Semi-Supervised Breast Cancer Histopathology Classification: Final Report

G014 (s1961570, s1994033, s1928467)

Abstract

Semi-supervised learning allows one to leverage unlabeled data to improve classifier performance. Due to the expense and expertise required to label medical data, semi-supervised learning could improve the diagnostic capabilities of deep learning tools in a medical context. We experiment with semi-supervised learning algorithms called MixMatch and FixMatch in order to minimize the number of labeled training examples required to perform binary classification on BreaKHis, a breast cancer imaging benchmark data set. We show that by using only 5 labeled data points for every sub-class, we can achieve a test F_1 score of 0.822 ± 0.001. We additionally demonstrate that by pre-training our network weights using an autoencoder we can greatly reduce training stochasticity.

1. Introduction

Developments in medical imaging techniques have yielded considerable increases in the diagnostic capabilities of medical professionals. This has resulted in the collection of vast quantities of medical imaging data, which could potentially be leveraged to create automated diagnostic tools (Jiang et al., 2019). Breast cancer screenings are a highly time-consuming specialized task and involve multiple testing methods, though if a tumor is found, a biopsy must be performed to determine whether the tumor is cancerous. This examination is performed by experts with many years of experience and perhaps under enormous workload (Gurcan et al., 2009).

Computer-aided diagnostic methods could be extremely useful in improving diagnostic efficiency and distributing expertise intelligently (Araujo et al., 2017). There are a number of challenges in implementing such systems. Given that a false positive result may subject a healthy patient to invasive or damaging treatment, while a false negative might cause a sick patient to miss a life-saving treatment, computer-aided diagnoses must operate at extremely high precision in order to be used in practice. To achieve this level of performance through machine learning methods, massive data sets are required to mitigate underlying biases and confirm experimental efficacy. Due to the expertise required to annotate such data sets, and the possible ethical concerns in releasing personal data from patient records, collecting such data sets can be extremely costly and practically unfeasible (de Bruijne, 2016; Weese & Lorenz, 2016).

Semi-supervised learning methods allow one to leverage unlabeled data in order to boost model performance on tasks such as classification. The implementation of such a method in the medical imaging domain would allow for models to be created based on much larger distributions of data and significantly reduce the cost of collecting and annotating it (Litjens et al., 2017). Our project will experiment with semi-supervised techniques in order to minimize the number of labeled examples required to maximize accuracy and F_1 score. We bound our target accuracy based on the fully supervised upper bounds set in the literature (where accuracies in excess of 90% have been demonstrated for binary classification across all magnifications (Toğaçar et al., 2019)). Additionally we validate that our architecture can reach an F_1 score of 0.9 on a fully supervised task (Table 2).

The data with which we will experiment with is the BreaKHis (Spanhol et al., 2016) dataset of microscopic images of breast tumor tissue samples (treated with a coloured stain) collected from anonymised patients at the P&D Laboratory – Pathological Anatomy and Cytopathology, Parana, Brazil. These images are classified as benign (non-cancerous) if they do not contain any malignant criteria, or malignant (cancerous) otherwise. The classification task is naturally difficult given the degree of expertise required for a human classifier, but the dataset features other technical challenges such as the latent variable of the patient, the unbalanced class proportions, and varying magnifications (Toğaçar et al., 2019).

The main objectives for this project can be summarized in the following points:

- Implement a neural network architecture based on DenseNet (Huang et al., 2017). This will serve as our supervised network that is trained using labeled data, to be paired with a semi-supervised learning algorithm.
- Implement the MixMatch semi-supervised algorithm (Berthelot et al., 2019). This will serve as our semi-supervised baseline.
- Improve upon Mixmatch by testing a combination of methods borrowed from semi-supervised learning literature.
- Investigate whether using unsupervised methods like convolutional autoencoders to pre-train our network helps with semi-supervised learning (Chen et al., 2017).

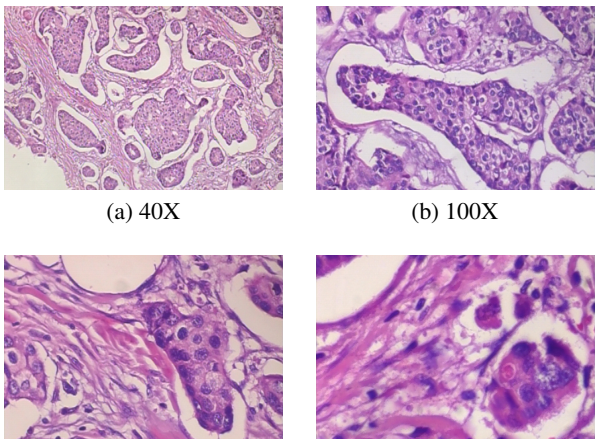
The rest of the structure of this report is as follows: In Section 2, we describe the dataset and the evaluation metrics

that were used. Section 3 explores the methodology used in our experimentation. Meanwhile, Section 4 provides the experiments that were conducted and their respective results. Finally, Sections 5 and 6 provides related work in this medical field, conclusions and future work.

