3: Dataset description for Camelyon17 (Bndi et al., 2019) and TBX11K (Liu et al., 2020). Test set GT is not available for TBX11K.

	Camelyon17			**TBX11K**			
Splits/Classes	Tumor	Healthy	Total	TB	Sick & Non-TB	Healthy	Total
Train	37135	37498	74633	759	3001	3129	6889
Val	7500	7500	15000	355	799	933	2087
Test	20004	20001	40005	-	-	-	3302

From the collected patches, 74,633 are used for training, 15000 for validation, and the remaining 40005 for testing. We have performed a classification task on the resultant dataset wherein the patches containing metastatic tumor cells are annotated as cancer and healthy otherwise.

2.3. TBX11K (Liu et al., 2020)

It is a Tuberculosis X-ray dataset having images of 512 × 512 pixels. Each X-ray image is annotated as healthy or sick & non-TB or TB. The dataset also contains the images from Chauhan et al. (2014) and Jaeger et al. (2014). This results in the training set of 6889 images, a validation set of 2087 images, and a test set of 3302 images. The GT is available only for training and validation sets and not for the test set. There is a challenge (TBX11K Tuberculosis Classification and Detection Challenge, 2020) available on this dataset, and the test set performance can be evaluated through the challenge portal. The challenge consists of two tasks; TB detection and classification. We have performed analysis only on the classification task.

3. Methods

In this section, we will discuss the proposed projection loss, architecture, and noisy label handling method.

3.1. Projection Loss

We denote the d-dimension class vectors as $\mathbf{c}_i$ such that $i \in [0, C-1]$, where C is the total number of classes. Also, let y_i represents the label of class i. Let $\boldsymbol{a}_k$ be the activation obtained from the last layer of a CNN for the sample $\mathbf{X}_k$. We define the projection of $\boldsymbol{a}_k$ on $\mathbf{c}_i$ as:

$$p_{k,i} = \frac{\boldsymbol{a}_k^T \mathbf{c}_i}{||\mathbf{c}_i||_2}. \tag{1}$$

In an ideal scenario, if $\mathbf{X}_k \in y_i$, then $p_{k,i} = 1$ and if $\mathbf{X}_k \in y_{j \neq i}$, then $p_{k,i} = 0$. The above interpretation can be modeled through a distance d as:

$$\begin{align} d(\boldsymbol{a}_k, \mathbf{c}_i) &= \left|\left| \frac{\boldsymbol{a}_k^T \mathbf{c}_i}{||\mathbf{c}_i||_2} - 1 \right|\right|_2^2, \tag{2} \\ &= ||p_{k,i} - 1||_2^2 \, . \tag{3} \end{align}$$

The distance $d(\boldsymbol{a}_k, \mathbf{c}_i)$ should be ideally equal to 0, if $\mathbf{X}_k \in y_i$ and should be 1 otherwise. Hence, $d(\boldsymbol{a}_k, \mathbf{c}_i)$ should be minimum for $\mathbf{X}_k \in y_i$ and maximum otherwise. The conditional probability $P(y_i|\mathbf{X}_k; \boldsymbol{w}, \mathbf{c}_i)$ is represented in terms of the distance $d(\boldsymbol{a}_k, \mathbf{c}_i)$ as:

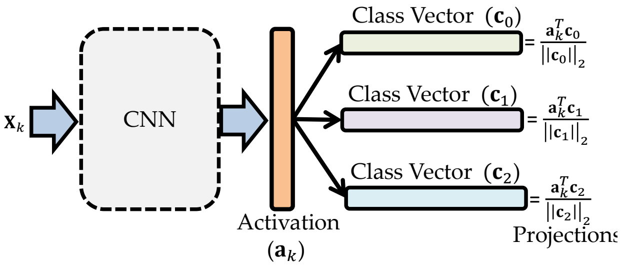
$$P(y_i|\mathbf{X}_k; \boldsymbol{w}, \mathbf{c}_i) = \frac{e^{-d(\boldsymbol{a}_k, \mathbf{c}_i)}}{\sum_{i=0}^{C-1} e^{-d(\boldsymbol{a}_k, \mathbf{c}_i)}}, \tag{4}$$

where the activation vectors $\boldsymbol{a}_k$s are functions of weights $\boldsymbol{w}$. The maximization of $P(y_i|\mathbf{X}_k; \boldsymbol{w}, \mathbf{c}_i)$ demands the maximization of the numerator in (4), which results in the minimization of $d(\boldsymbol{a}_k, \mathbf{c}_i)$. Without loss of generality, we introduce a variable β in (4), resulting in:

$$P(y_i|\mathbf{X}_k; \boldsymbol{w}, \mathbf{c}_i) = \frac{e^{-\beta d(\boldsymbol{a}_k, \mathbf{c}_i)}}{\sum_{i=0}^{C-1} e^{-\beta d(\boldsymbol{a}_k, \mathbf{c}_i)}}, \tag{5}$$

where β (Yang et al., 2018) acts as a scaling factor. The loss can then be formulated as

$$\mathcal{L}'(\boldsymbol{w}, \mathbf{c}) = -\sum_{i=0}^{C-1} y_i \left[log(P(y_i|\mathbf{X}_k; \boldsymbol{w}, \boldsymbol{c}_i)\right] \tag{6}$$



$\mathbf{X}_k$
CNN
Activation
$(\mathbf{a}_k)$
Class Vector $(\mathbf{c}_0)$
$= \frac{\mathbf{a}_k^T\mathbf{c}_0}{||\mathbf{c}_0||_2}$
Class Vector $(\mathbf{c}_1)$
$= \frac{\mathbf{a}_k^T\mathbf{c}_1}{||\mathbf{c}_1||_2}$
Class Vector $(\mathbf{c}_2)$
$= \frac{\mathbf{a}_k^T\mathbf{c}_2}{||\mathbf{c}_2||_2}$
Projections

Figure 3: Projections of the activation on the class vectors.

Orthogonality of the class vectors $\mathbf{c}_i$

Apart from maximizing the projections of the sample on the respective class vectors, we also induce a orthogonality constraint on the class vectors, i.e., $\mathbf{c}_i^T\mathbf{c}_{j\neq i} = 0$. For this, we introduce a regularization term $\mathcal{L}_{orth}(\cdot)$ given by

$$\mathcal{L}_{orth}(\mathbf{c}) = \lambda \sum_{j=i+1}^{C-1} \sum_{i=0}^{C-2} ||\mathbf{c}_i^T\mathbf{c}_j||_2^2. \tag{7}$$

This constraint helps in a better separation of classes. Combining (6) and (7), we obtain the overall loss function given by

$$\begin{aligned} \mathcal{L}_{PRL}(\boldsymbol{w}, \mathbf{c}) &= -\sum_{i=0}^{C-1} y_i \left[log(P(y_i|\mathbf{X}_k; \boldsymbol{w}, \boldsymbol{c}_i)\right] + \lambda \sum_{j=i+1}^{C-1} \sum_{i=0}^{C-2} ||\mathbf{c}_i^T\mathbf{c}_j||_2^2, \\ \mathcal{L}_{PRL}(\boldsymbol{w}, \mathbf{c}) &= \mathcal{L}'(\boldsymbol{w}, \mathbf{c}) + \lambda\mathcal{L}_{orth}(\mathbf{c}). \end{aligned} \tag{8}$$

Intuitively, the projection loss helps to maximize the projection of the learned activations on the learnable class vectors and also attempts to induce the orthogonality among the class vectors. The update equations for $\boldsymbol{w}$ and $\mathbf{c}$ are obtained from (8) as follows:

$$\boldsymbol{w} = \boldsymbol{w} - \alpha \frac{\partial \mathcal{L}'}{\partial \boldsymbol{a}} \frac{\partial \boldsymbol{a}}{\partial \boldsymbol{w}}, \tag{9}$$

$$\mathbf{c}_i = \mathbf{c}_i - \alpha \left(\frac{\partial \mathcal{L}'}{\partial \mathbf{c}_i} + \lambda \frac{\partial \mathcal{L}_{orth}}{\partial \mathbf{c}_i} \right), \tag{10}$$

where α is the learning rate. Hence, the class vectors $(\mathbf{c}_i)$ are updated through both the terms.

3.2. BaseCE-Net

As a starting point, we design a custom CNN classification network instead of using any existing pretrained architecture. The network consists of ten *Conv Sections*, where each *Conv Section* consists of a combination of 2D convolution filters, batch normalization, and parametric ReLu (PReLu) as an activation function. There is no max-pooling in the network; instead, stride≥2 is used to achieve spatial size reduction. This also helps the network to learn the required pooling operation. After the last *Conv Section*, the output features are given to a global averaging pooling (GAP) layer and finally passed to the output softmax layer. We name this architecture *BaseCE-Net Network* because we use binary cross-entropy loss function to train it.

3.3. BasePRL Net

Next, we replace the BCE loss with the proposed projection loss ($\mathcal{L}_{PRL}$) in the BaseCE-Net network and name this architecture as *BasePRL-Net.* As we have a binary class dataset, we initialize the two-class vectors $\mathbf{c}_0$ and $\mathbf{c}_1$ for class 0 (healthy) and 1 (cancer), respectively. The output of GAP is projected on the class vectors $\mathbf{c}_0$ and $\mathbf{c}_1$, where (8) is used as the objective function. The output of GAP and the class vectors are of the same dimension. During the training, $\mathbf{c}_0$ and $\mathbf{c}_1$ are also updated along with the network parameters according to (10).

3.4. PRLCE-Net

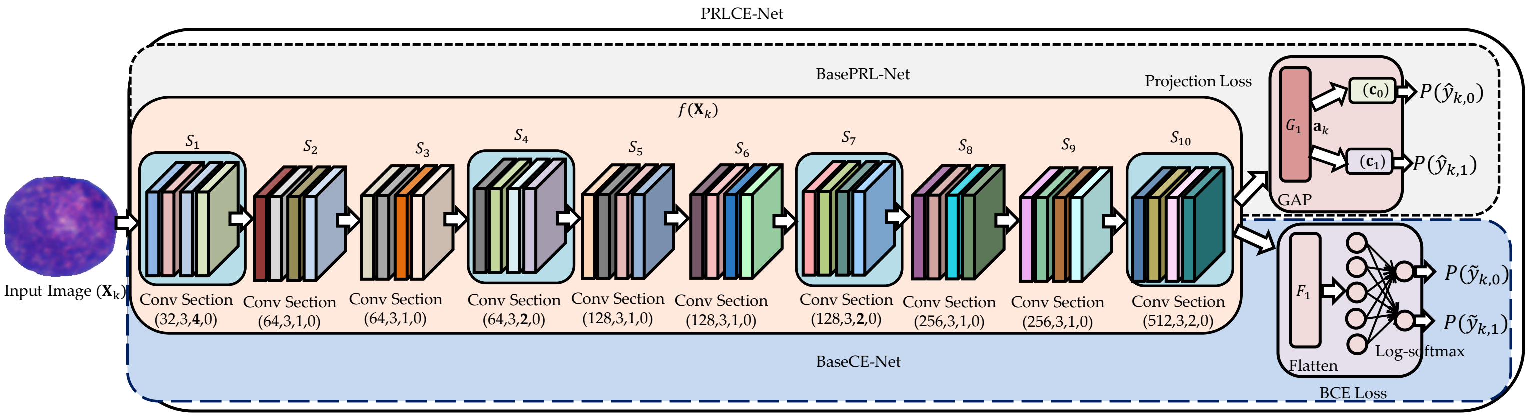
We design a hybrid architecture that uses both BCE loss and Projection Loss (PRL). However, instead of directly adding BCE loss and PRL, we combine the BaseCE-Net and BasePRL-Net such that the new architecture has shared convolution filters (*conv sections)* for both the objectives. After that, the network is fragmented into two branches. One branch is flattening (reshaping) the input features to use with the BCE loss, while the other branch is applying GAP on the incoming features and then using the PRL on the resultant output. Different pooling layers help capture different structures of the data that helps to utilize different information by each branch. The *PRLCE-Net* is shown in Fig. 4 and its loss function is given by

$$\mathcal{L}_{PR-CE} = \beta_1 \mathcal{L}_{PR} + \beta_2 \mathcal{L}_{CE} \tag{11}$$

As observed from Fig. 4, the predictions for the two loss functions, $\mathcal{L}_{PR}$ and $\mathcal{L}_{CE}$, are obtained differently. During the backpropagation, shared convolutional filters (feature extraction filters) will be updated with two different objective functions. Since both the loss functions are attempting to perform a common task, albeit with different approaches, the resultant weights will lead to more robust feature extraction and better final performance. This claim has been verified in Sec 4.1.2. One of the advantages of *PRLCE-Net* is that it provides scope for inducing robustness. As discussed in Sec 3.5 and Sec 3.6, we introduce ensembling and noise handling capability in *PRLCE-Net Network* to enhance its performance.