2. Dataset and Task

2.1. Dataset Description

The dataset that was used for this work is the Breast Cancer Histopathological Image Classification dataset (BreaKHis) (Spanhol et al., 2016) [1]. BreaKHis is composed of 7,909 images of histopathological breast tumor tissue taken from 82 patients. Every image is made up of 3 RGB channels with dimensions of 700x460 pixels. It contains samples obtained using various magnification factors (40X, 100X, 200X, and 400X), examples of which can be seen in Figure 1.



(a) 40X (b) 100X

(c) 200X (d) 400X

Figure 1. Example of Malignant breast tumor viewed with different magnifications. The samples are obtained from the same patient (Spanhol et al., 2016).

The images in the BreaKHis dataset are first classified as containing either benign or malignant tumors. Both classifications are then sub-divided into 4 sub-classes, as shown in Table 1. As can be seen in Table 1, the classes are heavily imbalanced (though class imbalance may reflect the true skew of a population who have undergone a breast tumor biopsy). We address the issue of class imbalance by randomly selecting the same amount of images per subclass since we only need small amount of labeled data. For evaluation, we used F_1 score instead of accuracy.

In our experiments, we focused only on the binary classification task. We determined that this would be sufficient as if such a system were implemented in a medical context, it would likely be used operationally as a flagging tool; i.e. a tool to recommend whether a patient needs to be seen by a specialist or not, and not as a diagnostic tool for say, recommending potential treatment. Given this justification of the binary task, we proceeded with experiments but suggest the harder multi-class problem as a further research area.

In pre-processing the data we used an approximate 60:20:20 ratio for the training, validation, and test sets respectively. We forced the patients to be disjoint across partitions, hence the approximation of the splits. This ensures that images of the same patient cannot be found in multiple sets, reducing the risk of data bleeding, which is consistent with most of the literature (Bayramoglu et al., 2016). We then randomly sub-divided the training data into labelled and 'unlabelled' partitions of varying sizes for each magnification. We primarily experimented with 5 labelled training images, treating the remaining training images as unlabelled. We reduced the image size to 224x224 and normalised images for each magnification using statistics calculated on the training partition.

2.2. Task and Evaluation Metrics

Our primary task is to explore semi-supervised learning methods in order to reduce the size of our labelled partition while maintaining as high an F_1 score as possible.

Due to the unbalanced class distributions, and the importance of reducing false positives and false negatives in a medical setting, we opted to use $F_1\,score$ instead of accuracy as our primary evaluation metric. $F_1\,score$ is calculated by first calculating the precision and recall using:

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN} \tag{1}$$

where TP is the number of true positives, FP the number of false positives and FN is the number of false negatives. Using precision and recall, f_1 is calculated using the following equation:

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{2}$$

We also reported the cross-entropy (CE) loss as a secondary metric. When the loss is lower, it indicates that the model has higher confidence in the classification made. The equation for the cross-entropy is:

$$CE = -\sum_{i}^{C} t_i \log(s_i) \tag{3}$$

where t_i is the groundtruth, s_i is the normalized predicted score (passed through a softmax) and C are the number of classes.

Our model selection procedure involved choosing our 'best' model by the highest F_1 score across all training epochs in all experiments, then computing test set results on that model.

[1] Available at https://web.inf.ufpr.br/vri/databases/breast-cancer-histopathological-database-BreaKHis/

Class	Sub-Class	Magnification				Patients	Total
		40X	100X	200X	400X		
Benign	Adenosis (A)	114	113	111	106	4	444
	Fibroadenoma (F)	253	260	264	237	10	1014
	Tubular Adenoma (TA)	109	121	108	115	3	453
	Phyllodes Tumor (PT)	149	150	140	130	7	569
Malignant	Ductal Carcinoma (DC)	864	903	896	788	38	3451
	Lobular Carcinoma (LC)	156	170	163	137	5	626
	Mucinous Carcinoma (MC)	205	222	196	169	9	792
	Papillary Carcinoma (PC)	145	142	135	138	6	560
Total		1995	2081	2013	1820	82	7909

Table 1. The BreakHis dataset consisting of 7,909 histopathological images from 82 patients. The images are divided into Benign and Malignant tumors, and each class has 4 further sub-classes.