3.5. Label Noise Handling

Medical datasets often suffer from the problem of noisy labels due to several reasons such as decision ambiguity, variations in the acquisition process, etc. Unlike the natural images, samples with noisy labels in medical datasets can not be identified manually, even for small datasets, due to the inter-class visual



Conv Section (N, F, S, P): Convolution2D with number of filters 'N', filter size 'F×F', stride 'S', and Padding 'P' followed by Batch Normalization and Parametric ReLu (PReLu) as the activation. GAP: Global Averaging Pooling. Sections having stride > 1 are highlighted in the box.

Figure 4: PRLCE-Net.
It uses CE and projection loss in the two branches.

similarity. This makes the problem more challenging in medical data. At the same time, handling the noisy labels appropriately may improve the model's performance. The label noise in the medical images have been addressed through label cleaning, noise layer, loss functions, data re-weighting, and training procedures (Karimi et al., 2020). Among the label cleaning based approaches, Veit et al. (2017) trained two CNNs, one using the clean data for learning to denoise data to be used by the another CNN. However, this approach also requires clean data, which may not be available in some scenarios. Another approach that requires clean data is discussed in Lee et al. (2018). In this work attention utilizing encoder is used to generate the embedding vector of each class. In parallel, another encoder is used to generate the query image's embedding vector. The similarity between query and reference embedding vector is used to predict the samples with the label noise. Han et al. (2018a) eliminates the necessity of clean data. It maintains two CNNs, and the clean labels identified by one network based on loss criteria are used to update the peer CNN parameters. The proposed approach also does not require a clean dataset. In contrast to Han et al. (2018a), our approach utilizes only a single CNN and two different loss functions. We have also used different label noise identification criteria that are not based on loss function but on the predictions of two different branches. The training procedure is also different. While the training of Han et al. (2018a) consists of only one stage, our approach is based on two-steps training. The proposed technique does not require any prior knowledge of class labels and is unsupervised from this perspective. Consider the *PRLCE-Net* trained with $\{\mathbf{X}_k, y_k\}_{k=0}^{N-1}$ samples for T epochs. At any given epoch, the label predicted by

PRL branch for $\mathbf{X}_k$ is given by

$$\hat{y}_k = \underset{l \in \{0,1\}}{\arg\max}\, P(\hat{y}_{k,l}). \tag{12}$$

This can be easily extended to C classes, wherein there will be C class vectors with $l \in \{0, 1, ..C-1\}$. Similarly, for the CE branch, the prediction for $\mathbf{X}_k$ is

$$\tilde{y}_k = \underset{l \in \{0,1\}}{\arg\max}\, P(\tilde{y}_{k,l}). \tag{13}$$

Let $P(\hat{y}_k)$ and $P(\tilde{y}_k)$ denote the probability scores of the predicted labels. As the training of a CNN progresses, i.e., as more numbers of epochs are completed, the networks' performance on the training set improves. Accordingly, the performance on the validation set will improve if there is no overfitting. Let the total epochs T be divided into two sets: $\{0, 1, ..P-1\}$ and $\{P, P+1..T-1\}$. The number P is chosen to be sufficiently large for the convergence of the training process. This parameters will also be updated such that both the PRL branch and CE branch yield an optimal performance. We hypothesize that even though the two branches use different loss functions, both will give approximately the same performance, at least on the non-noisy training data. After the $(P-1)^{th}$ epoch, there are three possible scenarios for a training sample $\mathbf{X}_k$:

(S_1) $\hat{y}_k = \tilde{y}_k$ i.e., predictions of the branches are same.

(S_2) $\hat{y}_k \neq \tilde{y}_k$ with $P(\hat{y}_k) \geq \tau$ **and/or** $P(\tilde{y}_k) \geq \tau$ i.e both branches are predicting a different class but at least one of them has the prediction probability greater than or equal to the predefined threshold τ.

(S_3) $\hat{y}_k \neq \tilde{y}_k$ with $P(\hat{y}_k) < \tau$ **and** $P(\tilde{y}_k) < \tau$ i.e., both the branches have different predictions and their prediction probabilities are also lower than the predefined threshold τ.

For (S_1), both the branches are in agreement and we assume that the label is noise-free. For (S_2), although the predictions are different, either one of the two branches is confident about its prediction. This difference may arise due to different loss functions in each branch. However, since one of the branches is certain about its decision, we assume that the sample's label is correct. For (S_3), both the branches are in disagreement, and also none of them is confident about its prediction. If we choose $\tau = 0.60$, then the prediction probabilities will be in the range of $(0.50, 0.60]$, which is very low. This shows a low confidence of both the branches along with the disagreement. We assume that such samples have noisy labels. Samples that meet (S_3) and hence, have noisy labels, can be handled with the below two approaches.

(A_1) **Label Flipping:** If we have a binary class dataset, each sample will belong to either class 0 or class 1. Therefore, when we suspect a sample to be having an incorrect label, we flip its label to that of the other class, i.e.,

$$y_k = 1 - y_k \quad \text{if} \;\; \hat{y}_k \neq \tilde{y}_k \text{ with } P(\hat{y}_k) < \tau \text{ and } P(\tilde{y}_k) < \tau. \tag{14}$$

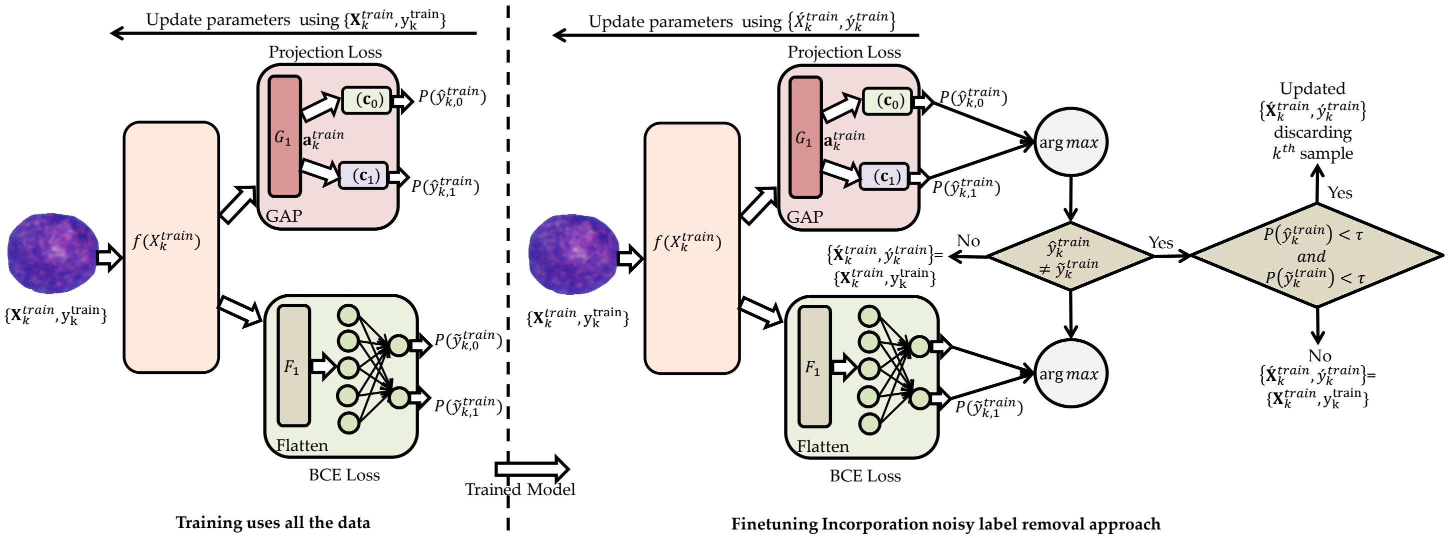

Update parameters using $\{\mathbf{X}_k^{train}, \mathrm{y}_k^{\mathrm{train}}\}$
Projection Loss
$(\mathbf{c}_0)$
$P(\hat{y}_{k,0}^{train})$
G_1
$\mathbf{a}_k^{train}$
$(\mathbf{c}_1)$
$P(\hat{y}_{k,1}^{train})$
GAP
$f(X_k^{train})$
$\{\mathbf{X}_k^{train}, \mathrm{y}_k^{\mathrm{train}}\}$
F_1
$P(\tilde{y}_{k,0}^{train})$
$P(\tilde{y}_{k,1}^{train})$
Flatten
BCE Loss
Trained Model
Training uses all the data
Update parameters using $\{\acute{X}_k^{train}, \acute{y}_k^{train}\}$
Projection Loss
$(\mathbf{c}_0)$
$P(\hat{y}_{k,0}^{train})$
G_1
$\mathbf{a}_k^{train}$
$(\mathbf{c}_1)$
$P(\hat{y}_{k,1}^{train})$
GAP
arg max
$f(X_k^{train})$
$\{\mathbf{X}_k^{train}, \mathrm{y}_k^{\mathrm{train}}\}$
$\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}=$ $\{\mathbf{X}_k^{train}, \mathrm{y}_k^{\mathrm{train}}\}$
No
$\hat{y}_k^{train} \neq \tilde{y}_k^{train}$
Yes
$P(\hat{y}_k^{train}) < \tau$ and $P(\tilde{y}_k^{train}) < \tau$
Yes
Updated $\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}$ discarding k^{th} sample
No
$\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}=$ $\{\mathbf{X}_k^{train}, \mathrm{y}_k^{\mathrm{train}}\}$
F_1
arg max
$P(\tilde{y}_{k,1}^{train})$
Flatten
BCE Loss
Finetuning Incorporation noisy label removal approach

Figure 5: The training process of the *PRLCE-Net* with the incorporation of the *sample discarding* as a label noise handling approach. The complete training occurs in two phases: first, the network is trained with all the training data and later, it is fine-tuned with *sample discarding* approach.

Hence, true label (y_k) is changed if (S_3) is satisfied by the predicted labels ($\hat{y}_k$ and $\tilde{y}_k$). For a C class dataset, there are $C-1$ possibilities with which k^{th} true label can be flipped, out of which only one scenario is correct. Hence, we require to take a computationally-expensive iterative approach.

(A_2) **Sample Discarding:** Another approach to handle (S_3) is sample discarding. If any sample satisfies (S_3), we remove that sample from the training set. In this way, we may have a clean dataset with no samples having noisy labels. Also, since no flipping is involved, this approach is independent of the number of classes and has the same computational cost irrespective of the classes.

Sample discarding reduces the number of training samples, which is not the case with *label flipping*. Once we decide to opt for either of the two approaches to tackle (S_3), we fine-tune the model during $\{P, P+1..T-1\}$ epochs with the modified set $\{\mathbf{X}_k, y_k\}$. This fine-tuning with possibly clean data will try to adjust the decision boundary that was obtained earlier until P epochs with the noisy data. With *label flipping*, we have $C-1$ final models for a C class dataset. This has a high computational complexity, especially, if C is very large. However, for $C=2$, we have only one final model. On the other hand, there will always be a single final model with *sample discarding* approach, irrespective of the number of classes.

3.6. Coupling Classifier

Consider *PRLCE-Net* (Fig. 4). As the network contains the two branches, there will be two possible scenarios for any test sample $\mathbf{X}_k^{test}$:

1. Both branches have same predictions i.e $\hat{y}_k^{test} = \tilde{y}_k^{test}$.
2. Both branches yield different outcomes, i.e., $\hat{y}_k^{test} \neq \tilde{y}_k^{test}$.

For case (1), we consider the prediction of either branch to be the final prediction ($\acute{y}_k^{test}$) or $\acute{y}_k^{test} = \hat{y}_k^{test} = \tilde{y}_k^{test}$. However, in case of a conflict, the prediction of any branch could be correct. Hence, choosing the label of any one branch will make us biased towards that particular branch. It also gives uncertainty in choosing the right prediction. Hence, it is ideal to output a single label instead of two different predictions. We propose a solution to this problem in the form of a *coupling classifier.* This couples the two branches and helps to obtain a unique prediction from the *PRLCE-Net.*

Training of the Coupling Classifier. Once the training of the *PRLCE-Net* is completed, we find all the training samples on which both the branches are yielding correct and same predictions.

$$\acute{k} = \{k | \hat{y}_k^{train} = \hat{y}_k^{train} = y_k^{train}, \text{ and } 0 \le k \le N-1\} \tag{15}$$

We use $\{\mathbf{X}_{\acute{k}}^{train}, y_{\acute{k}}^{train}\}$ to train the coupling classifier. However, instead of using flattening or GAP, $\{\mathbf{X}_{\acute{k}}^{train}\}$ is passed through a spectral averaging layer. Hence, $\{S\left(f\left(\mathbf{X}_{\acute{k}}^{train}\right)\right), y_{\acute{k}}^{train}\}$ are used for training the *coupling classifier.* The use of a different type of pooling provides a different set of features to the *coupling classifier* as compared to the ones obtained from the other two branches. This helps the classifier in making a better decision.

Testing of the Coupling Classifier. During the testing, test samples having distinct predictions by both the classifiers are predicted by the *coupling classifier.* Let $\mathbf{X}_k^{test}$ be the sample for which we have a conflicting decision. The $f(\mathbf{X}_k^{test})$ is then passed to the spectral averaging layer and the resultant output is given to the *coupling classifier* to predict its label. Again, there are two possibilities to use the labels predicted by the *coupling classifier.*

(D_1) **Stand-Alone Decision:** In this case, we consider the label of the *coupling classifier* to be the final prediction, i.e., $\acute{y}_k^{test} = C\left(S(f(\mathbf{X}_k^{test})\right)$. Through this, an independent decision is obtained because we ignore the predictions of both the branches.

(D_2) **Ensemble Decision:** Another possibility is to consider the decisions of both the branches along with the *coupling classifier.* For example, we can obtain the final decision as:

$$\acute{y}_k^{test} = g\left(\hat{y}_k^{test}, \tilde{y}_k^{test}, C\left(S(f(\mathbf{X}_k^{test}))\right)\right), \tag{16}$$

where $g(\cdot)$ is some ensembling function. If we consider $g(\cdot)$ to be the majority voting, the final label is same as that predicted by two or more

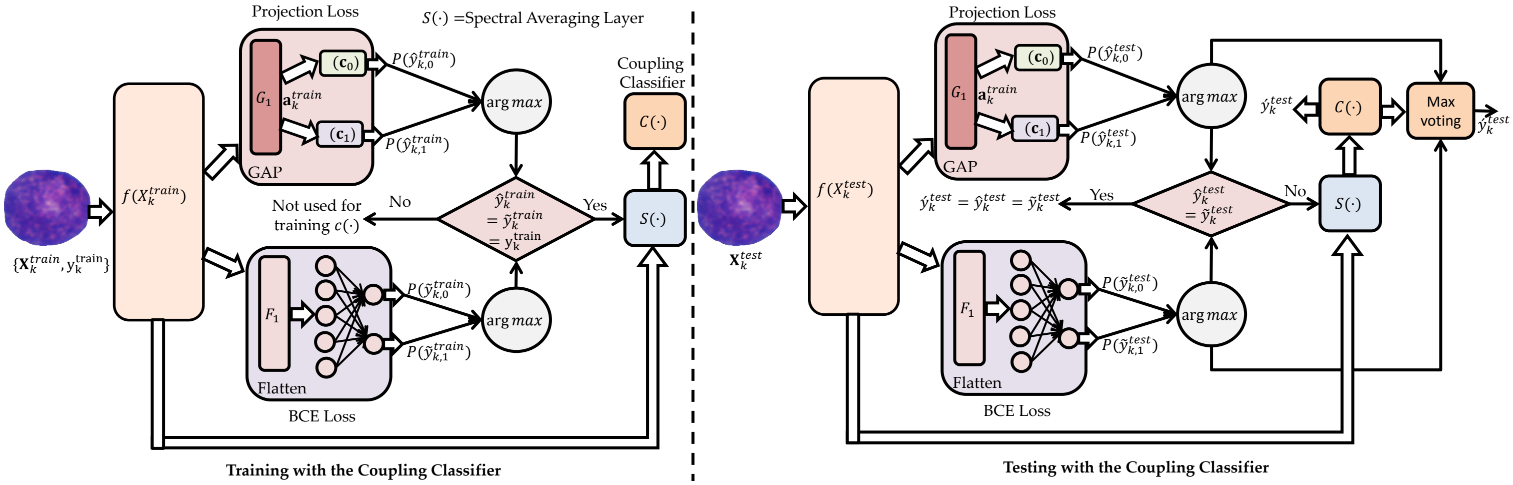

Projection Loss
$S(\cdot)$ =Spectral Averaging Layer
$(\mathbf{c}_0)$
$P(\hat{y}_{k,0}^{train})$
G_1
$\mathbf{a}_k^{train}$
$(\mathbf{c}_1)$
$P(\hat{y}_{k,1}^{train})$
arg max
Coupling Classifier
$C(\cdot)$
GAP
$f(X_k^{train})$
Not used for training $c(\cdot)$
No
$\hat{y}_k^{train} = \tilde{y}_k^{train} = \mathrm{y}_k^{\mathrm{train}}$
Yes
$S(\cdot)$
$\{\mathbf{X}_k^{train}, \mathrm{y}_k^{\mathrm{train}}\}$
F_1
$P(\tilde{y}_{k,0}^{train})$
$P(\tilde{y}_{k,1}^{train})$
arg max
Flatten
BCE Loss
Training with the Coupling Classifier
Projection Loss
$(\mathbf{c}_0)$
$P(\hat{y}_{k,0}^{test})$
G_1
$\mathbf{a}_k^{train}$
$(\mathbf{c}_1)$
$P(\hat{y}_{k,1}^{test})$
arg max
$\acute{y}_k^{test}$
$C(\cdot)$
Max voting
$\acute{y}_k^{test}$
GAP
$f(X_k^{test})$
$\acute{y}_k^{test} = \hat{y}_k^{test} = \tilde{y}_k^{test}$
Yes
$\hat{y}_k^{test} = \tilde{y}_k^{test}$
No
$S(\cdot)$
$\mathbf{X}_k^{test}$
F_1
$P(\tilde{y}_{k,0}^{test})$
$P(\tilde{y}_{k,1}^{test})$
arg max
Flatten
BCE Loss
Testing with the Coupling Classifier

Figure 6: Training and Testing with Coupling Classifier. Features to the *coupling classifier* are fed through the spectral averaging layer.

classifiers. This criterion will always work for the binary class datasets but fail in the case of multi-class datasets if each classifier's prediction is different.

The training and testing of the *custom classifier* are also summarized in Fig. 6. Also, for the case of binary class datasets, both the testing approaches will lead to the same results.

To summarize, we start with *PRLCE-Net* and include *label flipping* or *sample discarding* to address the label noise problem. Finally, we include the *coupling classifier* to make the final decision without any conflict. This process is also summarized in Fig. 7. Specifically, the training and testing process of *PRLCE-Net* with *sample discarding* and *coupling classifier* is elaborated in the below algorithm.

Algorithm: Training and Testing of *PRLCE-Net+SD+CC* (PRLCE-Net with sample discarding and coupling classifier)

```
Input: PRLCE-Net network, Epoch sets: $\{0, 1, ..P-1\}$ and
       $\{P, P+1..T-1\}$, and coupling classifier: $C(\cdot)$
Output: Final Predictions: $\{\acute{y}_k^{test}\}$
Data: Train set: $\{\mathbf{X}_k^{train}, y_k^{train}\}$, validation set: $\{\mathbf{X}_k^{val}, y_k^{val}\}$, test set:
      $\{\mathbf{X}_k^{test}\}$
Initial Training
while epoch $(e) \in \{0, 1, ..P-1\}$ do
    Update the parameters of PRLCE-Net network (Fig. 4) using (11)
Finetuning incorporating Noisy Labels Handling Approach
while epoch $(e) \in \{P, P+1..T-1\}$ do
    Identify noisy labels' samples using $(S_3)$
    Use $(A_2)$ to update the training set to $\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}$
    Update the parameters of PRLCE-Net network using the
      $\{\acute{\mathbf{X}}_k^{train}, \acute{y}_k^{train}\}$
Train the Coupling Classifier $C(\cdot)$
for $\{\boldsymbol{X}_k^{train}\}$ do
    $\acute{k} = \{k | \acute{y}_k^{train} = \hat{y}_k^{train} = y_k^{train}\}$
    $C = \phi\left(S\left(f\{\mathbf{X}_{\acute{k}}^{train}, y_{\acute{k}}^{train}\}\right)\right)$
Testing on $\{\mathbf{X}_k^{test}\}$
Predict $\hat{y}_k^{test}$ and $\tilde{y}_k^{test}$ using PRLCE-Net network
if $\hat{y}_k^{test} \neq \tilde{y}_k^{test}$ then
    $\acute{y}_k^{test} = C\left(S(f(\mathbf{X}_k^{test}))\right)$
else
    $\acute{y}_k^{test} = \hat{y}_k^{test} = \tilde{y}_k^{test}$
```

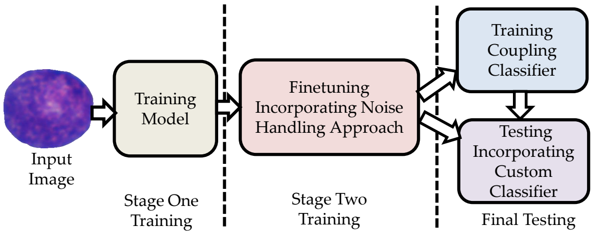

Input Image
Training Model
Stage One Training
Finetuning Incorporating Noise Handling Approach
Stage Two Training
Training Coupling Classifier
Testing Incorporating Custom Classifier
Final Testing

Figure 7: Complete training and testing strategy incorporating label noise handling and coupling classifier.

4. Results and Discussion

4.1. Multiple Myeloma (MM) Dataset

In this section, we will validate all the proposed methodologies. We will also compare the proposed architectures with the existing networks. First, we will discuss the results on MM dataset.

4.1.1. Training and Testing details

Stochastic gradient descent with a momentum of 0.9 is used as an optimizer. We have also used a weight decay of 0.01 and a batch size of 64. The training is carried out for a total of 150 epochs, starting from a learning rate of 0.001. The learning rate is reduced to one-tenth of its present value after 80^{th}, 120^{th}, and 140^{th} epoch. The parameters β and γ in (8) are set to 2 and 1, respectively, and τ is set to 0.6 in (S_2) and (S_3). Also, the values of β_1 and β_2 in (11) are 1. PyTorch deep learning library is used for the implementation and GeForce RTX 2080 Ti is used to accelerate the training and testing processes. This strategy is used for the training of *BaseCE-Net*, *BasePRL-Net*, and *PRLCE-Net.* For training *PRLCE-Net+SD* or *PRLCE-Net+LF*, we have fine-tuned *PRLCE-Net* for another 35 epochs with an initial learning rate of 0.00001 using the Adam optimizer after incorporating *sample discarding* or *label flipping.* The current value of the learning rate is multiplied by 0.1 after 10^{th} and 30^{th} epoch. Results of the fine-tuning of *PRLCE-Net* with Adam optimizer and without including sample discarding or label flipping is provided in the supplementary. We have also used kernel SVM with radial basis function (RBF) as the coupling classifier's kernel.

We used five-fold cross-validation for the training and testing of the architectures. One fold is used for validation at a given instance and the remaining four folds are used for training. Thus, we obtain five trained models for any particular architecture. We use Model-n to denote the architecture trained using the n^{th} fold as the validation set and remaining folds as the training sets. The best model achieved according to the validation performance is used for final analysis. We used *majority voting* on the predictions obtained from the five models (Model-1 to Model-5) to arrive at the final decision on test samples.

To augment the training data, we used random rotation in $[0, 360)$, and horizontal and vertical flips (Huang et al., 2017; He et al., 2016; Springenberg et al., 2015). We also normalized the dataset through mean and standard deviation before feeding it to the architectures. Oversampling is also used to handle class imbalance, if any. The training curves of both the branches of *PRLCE-Net* are shown in Fig. 8 and indicate the training convergence.

For performance comparison, we used weighted F_1 score as well as the individual F_1 scores for each class. We also used balanced accuracy, which is the average of recall and specificity. Class-wise F_1 score helps highlight the performance on the individual classes, whereas weighted F_1 score and balanced accuracy take the class imbalance into consideration. We also computed the area under the curve (AUC) to highlight different thresholding effects. We utilized the accuracy metric to analyze the subject-level performance.

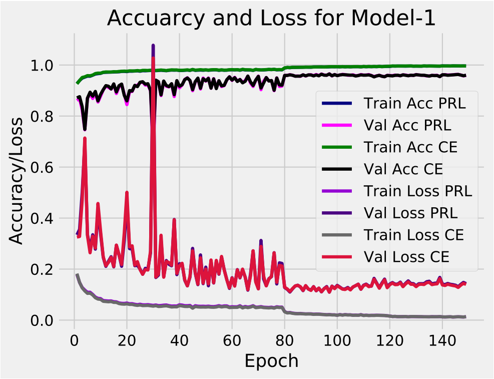


Figure 8: Training curves of PRL and CE branches of *PRLCE-Net*: Model-1. Network trained with fold-1 as the validation set is represented as Model-1.

4.1.2. Ablation Study

We perform an ablation study to highlight the significance of all the proposed techniques. We start with the *BaseCE-Net* and subsequently, analyze the contribution of each additional component. As a notation, *PRLCE-Net+SD (CE)* denotes the *CE* branch of *PRLCE-Net* trained with incorporating *Sample Discarding*. Similarly, *PRLCE-Net+SD (PRL)* represents the *PRL* branch of the same network. Also, *PRLCE-Net+SD+CC* denotes *PRLCE-Net* augmented by *Sample Discarding* and *Coupling Classifier*. Other notations can be followed on similar lines.