3. Methodology

The core architecture with which we will be conducting our baseline and future experiments is called **DenseNet** (Huang et al., 2017). In a densely connected network, the output feature maps of each convolutional layer are concatenated with all the previous outputs, such that every layer sees every previous feature map as input while also being connected to every subsequent layer. The DenseNet architecture is described by its growth rate (number of new feature maps created in each layer), and block configuration (number of blocks/layers per block). This design allows for the network gradient to efficiently flow through the entire network as in ResNet (He et al., 2016), thus avoiding the problem of vanishing gradient seen in deep neural network architectures. Additionally, DenseNet requires much fewer parameters than its counterparts, such as ResNet, as the continuous concatenation of new feature maps eliminates the possible need to redundantly relearn features (Huang et al., 2017). This architecture has achieved state-of-the-art results on various image benchmark tasks and has proven efficacy on BreaKHis with such architectures yielding results above 96% accuracy (Toğaçar et al., 2019).

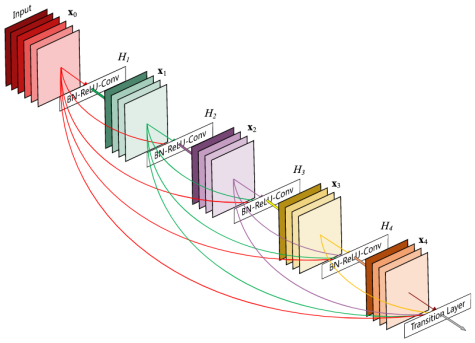


Figure 2. A dense block with 5 layers and a growth rate of 4. The arrows denote that the feature maps are concatenated with the inputs to the indicated stage. Figure from (Huang et al., 2017).

The following techniques are those which we included in our primary experiments with semi-supervised learning and fine-tuning our final architecture.

A **Convolutional Block Attention Module (CBAM)** is an attention module that re-calibrates the outputs of a convolutional layer in order to weight the most important spatial and channel-wise information (Hu et al., 2018; Woo et al., 2018; Toğaçar et al., 2019). CBAM first computes average and max pooling vectors over channels. These vectors are both fed through a shared linear layer with one hidden layer and ReLU activation, their outputs are then added before an element-wise sigmoid operation is applied and the output channels are scaled by their corresponding weights. This process is illustrated in Figure 3.

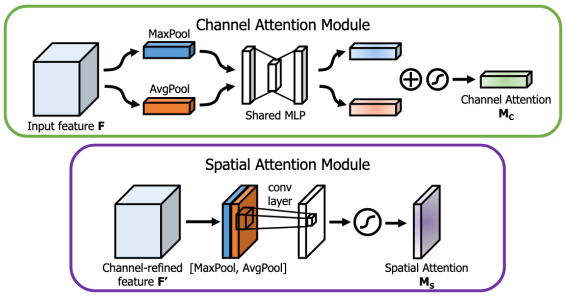


Figure 3. Figure illustrating the CBAM channel and spatial attention modules. Figure from (Woo et al., 2018)

CBAM then takes these attention weighted channels and performs an average-pooling and max-pooling operation across the channel dimension, yielding a $(2, H, W)$ output. A 7×7 convolution is then applied, followed by an element-wise sigmoid to produce a $H \times W$ dimensional matrix of spatial attention weights which are used to scale the convolutional block outputs across the spatial dimensions.

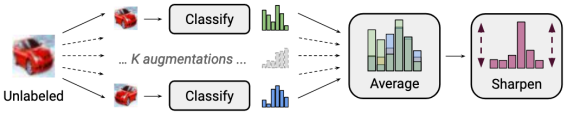


Figure 4. Figure illustrating the MixMatch label sharpening process. Figure from (Berthelot et al., 2019).

MixMatch is a highly peer-reviewed semi-supervised learning algorithm that is proven to work on several standard

image benchmarks (Berthelot et al., 2019). MixMatch leverages unlabeled data at each training epoch via a data augmentation method. For each batch of labeled training examples, a batch of unlabeled examples U is considered. Each $u_b \in U$ is augmented k times (via affine transformations), the average model softmax prediction (the average distribution over label predictions) is taken across all augmentations of u_b and then sharpened, i.e. the distribution temperature is adjusted to transform the distribution closer towards a discrete distribution with a spike at the most likely label. This process can be seen in Figure 4. All augmented unlabeled examples with their associated sharpened labels are then collected in a set M along with augmented labeled training examples for that batch and shuffled.