As compared to *BaseCE-Net*, *BasePRL-Net* performs better on three models (Model-1, Model-4, and Model-5), while the latter leads on the remaining two models. Overall, with ensembling (majority voting), *BasePRL-Net* leads *BaseCE-Net*. This trend is seen on all the four metrics. Specifically, *BasePRL-Net* performs better than *BaseCE-Net* with a margin of 0.19% on the healthy class F_1 score (NF_1 score), 0.35% on the cancer class F_1 score (NF_1 score), 0.29% on the weighted F_1 score (NF_1 score), and 0.27% on the balanced accuracy, which shows the contribution of the projection loss.

PRLCE-Net consists of CE loss and projection loss in different branches. This combination helps boost the performance of each branch as compared to the individual networks (*BaseCE-Net* and *BasePRL-Net*). For example, with ensembling, as compared to *BaseCE-Net*, *PRLCE-Net (CE)* gains by 0.55%, 0.76%, 0.66%, 0.66% on NF_1 score, CF_1 score, WF_1 score and balanced accuracy, respectively. Similar trends are seen on individual models (Model-1 to Model-5). Similarly, *PRLCE-Net (PRL)* performs better than *BasePRL-Net* on each model (Model-1 to Model-5), as well as with ensembling. Overall, the former is performing better than the latter by a margin of 0.31% (NF_1 score), 0.35% (CF_1 score), 0.33% (WF_1 score), and 0.32% (balanced accuracy).

Table 4: Results of Healthy Class F_1 Score and Cancer Class F_1 Score with all the proposed methods. These results are obtained on 40440 test images. Best results are highlighted in bold. Same results are depicted in italics. Network trained with fold-n (n=1,2..5) as the validation set is represented as Model-n.

Healthy Class F_1 Score						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9212	0.9391	0.9275	0.9321	0.9309	0.9374
BasePRL-Net	0.9285	0.9310	0.9253	0.9378	0.9316	0.9393
PRLCE-Net (CE)	0.9312	0.9435	0.9304	0.9388	0.9368	0.9429
PRLCE-Net (PRL)	0.9314	0.9424	0.9307	0.9389	0.9371	0.9424
PRLCE-Net+CC	0.9317	0.9439	0.9308	0.9394	0.9378	0.9435
PRLCE-Net+LF (CE)	0.9356	0.9360	0.9276	0.9458	0.9408	0.9440
PRLCE-Net+LF (PRL)	0.9309	0.9325	0.9253	0.9443	0.9372	0.9404
PRLCE-Net+LF (CC)	0.9357	0.9361	0.9278	0.9463	0.9410	0.9443
PRLCE-Net+SD (CE)	**0.9404**	0.9435	0.9359	0.9437	0.9414	0.9481
PRLCE-Net+SD (PRL)	0.9367	0.9409	0.9356	0.9423	0.9384	0.9457
PRLCE-Net+SD+CC	0.9403	**0.9436**	**0.9360**	**0.9442**	**0.9416**	**0.9482**
Cancer Class F_1 Score						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9006	0.9274	0.9094	0.9156	0.9143	0.9233
BasePRL-Net	0.9121	0.9187	0.9073	0.9251	0.9167	0.9268
PRLCE-Net (CE)	0.9160	0.9320	0.9140	0.9249	0.9238	0.9309
PRLCE-Net (PRL)	0.9161	0.9305	0.9145	0.9251	0.9240	0.9303
PRLCE-Net+CC	0.9167	0.9327	0.9148	0.9259	0.9252	0.9318
PRLCE-Net+LF (CE)	0.9223	0.9224	0.9104	0.9351	0.9291	0.9327
PRLCE-Net+LF (PRL)	0.9150	0.9171	0.9068	0.9329	0.9238	0.9275
PRLCE-Net+LF (CC)	0.9224	0.9225	0.9107	0.9357	0.9295	0.9331
PRLCE-Net+SD (CE)	**0.9291**	*0.9321*	0.9223	0.9320	0.9301	0.9381
PRLCE-Net+SD (PRL)	0.9235	0.9283	0.9217	0.9300	0.9258	0.9346
PRLCE-Net+SD+CC	0.9289	*0.9321*	**0.9225**	**0.9328**	**0.9304**	**0.9383**

4.1.3. Effect of Label Flipping

The effect of label flipping ((A_1)) in handling label noise is shown in Table 4 and Table 5. Comparing *PRLCE-Net+LF (CE)* with *PRLCE-Net (CE)*, an improvement is seen on three models, and finally, after majority voting, the gain is 0.11%, 0.18%, 0.14%, 0.15% for NF_1 score, CF_1 score, WF_1 score, and balanced accuracy, respectively. However, an improvement is observed with *PRLCE-Net+LF (PRL)* as compared to *PRLCE-Net (CE)* only for one model (Model-4). On the remaining models or with majority voting, there is no gain on either of the metrics. Overall, there is a reduced performance with the inclusion of label flipping.

4.1.4. Effect of Sample Discarding

Next, we analyze the contribution of sample discarding as a noise label handling approach. We compare the performance of *PRLCE-Net+SD* with other

Table 5: Results in terms of Weighted F_1 Score and Balanced Accuracy with all the proposed methods. These results are obtained on 40440 test images. Best results are highlighted in bold. Network trained with fold-n (n=1,2..5) as the validation set is represented as Model-n.

Weighted F_1 Score						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9113	0.9335	0.9188	0.9242	0.9230	0.9306
BasePRL-Net	0.9206	0.9251	0.9167	0.9317	0.9244	0.9333
PRLCE-Net (CE)	0.9239	0.9380	0.9226	0.9321	0.9306	0.9372
PRLCE-Net (PRL)	0.9241	0.9367	0.9229	0.9323	0.9308	0.9366
PRLCE-Net+CC	0.9245	0.9385	0.9231	0.9329	0.9318	0.9379
PRLCE-Net+LF (CE)	0.9293	0.9295	0.9193	0.9407	0.9352	0.9386
PRLCE-Net+LF (PRL)	0.9233	0.9251	0.9165	0.9389	0.9308	0.9342
PRLCE-Net+LF (CC)	0.9293	0.9296	0.9196	0.9413	0.9355	0.9390
PRLCE-Net+SD (CE)	**0.9350**	0.9380	0.9294	0.9381	0.9360	0.9433
PRLCE-Net+SD (PRL)	0.9304	0.9348	0.9289	0.9364	0.9323	0.9404
PRLCE-Net+SD+CC	0.9348	**0.9381**	**0.9295**	**0.9388**	**0.9362**	**0.9435**
Balanced Accuracy						
Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting
BaseCE-Net	0.9088	0.9317	0.9164	0.9218	0.9206	0.9285
BasePRL-Net	0.9184	0.9235	0.9144	0.9298	0.9223	0.9314
PRLCE-Net (CE)	0.9218	0.9360	0.9203	0.9299	0.9286	0.9351
PRLCE-Net (PRL)	0.9220	0.9347	0.9206	0.9301	0.9289	0.9346
PRLCE-Net+CC	0.9224	0.9366	0.9209	0.9307	0.9298	0.9359
PRLCE-Net+LF (CE)	0.9273	0.9274	0.9170	0.9388	0.9333	0.9366
PRLCE-Net+LF (PRL)	0.9211	0.9229	0.9141	0.9369	0.9287	0.9321
PRLCE-Net+LF (CC)	0.9274	0.9275	0.9173	0.9394	0.9337	0.9371
PRLCE-Net+SD (CE)	**0.9332**	0.9361	0.9273	0.9361	0.9342	0.9415
PRLCE-Net+SD (PRL)	0.9284	0.9328	0.9269	0.9343	0.9304	0.9384
PRLCE-Net+SD+CC	0.9331	**0.9362**	**0.9275**	**0.9368**	**0.9345**	**0.9417**

methods in Table 4 and Table 5. On comparing *PRLCE-Net+SD (PRL)* with *PRLCE-Net (PRL)* or *PRLCE-Net+SD (CE)* with *PRLCE-Net (CE)*, we see an enhanced performance on each model as well as with ensembling. For NF_1 score, *PRLCE-Net+SD (CE)* leads *PRLCE-Net (CE)* by 0.52%, whereas with *BaseCE-Net* the margin is 1.07%. Similarly, margins, after including sampling discarding, are 0.72% for CF_1 score, 0.61% for WF_1 score, and 0.64% on balanced accuracy in the CE branch. For the PRL branch, the introduction of sample discarding led to a gain of 0.33%, 0.43%, 0.38%, 0.38% in terms of NF_1 score, CF_1 score, WF_1 score, and balanced accuracy, respectively. These values are compared to the results of *PRLCE-Net (PRL)*. The gain is higher in comparison to the *BasePRL-Net*.

4.1.5. Role of Coupling Classifier

It is observed from Tables 4-5 that the networks' predictions involving the CE and PRL branches are not the same for both the branches, and that there is

Table 6: The number of predictions aided by the coupling classifier. These results are obtained on 40440 test images. Network trained with fold-n (n=1,2,...,5) as the validation set is represented as Model-n.

Architectures/Models	Model-1	Model-2	Model-3	Model-4	Model-5
PRLCE-Net+CC	153	145	120	120	154
PRLCE-Net+SD+CC	334	193	135	174	259

a conflict in the decision of the two classifiers. Table 6 shows the exact number of samples for *PRLCE-Net* and *PRLCE-Net+SD* on which there is a disagreement between the two branches. Without a coupling classifier, there will be an uncertainty in choosing the prediction of either branch. Apart from providing certainty in the final predictions, the coupling classifier also yields somewhat improved performance. Although the improvement is not that significant, the coupling classifier serves its primary purpose of removing the discrepancy.

4.1.6. Summary of the Ablation Study

These results indicate that each proposed component aids in performance enhancement. *PRLCE-Net+SD+CC* is obtained after modifying *BaseCE-Net* by including projection loss, label noise handling, and the coupling classifier. On comparing *BaseCE-Net* with *PRLCE-Net+SD+CC*, we observe an improvement of 1.5% on CF_1 score. This improvement is 1.08% in terms of NF_1 score. Similarly, we see an increment of 1.29%, and 1.32% on WF_1 score, and balanced accuracy, respectively. We also observe the significance of *PRLCE-Net* because it provides the scope for introducing label noise handling strategy like *sample discarding* and *label flipping*, and ensembling with the *coupling classifier.*

4.1.7. Receiver Operating Characteristics (ROC) and Area Under Curve (AUC)

Next, we analyze ROC and AUC for all the proposed architectures. Results of different models (Model-1 to Model-7) are shown in Fig. 9. There is an incremental trend in the AUC from *BaseCE-Net* to *BasePRL-Net.* On all models, except for Model-5, *PRLCE-Net+SD (PRL)* has the highest value of AUC. Same values are obtained for *PRLCE-Net+SD (PRL)* and *PRLCE-Net(PRL)* in Model-4. Also, among the PRL and CE branches, the former has the dominant AUC. Again, on majority of the models, sample discarding yields better AUC compared to label flipping. Also, on some of the models (Model-2 and Model-4), addition of label flipping does not provide improved performance. All architectures are observed to have high AUC that is close to one in some cases.

4.1.8. Subject Level Performance Analysis

Apart from the overall performance on all the test images, another essential aspect in the computer-aided diagnosis is subject-level performance. This is because it will be tested on new test subjects in a realistic environment once the model is deployed. As there might be subject-level variability, the performance

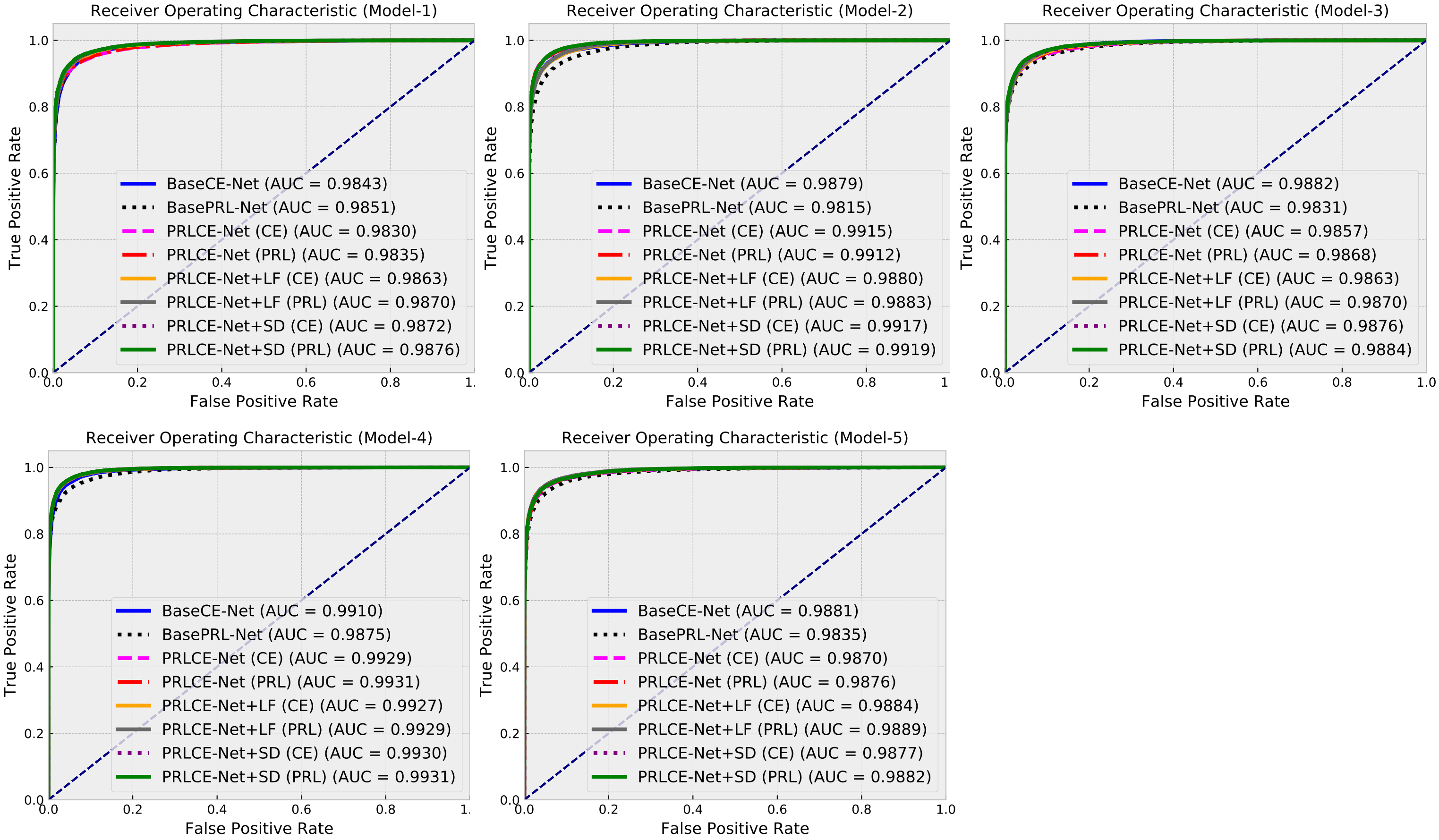
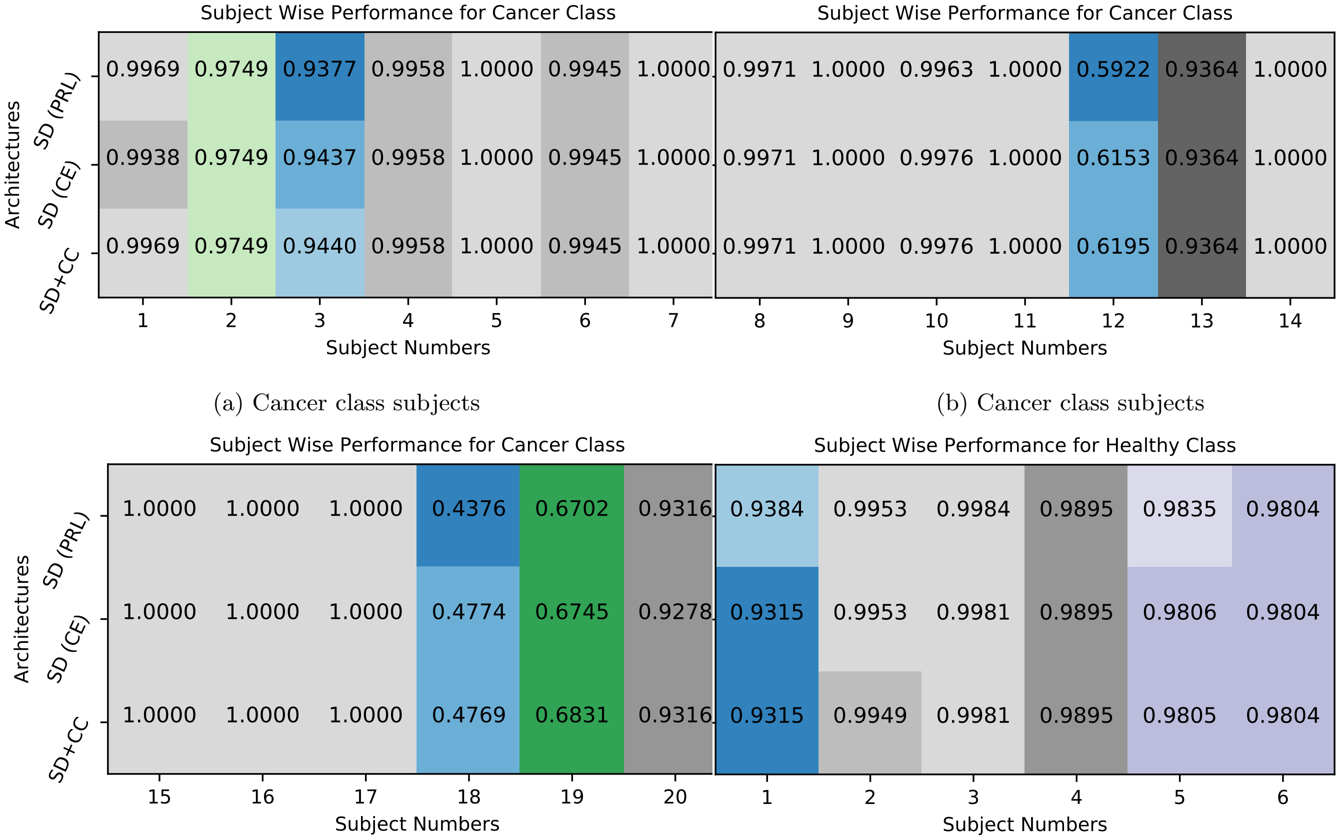


Figure 9: Receiver Operating Characteristics and Area Under Curve (AUC) with different architectures. These results are obtained on 40440 test images. The network trained with fold-n (n=1,2,...,5) as the validation set is represented as Model-n.

may vary at the subject-level. The variations could be due to some noise in the image capturing process or some variations in the slide preparation pipeline, or other related issues. To highlight this issue, we have carried out a subject-level analysis, as shown in Fig. 10. The analysis is performed on twenty subjects of the cancer class and five subjects belonging to the healthy class. With *PRLCE-Net+SD+CC*, the network performance is very high on the seventeen subjects of the cancer class, with 100% accuracy on some subjects (subject no. 5,7,9,11,14-17). For the rest of them, the accuracy is close to 1.00, while for some, it varies from $93.13\% - 97.49\%$. On three subjects, the performance is poor. The accuracy on subject numbers 12 and 19 is 61.95% and 68.31%, respectively. On the last remaining subject (no. 18), the accuracy is only 47.69%. Hence, the classifier performs well on most subjects, but its performance is non-optimal



(a) Cancer class subjects

(b) Cancer class subjects

(c) Cancer class subjects

(d) Healthy class subjects

Figure 10: Subject-level performance in terms of accuracy with *PRLCE-Net* and its variants: cancer class subjects (a-c), and healthy class subjects (d)

on some of them. This difference in the performance highlights the impact of subject-level variations. This also implies that overall performance is impacted only due to some subjects. Overall, we are able to design a classifier that performs well on most of the subjects (17 out of 20).

On the healthy subjects, the performance is consistently good with minimum performance being 93.15% and a maximum being 99.81%. There is not a significant decline in the performance on any subject in the healthy class as was observed in the cancer class. The standard deviation of the accuracy between the subjects is 0.0244 that again indicates stable inter-subject performance.

Again, a difference is observed in the performance of *PRLCE-Net+SD (CE)* and *PRLCE-Net+SD (PRL)*. The coupling classifier helps in deciding with certainty. It either deflects the outcomes towards one branch or leads to better performance compared to both the branches. Hence, the coupling classifier is also helpful in improving the subject-level performance.

4.1.9. Comparison With Existing Architectures

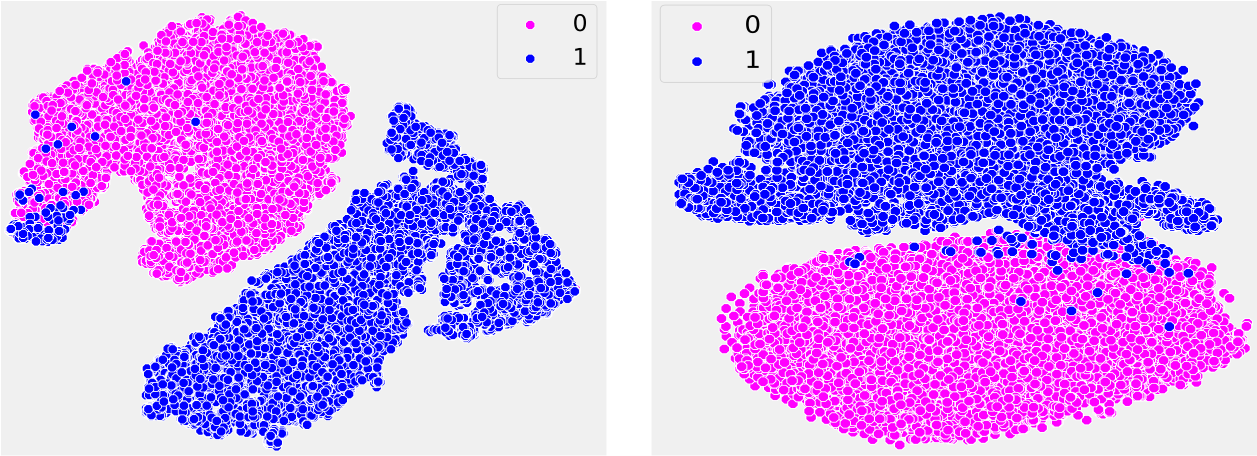
We have also compared the performance of the proposed methodology with some existing architectures, and some of them have achieved state-of-the-art results on the classification task. These results are depicted in Table 7. For training, these architectures are initialized with pre-trained weights on the ImageNet dataset. The input image size is resized to 224 × 224 (299 × 299 for Inception-v3) as required by these networks. Apart from WF_1 score and balanced accuracy, we also analyzed the number of parameters and the test compute time of each network. As seen from Table 7, SqueezeNet (Iandola et al., 2016) has the least WF_1 score and balanced accuracy. The SDCT-Net (Gehlot et al., 2020b) is a very compact architecture with least number of parameters and testing time, and yet it is performing better than SqueezeNet. The SDCT-AuxNet$^{\theta}$ (Gehlot et al., 2020b) is performing better than SDCT-Net, highlighting the impact of auxiliary classifier. Also, both of these architectures are trained from scratch with original image size (350 × 350). ShuffleNet-V2 (Ma et al., 2018) performs better than SqueezeNet by 4.38% and 4.14% for WF$_1$ score and balanced accuracy, respectively. Also, the number of parameters (and test time) has increased significantly. GoogleNet (Szegedy et al., 2015) performs with a marginal increase of 4.68% and 4.48% over SqueezeNet in terms of WF_1 score and balanced accuracy. DenseNet121 (Huang et al., 2017) is performing almost similar to GoogleNet, but has a higher number of parameters and greater test time. ResNet34 (He et al., 2016) and ResNeXT (Xie et al., 2017) have better performance compared to ResNet18 (He et al., 2016), but the former's number of parameters and test time are also relatively high. All of these three networks have residual connections and have better performance than DenseNet121 and a higher number of the parameters. MobileNet-V2 (Sandler et al., 2018) has the least number of parameters (and test time) than all these architectures (except for SqueezeNet and ShuffleNet), but has better performance. Results with distance based cross entropy loss (DCE loss) are calculated by replacing the loss function in *BasePRL-Net* and training the network from scratch. The performance with DCE loss (Yang et al., 2018) is better than all the existing architectures discussed to this point. Inception-V3 (Szegedy et al., 2016) has the highest number of parameters and test time, but it also has better performance than these architectures. From Table 5, we observe that *BaseCE-Net* has better performance than all these architectures, which are also trained with BCE loss.

Further, projection loss performs better than the DCE loss. The DCE loss (Yang et al., 2018) minimizes the euclidean distance between the class centers and respective activations, with no constraints on the centers. In contrast, the projection loss is minimizing the projection of the features on respective class vectors. It is also constraining class vectors to be orthogonal to each other. These results highlight the importance of the network's depth used in this proposed work and the contribution of projection loss over DCE loss. Finally, *PRLCE-Net+SD+CC* has a leading performance than the rest of the network in terms of both the metrics.

Table 7: Comparison of proposed method with some existing architectures in terms of weighted F_1 score, balanced accuracy, number of parameters, and test time. Results are computed on all 40441 test samples including time taken in decision making. Best results are highlighted in bold. $*$: parameters without coupling classifier. †: parameters without auxiliary classifier.

Weighted F_1 Score								
Models/ Architectures	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting	Parameters	Time (in sec)
SqueezeNet	0.8161	0.8414	0.8356	0.8449	0.8447	0.8514	736450	2469.5092
SDCT-Net	0.8493	0.8849	0.8626	0.8354	0.8605	0.8700	95937	2064.1083
SDCT-AuxNet$^{\theta}$	0.8547	0.8819	0.8661	0.8447	0.8677	0.8811	95937†	2999.2237
ShuffleNet-V2	0.8539	0.8772	0.8928	0.8823	0.9072	0.8952	1255654	9259.4499
GoogleNet	0.8730	0.8943	0.9012	0.8702	0.9001	0.8982	5601954	12419.9610
DenseNet121	0.9041	0.8980	0.8890	0.8631	0.8942	0.8984	6955906	19884.8681
ResNet18	0.8892	0.8924	0.9018	0.8442	0.8994	0.9052	11177538	9360.4598
ResNeXt	0.9114	0.8931	0.9119	0.8581	0.8940	0.9058	22984002	18348.7165
ResNet34	0.8866	0.9211	0.9054	0.8372	0.9057	0.9094	21285698	15202.9181
MobileNet-V2	0.9098	0.8664	0.9131	0.9057	0.9093	0.9120	2226434	9505.9263
DCE Loss	0.9005	0.8813	0.9045	0.9090	0.9093	0.9142	2531979	3105.1951
Inception-V3	0.9210	0.9331	0.9043	0.8877	0.8944	0.9171	24348900	23590.9895
PRLCE-Net+SD+CC	0.9348	0.9381	0.9295	0.9388	0.9362	**0.9435**	2569871*	3694.9547
Balanced Accuracy								
Models/ Architectures	Model-1	Model-2	Model-3	Model-4	Model-5	Majority Voting	Parameters	Time (in sec)
Squeezenet	0.8142	0.8414	0.8381	0.8462	0.8435	0.8512	736450	2469.5092
SDCT-Net	0.8472	0.8832	0.8606	0.8332	0.8580	0.8677	95937	2064.1083
SDCT-AuxNet$^{\theta}$	0.8552	0.8799	0.8672	0.8429	0.8653	0.8796	95937†	2999.2237
Shufflenet-V2	0.8516	0.8746	0.8903	0.8796	0.9048	0.8926	1255654	9259.4499
GoogleNet	0.8709	0.8920	0.8993	0.8679	0.8979	0.8959	5601954	12419.9610
DenseNet121	0.9017	0.8958	0.8865	0.8604	0.8915	0.8957	6955906	19884.8681
ResNet18	0.8869	0.8898	0.8997	0.8432	0.8967	0.9028	11177538	9360.4598
ResNeXt	0.9093	0.8908	0.9094	0.8555	0.8912	0.9031	22984002	18348.7165
ResNet34	0.8844	0.9188	0.9028	0.8352	0.9031	0.9068	21285698	15202.9181
MobileNet-V2	0.8637	0.9108	0.9033	0.9065	0.9094	0.9071	2226434	9505.9263
DCE Loss	0.8990	0.8793	0.9030	0.9072	0.9069	0.9124	2531979	3105.1951
Inception-V3	0.9190	0.9320	0.9019	0.8868	0.8917	0.9149	24348900	23590.9895
PRLCE-Net+SD+CC	0.9330	0.9361	0.9275	0.9368	0.9344	**0.9417**	2569871*	3694.9547

In conclusion, we observe that a larger number of parameters does not lead to significantly higher performance. This may be due to the relatively smaller size of the dataset, although the dataset is very large looking from the perspective of medical domain. Also, fewer parameters are not sufficient for the satisfactory performance either. Hence, it is necessary to design an optimal depth network that has, perhaps, been achieved with our custom network. Also, architectures trained from scratch have better performance than the ones initialized with pre-trained weights.



(a) PRL branch

(b) CE branch

Figure 11: t-SNE plots with Model-5 on the training data for (a) PRL branch, and (b) CE branch.

We also visualized the t-SNE plots for *PRLCE-Net*. Since this network has two branches containing CE loss and PR loss, the scatter plots are depicted for both the losses. The t-SNE is used to reduce the feature dimension to 2 from 18432 and 512 for the CE branch and PRL branch, respectively. Resulting plots in Fig. 11 show the class separation with both of these branches on the training data with Model-5. Perfect classification is not achieved with either loss, as some samples lie on the opposite side of the boundary. In both of the cases, some cancer class samples are on the opposite side. However, with PR loss, there is a clear separation (with some error) of the two classes, which is not the case with the CE loss.

4.2. Camelyon7 and TBX11K datasets

We use these two datasets for the validation of the label noise handling with the proposed methodology.

4.2.1. ***Experiments Set-Up***

The label noise is introduced in both the datasets and experiments are carried out with the proposed methodology to analyze its impact in handling the introduced label-noise. We have used pair flipping (Han et al., 2018a) to introduce the label-noise. In pair flipping, for a noise level p, and the number of classes C, the transition matrix $\mathbf{A}_{C\times C}$ is given as:

$$\mathbf{A}_{C\times C} = \begin{bmatrix} 1-p & p & 0 & \dots & 0 \\ 0 & 1-p & p & \dots & 0 \\ & \dots\dots & \dots\dots & \dots & \\ p & 0 & 0 & \dots & 1-p \end{bmatrix}, \tag{17}$$

Table 8: Results on clean Camelyon17 and TBX11K. The best results are highlighted in bold.

Camelyon17			**TBX11K**	
Architecture	WF1	BAC	Architecture	Accuracy
PRLCE-Net (CE)	0.9751	0.9751	PRLCE-Net (CE)	0.9255
PRLCE-Net (PRL)	0.9741	0.9741	PRLCE-Net (PRL)	**0.9364**
PRLCE-Net+CC	**0.9755**	**0.9755**	PRLCE-Net+CC	0.9313

where $\mathbf{A}_{m,k} = P(\hat{y} = k|y = m)$. As an example, for three classes ($C = 3$), and noise level(p) of 0.4:

$$\mathbf{A}_{C\times C} = \begin{bmatrix} 0.6 & 0.4 & 0 \\ 0 & 0.6 & 0.4 \\ 0.4 & 0 & 0.6 \end{bmatrix} \tag{18}$$

Hence, the label is flipped only to the next class with a given probability.

The PRLCE-Net architecture is used for both the datasets except for TBX11K in which a stride of 2 is used in the second last Conv Section also, and the number of class vectors is three. Since there is a class imbalance in TBX11K, oversampling is used during the training to balance the classes. The training details for both datasets are the same as for the MM dataset.

4.2.2. Results on Camelyon17

Results on clean (noise-free) Camelyon17 are summarized in Table-8 in terms of weighted F1 score (WF1) and balanced accuracy (BAC). Next, the noise is introduced through pair flipping, and results are reported with PRLCE-Net in combination with label flipping (LF) and sample discarding (SD) in Table-9. From Table-8 & 9, the best results are obtained with clean data, and performance is decremented with increment in noise level. Further, LF and SD are able to increase the performance with the noisy data. For example, the best performance gain in terms of WF1 at noise levels $0.1, 0.2, 0.3, 0.4$ with inclusion of coupling classifier is $1.3\%, 1.08\%, 1.23\%,$ and 3.7%, respectively. At 0.45 the gain is 6.76%. Similar gains are observed for BAC. Also, both LF and SD provide approximately similar incremental performance, with LF giving the best performance at $p = 0.1$ and 0.4 while SD gives maximum performance on the remaining noise-levels. Another observation is higher gain at higher noise levels. Hence, incorporating noise handling approaches can provide significant gains, particularly at the higher noise.

4.2.3. Results on TBX11K

Results on the clean dataset are provided in Table-8. For coupling-classifier, results are reported with an ensembling approach. The best accuracy of 0.9364 is obtained with PRLCE-Net. These results are obtained through submission on the challenge portal. At the time of writing, these are the best classification results available on the leaderboard (TBX11K Tuberculosis Classification and Detection Challenge, 2020). As compared to the second-best results, our method is providing a gain of 2.85%. This performance is achieved with a very light

Table 9: Results on Camelyon17 at different noise-levels with PRLCE-Net and incorporation of label flipping (LF) and sample discarding (SD). Best results are highlighted in bold.

	WF1					BAC				
Noise-level (p)/ Architectures	0.1	0.2	0.3	0.4	0.45	0.1	0.2	0.3	0.4	0.45
PRLCE-Net (CE)	0.9600	0.9557	0.9421	0.9210	0.8832	0.9601	0.9557	0.9422	0.9214	0.8839
PRLCE-Net (PRL)	0.9596	0.9555	0.9424	0.9171	0.8792	0.9555	0.9555	0.9425	0.9176	0.8801
PRLCE-Net+CC	0.9609	0.9559	0.9424	0.9210	0.8832	0.9610	0.9560	0.9425	0.9214	0.8839
PRLCE-Net+LF (CE)	0.9732	0.9646	0.9505	0.9573	0.9460	0.9733	0.9646	0.9506	0.9574	0.9461
PRLCE-Net+LF (PRL	0.9732	0.9652	0.9524	**0.9580**	0.9507	0.9733	0.9653	0.9525	**0.9580**	0.9507
PRLCE-Net+LF (CC)	**0.9739**	0.9653	0.9523	**0.9580**	0.9507	**0.9739**	0.9653	0.9524	**0.9580**	0.9507
PRLCE-Net+SD (CE)	0.9686	0.9664	0.9534	0.9546	0.9506	0.9686	0.9664	0.9535	0.9547	0.9507
PRLCE-Net+SD (PRL)	0.9689	0.9665	**0.9547**	0.9551	**0.9508**	0.9689	0.9665	**0.9548**	0.9552	**0.9508**
PRLCE-Net+SD+CC	0.9696	**0.9667**	**0.9547**	0.9554	**0.9508**	0.9696	**0.9667**	**0.9548**	0.9554	**0.9508**

Table 10: Results on TBX11K at different noise-levels with PRLCE-Net and incorporation of sample discarding (SD). Best results are highlighted in bold.

Accuracy					
Noise-level (p)/ Architecture	0.1	0.2	0.3	0.4	0.45
PRLCE-Net (CE)	0.9122	0.8898	0.8486	0.8144	0.7020
PRLCE-Net (PRL)	0.9237	0.9079	0.8937	0.8640	0.7753
PRLCE-Net+CC	0.9204	0.9043	0.8937	0.8680	0.8071
PRLCE-Net+LF (CE)	0.9131	0.8934	0.8695	0.8531	0.8038
PRLCE-Net+LF (PRL)	**0.9249**	0.9064	**0.8943**	**0.8886**	0.8625
PRLCE-Net+LF+CC	0.9216	0.9037	0.8898	0.8852	**0.8692**
PRLCE-Net+SD (CE)	0.9146	0.8886	0.8422	0.7789	0.6814
PRLCE-Net+SD (PRL)	0.9237	**0.9085**	0.8904	0.8755	0.8107
PRLCE-Net+SD+CC	0.9194	0.9046	0.8901	0.8767	0.8295

Table 11: Average number of the samples affected in sample discarding (SD) and label flipping (LF) at different noise levels (p) for TBX11K.

Average Number of Samples					
Method/Noise level	0.1	0.2	0.3	0.4	0.45
SD	163	221	424	1350	1772
LF	162	213	354	740	954

architecture (only ten convolutional layers) and no other data augmentation apart from random rotation and oversampling.

Results with noise addition are provided in Table-10. For brevity, we have used the knowledge of pair flipping in experiments with LF. Although this knowledge is not available in the practical scenario, we have used it for concept validation. There are $C-1$ possible scenarios in LF in the absence of such knowledge.

As expected, increased noise label results in decreased performance. Again there is an improvement with LF and SD. At 0.1, there no gain with SD, and with LF, the maximum gain is 0.12%. At 0.2, the maximum gain with SD is 0.06%. At $p = 0.3$ and 0.4, the maximum gain with LF is 0.06% and 2.46%, respectively. Also, there is a maximum gain of 0.87% at $p = 0.4$ with SD. Hence, there is no significant gain up to $p = 0.3$, but at $p = 0.4$, there is a significant

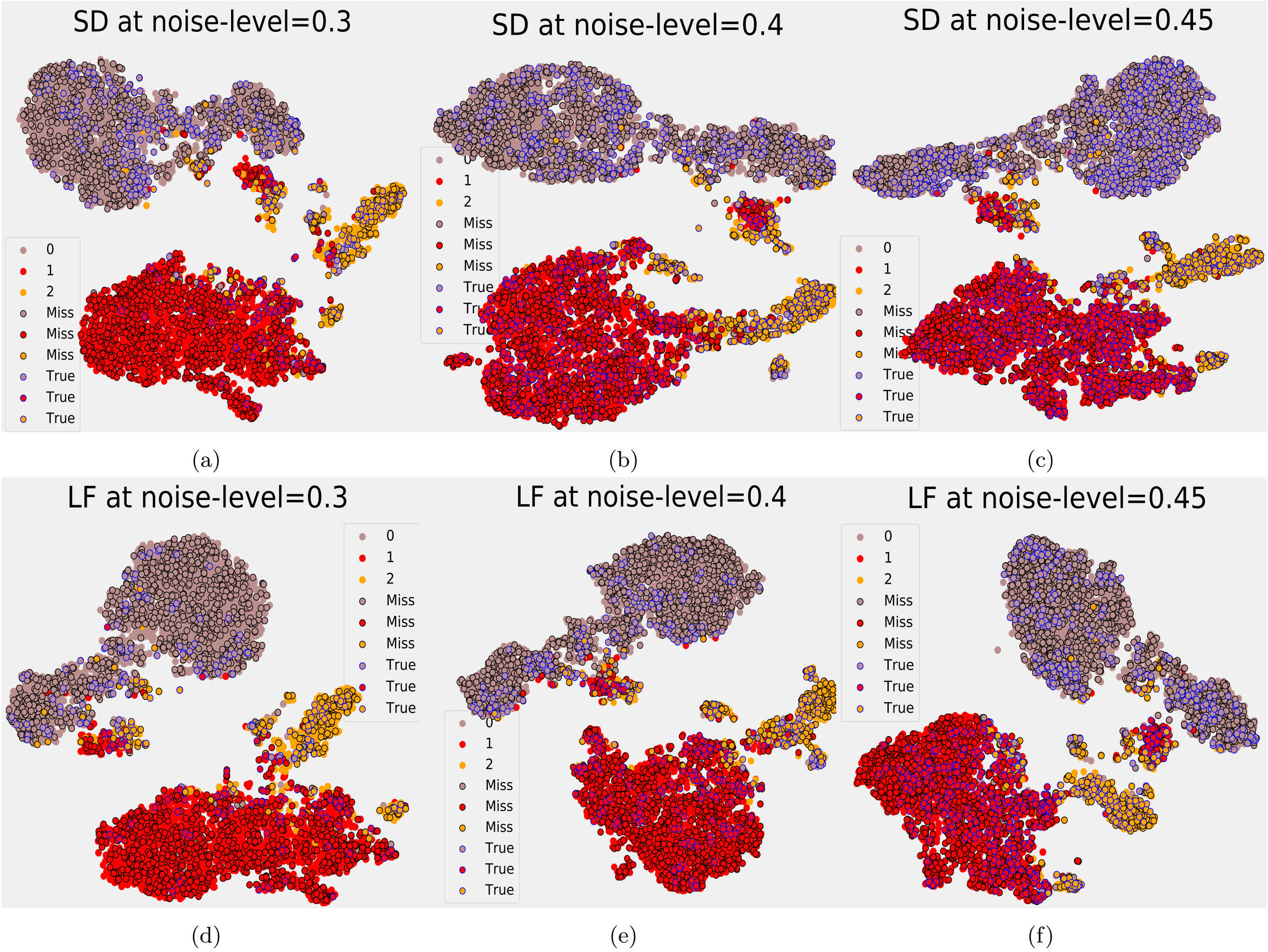
Table 12: True detection rate (TDR) and false detection rate (FDR) for TBX11K at different noise level with sample discarding (SD) and label flippng (LF)

Method	Noise (p)/ Detection Rate	0.1	0.2	0.3	0.4	0.45
LF	TDR (in %)	4.95	6.25	9.09	11.66	16.73
	FDR (in %)	1.53	2.60	4.62	6.51	9.82
SD	TDR (in %)	5.81	7.82	14.22	20.23	28.62
	FDR (in %)	1.53	2.78	7.26	12.51	19.73

improvement. Similarly, at 0.45, the maximum gain is 8.72% and 3.54% with LF and SD, respectively. For Camelyon7 also, the gain is very significant at $p = 0.4$ and 0.45.

Hence, the approach is very effective at the higher noise levels. Another aspect of TBX11K is higher LF performance than SD, especially at the higher noise levels. This is because a more number of the samples affected at the higher noise levels. Hence, in SD, the samples are dropped in a similar proportion making the dataset smaller. It is crucial if the dataset already has fewer samples. It is the case with TBX11K, as it has only 6889 training samples. Whereas in LF, the samples are retained and are some of them are potentially assigned correct labels leading to much more improved performance. To highlight this issue, the average number of the samples affected at different noise levels in SD and LF are reported in Table-11. On average, 1772 samples have been discarded in SD, which is why there is a large margin between SD and LF performance. We have also used the true detection rate (TDR) to represent the percentage of the noisy samples detected correctly. Similarly, the false detection rate (FDR) represents the fraction of the clean samples detected as the noisy samples. We report these two metrics for dataset TBX11K in Table-12 with sample discarding and label flipping. The two main observations from Table-12 are as follows: i) both TDR and FDR are increasing with the increasing noise level. Also, the increment is sharp for each additional noise level. For example, as the noise level increased from 0.4 to 0.45, TDR and FDR in LF increased by 5.07%, and 3.31% respectively. Similarly, the increment in TDR and FDR with SD is 8.39%, and 7.22%, respectively. Hence, the approach is becoming aggressive with increment in the noise level. ii) The TDR and FDR are higher with SD as compared to LF. For example, at $p = 0.45$, SD has an additional 11.89% TDR over LF.

The gap between PRLCE-Net (CE) and PRLCE-Net (PRL) is also covered with the coupling classifier. The t-SNE plots with SD and LF at noise-levels 0.3, 0.4, and 0.45 for dataset TBX11K are shown in Fig-12. The plots show the detected and missed noisy samples. The noisy samples have been detected for all three classes. As observed from Table-12, detection increases with the increased noise level. Also, SD is more aggressive than LF.



(a) (b) (c)

(d) (e) (f)

Figure 12: t-SNE plots for TBX11K with SD (a-c) and LF (d-f) showing the detected noisy samples and missed noisy samples at noise level 0.3 (first column), 0.4 (second column), and 0.45 (third column).

5. Conclusion and Future Work

We have proposed a CNN based unified framework for the diagnosis of multiple myeloma (MM). The problem is challenging due to inter-class visual homogeneity. We have addressed this classification problem through a methodology involving cross-entropy loss, novel projection loss, label noise handling, and coupling classifier. The resultant approach provides a final weighted F_1 score of 94.35%, and a balanced accuracy of 94.17% on a large test set of 40441 images. The method has a good subject-level performance, which is essential. However, the performance lacks on some of the subjects of the cancer class. The low performance on some subjects may be due to subject-level variability in the data. An approach to improve the performance may be to add more subjects to the training dataset. However, the collection of annotated medical datasets is a challenging task. Generative models such as GANs can be used to generate the new subjects' data. However, there could always be cases where testing subject distribution is relatively different from the training subjects. Hence, a robust approach may be to make the architecture immune to such variability. We will consider this aspect in future work.

We have also shown the application of the proposed methodology on the two other datasets (one binary and one three classes dataset). The proposed approach can work on both the datasets in the presence of noise also. Also, on a multi-class dataset (TBX11K), we achieve state-of-the-art classification results with the proposed architecture.

6. Acknowledgements

Authors gratefully acknowledge the research funding support (Grant Number: EMR/2016/006183) from the Department of Science and Technology, Govt. of India. Shiv Gehlot would like to thank University Grant Commission (UGC), Govt. of India for the UGC-Senior Research Fellowship. We also acknowledge the Infosys Center for Artificial Intelligence, IIIT-Delhi for our research work.

References

Amin, M. M., Kermani, S., Talebi, A., & Ghelich Oghli, M. (2016a). Recognition of acute lymphoblastic leukemia cells in microscopic images using k-means clustering and support vector machine classifier. *Journal of Medical Signals and Sensors*, *6*, 183–93.

Amin, M. M., Memari, A., Samadzadehaghdam, N., Kermani, S. et al. (2016b). Computer aided detection and classification of acute lymphoblastic leukemia cell subtypes based on microscopic image analysis. *Microscopy Research and Technique*, *79*, 908–916.

Bayramoglu, N., & Heikkilä, J. (2016). Transfer learning for cell nuclei classification in histopathology images. In *Computer Vision – ECCV 2016 Workshops* (pp. 532–539).

Bayramoglu, N., Kannala, J., & Heikkil, J. (2016). Deep learning for magnification independent breast cancer histopathology image classification. In *2016 23rd International Conference on Pattern Recognition (ICPR)* (pp. 2440–2445).

Bray, F., Ferlay, J., Soerjomataram, I., Siegel, R., Torre, L., & Jemal, A. (2018). Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA Cancer Journal for Clinicians*, *68*, 394–424. doi:10.3322/caac.21492.

Bndi, P., Geessink, O., Manson, Q., Van Dijk, M., Balkenhol, M., Hermsen, M., Ehteshami Bejnordi, B., Lee, B., Paeng, K., Zhong, A., Li, Q., Zanjani, F. G., Zinger, S., Fukuta, K., Komura, D., Ovtcharov, V., Cheng, S., Zeng, S., Thagaard, J., Dahl, A. B., Lin, H., Chen, H., Jacobsson, L., Hedlund, M., etin, M., Halc, E., Jackson, H., Chen, R., Both, F., Franke, J., Ksters-Vandevelde, H., Vreuls, W., Bult, P., van Ginneken, B., van der Laak, J., & Litjens, G. (2019). From detection of individual metastases to classification of lymph node status at the patient level: The camelyon17 challenge. *IEEE Transactions on Medical Imaging*, *38*, 550–560. doi:`10.1109/TMI.2018.2867350`.

Cancer Tomorrow (2020). `https://gco.iarc.fr/tomorrow/graphic-isotype`. Online; accessed 07 Oct 2020.

Chang, Y. H., Thibault, G., Madin, O., Azimi, V., Meyers, C., Johnson, B., Link, J., Margolin, A., & Gray, J. W. (2017). Deep learning based Nucleus Classification in pancreas histological images. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (pp. 672–675).

Chatap, N., & Shibu, S. (2014). Analysis of blood samples for counting leukemia cells using support vector machine and nearest neighbour. *IOSR Journal of Computer Engineering*, *16*, 79–87.

Chauhan, A., Chauhan, D., & Rout, C. (2014). Role of gist and phog features in computer-aided diagnosis of tuberculosis without segmentation. *PloS One*, *9*, e112980.

Deng, S., Zhang, X., Yan, W., Chang, E. I. C., Fan, Y., Lai, M., & Xu, Y. (2020). Deep learning in digital pathology image analysis: a survey. *Front. Med.*, *14*, 470487.

Ding, Y., Yang, Y., & Cui, Y. (2019). Deep learning for classifying of white blood cancer. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 33–41). Springer Singapore.

Duggal, R., Gupta, A., Gupta, R., & Mallick, P. (2017). SD-Layer: Stain deconvolutional layer for cnns in medical microscopic imaging. In M. Descoteaux, L. Maier-Hein, A. Franz, P. Jannin, D. L. Collins, & S. Duchesne (Eds.), *Medical Image Computing and Computer Assisted Intervention – MICCAI 2017* (pp. 435–443). Springer International Publishing.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J. et al. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, *542*, 115–118.

Gao, Z., Wang, L., Zhou, L., & Zhang, J. (2017). Hep-2 cell image classification with deep convolutional neural networks. *IEEE Journal of Biomedical and Health Informatics*, *21*, 416–428.

Gehlot, S., Gupta, A., & Gupta, R. (2020a). EDNFC-Net: Convolutional Neural Network with Nested Feature Concatenation for Nuclei-Instance Segmentation. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1389–1393). IEEE.

Gehlot, S., Gupta, A., & Gupta, R. (2020b). SDCT-AuxNet$^{\theta}$: DCT augmented stain deconvolutional CNN with auxiliary classifier for cancer diagnosis. *Medical Image Analysis*, *61*, 101661. doi:https://doi.org/10.1016/j.media.2020.101661.

Goswami, S., Mehta, S., Sahrawat, D., Gupta, A., & Gupta, R. (2020). Heterogeneity loss to handle intersubject and intrasubject variability in cancer. arXiv:2003.03295.

Gupta, A., Duggal, R., Gehlot, S., Gupta, R., Mangal, A., Kumar, L., Thakkar, N., & Satpathy, D. (2020). GCTI-SN: geometry-inspired chemical and tissue invariant stain normalization of microscopic medical images. *Medical Image Analysis*, *65*, 101788.

Gupta, A., & Gupta, R. (2019a). ALL challenge dataset of ISBI 2019 [Dataset]. *The Cancer Imaging Archive*, . URL: https://doi.org/10.7937/tcia.2019.dc64i46r.

Gupta, A., & Gupta, R. (2019b). *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging*. Springer, Singapore. doi:https://doi.org/10.1007/978-981-15-0798-4.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., & Sugiyama, M. (2018a). Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems 31* (pp. 8527–8537).

Han, S. S., Kim, M. S., Lim, W., Park, G. H., Park, I., & Chang, S. E. (2018b). Classification of the Clinical Images for Benign and Malignant Cutaneous Tumors Using a Deep Learning Algorithm. *Journal of Investigative Dermatology*, *138*, 1529 – 1538. doi:https://doi.org/10.1016/j.jid.2018.01.028.

Han, X.-H., Lei, J., & Chen, Y.-W. (2016). HEp-2 Cell Classification Using K-Support Spatial Pooling in Deep CNNs. In G. Carneiro, D. Mateus, L. Peter, A. Bradley, J. M. R. S. Tavares, V. Belagiannis, J. P. Papa, J. C. Nascimento, M. Loog, Z. Lu, J. S. Cardoso, & J. Cornebise (Eds.), *Deep Learning and Data Labeling for Medical Applications* (pp. 3–11).

Harangi, B. (2018). Skin lesion classification with ensembles of deep convolutional neural networks. *Journal of Biomedical Informatics*, *86*, 25 – 32. doi:https://doi.org/10.1016/j.jbi.2018.08.006.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778).

Hekler, A., Utikal, J. S., Enk, A. H., Solass, W., Schmitt, M., Klode, J., Schadendorf, D., Sondermann, W., Franklin, C., Bestvater, F., Flaig, M. J., Krahl, D., von Kalle, C., Frhling, S., & Brinker, T. J. (2019). Deep learning outperformed 11 pathologists in the classification of histopathological melanoma images. *European Journal of Cancer*, *118*, 91 – 96. doi:`https://doi.org/10.1016/j.ejca.2019.06.012`.

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2261–2269).

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <0.5mb model size. *arXiv:1602.07360*, .

Jaeger, S., Candemir, S., Antani, S., Wáng, Y.-X. J., Lu, P.-X., & Thoma, G. (2014). Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative Imaging in Medicine and Surgery*, *4*, 475.

Jiang, F., Liu, H., Yu, S., & Xie, Y. (2017). Breast mass lesion classification in mammograms by transfer learning. In *Proceedings of the 5th International Conference on Bioinformatics and Computational Biology* (p. 5962). New York, NY, USA. doi:`10.1145/3035012.3035022`.

Joshi, M. D., Karode, A. H., & Suralkar, S. (2013). White blood cells segmentation and classification to detect acute leukemia. *International Journal of Emerging Trends and Technology in Computer Science (IJETTCS)*, *2*, 147–151.

Karimi, D., Dou, H., Warfield, S. K., & Gholipour, A. (2020). Deep learning with noisy labels: Exploring techniques and remedies in medical image analysis. *Medical Image Analysis*, *65*, 101759.

Karthikeyan, D. R., & Poornima, N. (2017). Micros-copic image segmentation using fuzzy c means for leukemia diagnosis. *International Journal of Advanced Research in Science, Engineering and Technology*, *4*.

Kazemi, F., Abbasian Najafabadi, T., & Nadjar Araabi, B. (2015). Automatic Recognition of Acute Myelogenous Leukemia in Blood Microscopic Images Using K-means Clustering and Support Vector Machine. *Journal of Medical Signals and Sensors*, *5*, 49–58.

Lee, K., He, X., Zhang, L., & Yang, L. (2018). Cleannet: Transfer learning for scalable image classifier training with label noise. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 5447–5456). doi:`10.1109/CVPR.2018.00571`.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., van der Laak, J. A., van Ginneken, B., & Snchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60 – 88. doi:https://doi.org/10.1016/j.media.2017.07.005.

Liu, Y., & Long, F. (2019). Acute lymphoblastic leukemia cells image analysis with deep bagging ensemble learning. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 113–121). Springer Singapore.

Liu, Y., Wu, Y.-H., Ban, Y., Wang, H., & Cheng, M.-M. (2020). Rethinking computer-aided tuberculosis diagnosis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. In V. Ferrari, M. Hebert, C. Sminchisescu, & Y. Weiss (Eds.), *Computer Vision – ECCV 2018* (pp. 122–138). Cham.

Madhukar, M., Agaian, S., & Chronopoulos, A. T. (2012). New decision support tool for acute lymphoblastic leukemia classification. In *Proc. SPIE 8295, Image Processing: Algorithms and Systems X; and Parallel Processing for Imaging Applications II, 829518*.

Mazo, C., Bernal, J., Trujillo, M., & Alegre, E. (2018). Transfer learning for classification of cardiovascular tissues in histological images. *Computer Methods and Programs in Biomedicine*, *165*, 69 – 76. doi:https://doi.org/10.1016/j.cmpb.2018.08.006.

Meng, N., Lam, E. Y., Tsia, K. K., & So, H. K. (2019). Large-Scale Multi-Class Image-Based Cell Classification With Deep Learning. *IEEE Journal of Biomedical and Health Informatics*, *23*, 2091–2098.

Mishra, S., Majhi, B., & Sa, P. K. (2019). Texture feature based classification on microscopic blood smear for acute lymphoblastic leukemia detection. *Biomedical Signal Processing and Control*, *47*, 303 – 311.

Mishra, S., Majhi, B., Sa, P. K., & Sharma, L. (2017). Gray level co-occurrence matrix and random forest based acute lymphoblastic leukemia detection. *Biomedical Signal Processing and Control*, *33*, 272–280.

Mohapatra, S., Patra, D., & Satpathy, S. (2014). An ensemble classifier system for early diagnosis of acute lymphoblastic leukemia in blood microscopic images. *Neural Computing and Applications*, *24*, 1887–1904.

Mohapatra, S., Samanta, S. S., Patra, D., & Satpathi, S. (2011). Fuzzy based blood image segmentation for automated leukemia detection. In *2011 International Conference on Devices and Communications (ICDeCom)* (pp. 1–5).

Multiple Myeloma (2020). https://www.cancer.org/. Online; accessed 07 Oct 2020.

Neoh, S. C., Srisukkham, W., Zhang, L., Todryk, S. et al. (2015). An intelligent decision support system for leukaemia diagnosis using microscopic blood images. *Scientific Reports*, *5*, 1–14.

Pan, Y., Liu, M., Xia, Y., & Shen, D. (2019). Neighborhood-correction algorithm for classification of normal and malignant cells. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 73–82). Springer Singapore.

Patel, N., & Mishra, A. (2015). Automated leukaemia detection using microscopic images. *Procedia Computer Science*, *58*, 635 – 642.

Phan, H. T. H., Kumar, A., Kim, J., & Feng, D. (2016). Transfer learning of a convolutional neural network for HEp-2 cell image classification. In *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)* (pp. 1208–1211).

Prellberg, J., & Kramer, O. (2019). Acute lymphoblastic leukemia classification from microscopic images using convolutional neural networks. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 53–61). Springer Singapore.

Putzu, L., Caocci, G., & Ruberto, C. D. (2014). Leukocyte classification for leukaemia detection using image processing techniques. *Artificial Intelligence in Medicine*, *62*, 179 – 191.

Qin, F., Gao, N., Peng, Y., Wu, Z., Shen, S., & Grudtsin, A. (2018). Fine-grained leukocyte classification with deep residual learning for microscopic images. *Computer Methods and Programs in Biomedicine*, *162*, 243 – 252. doi:`https://doi.org/10.1016/j.cmpb.2018.05.024`.

Rawat, J., Singh, A., Bhadauria, H. S., Virmani, J. et al. (2017a). Classification of acute lymphoblastic leukaemia using hybrid hierarchical classifiers. *Multimedia Tools and Applications*, *76*, 19057–19085.

Rawat, J., Singh, A., HS, B., Virmani, J. et al. (2017b). Computer assisted classification framework for prediction of acute lymphoblastic and acute myeloblastic leukemia. *Biocybernetics and Biomedical Engineering*, *37*, 637 – 654.

Rehman, A., Abbas, N., Saba, T., Rahman, S. I. u. et al. (2018). Classification of acute lymphoblastic leukemia using deep learning. *Microscopy Research and Technique*, *81*, 1310–1317.

Reta, C., Altamirano, L., Gonzalez, J. A., Diaz-Hernandez, R. et al. (2015). Segmentation and classification of bone marrow cells images using contextual information for medical diagnosis of acute leukemias. *PLOS ONE*, *10*, 1–18.

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 4510–4520).

Shafique, S., & Tehsin, S. (2018). Acute lymphoblastic leukemia detection and classification of its subtypes using pretrained deep convolutional neural networks. *Technology in Cancer Research & Treatment*, *17*.

Shah, S., Nawaz, W., Jalil, B., & Khan, H. A. (2019a). Classification of normal and leukemic blast cells in b-all cancer using a combination of convolutional and recurrent neural networks. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 23–31). Springer Singapore.

Shah, S. C., Kayamba, V., Peek, R. M., & Heimburger, D. (2019b). Cancer Control in Low- and Middle-Income Countries: Is It Time to Consider Screening? *Journal of Global Oncology*, (pp. 1–8). doi:10.1200/JGO.18.00200.

Sharma, H., Zerbe, N., Klempert, I., Hellwich, O., & Hufnagl, P. (2017). Deep convolutional neural networks for automatic classification of gastric carcinoma using whole slide images in digital histopathology. *Computerized Medical Imaging and Graphics*, *61*, 2 – 13. doi:https://doi.org/10.1016/j.compmedimag.2017.06.001. Selected papers from the 13th European Congress on Digital Pathology.

Shi, T., Wu, L., Zhong, C., Wang, R., & Zheng, W. (2019). Ensemble convolutional neural networks for cell classification in microscopic images. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 43–51). Springer Singapore.

Singhal, V., & Singh, P. (2016). Texture features for the detection of acute lymphoblastic leukemia. In *Proceedings of International Conference on ICT for Sustainable Development* (pp. 535–543).

Sirinukunwattana, K., Raza, S. E. A., Tsang, Y., Snead, D. R. J., Cree, I. A., & Rajpoot, N. M. (2016). Locality sensitive deep learning for detection and classification of nuclei in routine colon cancer histology images. *IEEE Transactions on Medical Imaging*, *35*, 1196–1206.

Springenberg, J., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2015). Striving for simplicity: The all convolutional net. In *ICLR (workshop track)*.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2818–2826).

Szegedy, C., Wei Liu, Yangqing Jia, Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–9).

Tabibu, S., Vinod, P., & Jawahar, C. (2019). Pan-Renal Cell Carcinoma classification and survival prediction from histopathology images using deep learning. *Sci Rep*, *9*, 10509. doi:https://doi.org/10.1038/s41598-019-46718-3.

TBX11K Tuberculosis Classification and Detection Challenge (2020). https://competitions.codalab.org/competitions/25848. Online; accessed 02 Feb 2020.

The Global Cancer Observatory (2020). https://gco.iarc.fr/. Online; accessed 07 Oct 2020.

Veit, A., Alldrin, N., Chechik, G., Krasin, I., Gupta, A., & Belongie, S. (2017). Learning from noisy large-scale datasets with minimal supervision. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 6575–6583). doi:10.1109/CVPR.2017.696.

Verma, E., & Singh, V. (2019). ISBI challenge 2019: Convolution neural networks for b-all cell classification. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 131–139). Springer Singapore.

Vincent, I., Kwon, K., Lee, S., & Moon, K. (2015). Acute lymphoid leukemia classification using two-step neural network classifier. In *2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)* (pp. 1–4).

Vogado, L. H., Veras, R. M., Araujo, F. H., Silva, R. R. et al. (2018). Leukemia diagnosis in blood slides using transfer learning in cnns and svm for classification. *Engineering Applications of Artificial Intelligence*, *72*, 415 – 422.

Vogado, L. H. S., Veras, R. D. M. S., Andrade, A. R., Araujo, F. H. D. D. et al. (2017). Diagnosing leukemia in blood smear images using an ensemble of classifiers and pre-trained convolutional neural networks. In *2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)* (pp. 367–373).

Wong, K. C., Syeda-Mahmood, T., & Moradi, M. (2018). Building medical image classifiers with very limited data using segmentation networks. *Medical Image Analysis*, *49*, 105 – 116.

Xiao, F., Kuang, R., Ou, Z., & Xiong, B. (2019). DeepMEN: Multi-model ensemble network for b-lymphoblast cell classification. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 83–93). Springer Singapore.

Xie, S., Girshick, R., Dollr, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 5987–5995).

Xie, X., Li, Y., Zhang, M., Wu, Y., & Shen, L. (2019). Multi-streams and multi-features for cell classification. In A. Gupta, & R. Gupta (Eds.), *ISBI 2019 C-NMC Challenge: Classification in Cancer Cell Imaging* (pp. 95–102). Springer Singapore.

Xu, Y., Jia, Z., Ai, Y., Zhang, F. et al. (2015). Deep convolutional activation features for large scale brain tumor histopathology image classification and segmentation. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 947–951).

Yang, H., Zhang, X., Yin, F., & Liu, C. (2018). Robust classification with convolutional prototype learning. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 3474–3482).

Zhang, L., Le Lu, Nogues, I., Summers, R. M., Liu, S., & Yao, J. (2017). DeepPap: Deep Convolutional Networks for Cervical Cell Classification. *IEEE Journal of Biomedical and Health Informatics*, *21*, 1633–1643.