MixUp takes two sets of training examples and randomly interpolates pairs of examples and their respective label softmax distributions. This yields a set of interpolated training examples with interpolated labels (Zhang et al., 2018). MixUp is applied to the augmented labeled training examples $\tilde{X}$ and one partition of M, M_X, and again to the the augmented unlabeled examples $\tilde{U}$ and the remainder of M, M_U. The model is then trained on the resulting sets $X^* = MixUp(\tilde{X}, M_X)$ and $U^* = MixUp(\tilde{U}, M_U)$. During training the loss L is computed separately for X^* and U^*, where $L = L_X + \lambda_U L_U$. L_X is cross entropy loss computed over examples in X^*, while L_X is mean square error loss computed over examples in U^*. The unsupervised loss weight hyperparameter λ_U is increased linearly during training. MixMatch has reported error rates of 11.80% and 6.00% on Cifar-10[2] using 250 and 4000 labeled data points respectively when paired with a wide ResNet-28 architecture (Berthelot et al., 2019).

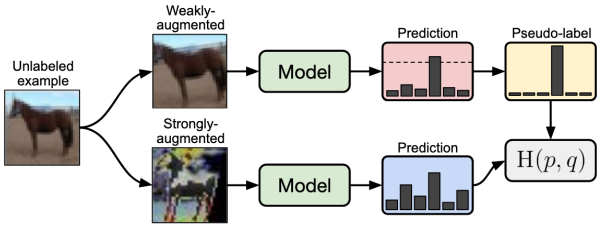


Figure 5. Figure illustrating the FixMatch pseudo-labeling and data augmentation process, for some loss function H. Figure from (Sohn et al., 2020).

FixMatch is a recently proposed alternative to MixMatch (Sohn et al., 2020). At each training epoch, FixMatch predicts pseudo-labels on weakly augmented versions of the unlabeled examples by querying the model itself. If the softmax layer for one such example x predicts some class y_i with confidence over a defined threshold (τ), then a strongly augmented version of x is added to the training set for that epoch with corresponding label y_i. Enforcing the loss against a strongly augmented version of the image enforces consistency regularization, thereby improving model generalization. For strong augmentations, we used RandAugment (Cubuk et al., 2019). RandAugment randomly applies N transformations (e.g. solarize and contrast adjustment) where a single fixed hyper-parameter, M, controls the strength of all distortions. For every epoch, we consider *batch size* $\times$ *unlabeled ratio* (μ) unlabeled examples where μ is a hyper-parameter. Early published results for FixMatch demonstrate accuracies of 88% and 78% on Cifar-10 for 4 and 1 labeled examples per class, respectively.

An **autoencoder** is a neural network model which learns a lower-dimensional representation of the input data, without any supervision (Rumelhart et al., 2013). One way this is done is by training the network to recreate its input signal after passing it through a lower-dimensional hidden layer that acts as a bottleneck. We calculate the reconstruction loss using a mean square error (MSE) loss function. The autoencoder network can be divided into two sub-architectures, the encoder, which extracts the features, and the decoder which reconstructs the input using those features. We used our fine-tune DenseNet, which will be explained in Section 4, as the encoder to be pre-trained. For our decoder, we approximately reverse our encoder architecture using ResNet inspired residual connections (He et al., 2016). ResNet involves 'skip connections'; the output of the previous layer is added in the output of the current layer. This improves gradient predictivity throughout the network, which positively impacts training convergence in deep networks. Typically, this is implemented in blocks of two $3x3$ convolutional layers, such that the input 'skips' every second layer. We do not use the output signal of the decoder at any stage and instead work to enforce a well-learned representation in the encoder stage so as to reuse the training weights later to initialize our classifier model.

We hypothesize that by pre-training our network weights with an autoencoder on the full (labeled and unlabeled) training data, the semi-supervised algorithms could leverage the learned representations to improve their predictive performance when trained on small amounts of labeled data (Chen et al., 2017). This should be the case since the network will be initialized with knowledge of features from the entire distribution, not just those learned from the labeled data, and hence the pseudo-labeling processes should be better 'informed'.

4. Experiments

For this report, we narrowed our experiments to only the 40X magnification data and used only 5 labeled images per subclass (40 images in total). We opted to focus on 40X magnification as this magnification generally receives the the lowest accuracy in the literature (for example in (Nawaz et al., 2018)), and hence was the hardest case. As will be evident in the following sub-sections, tuning the semi-supervised algorithm for three other magnifications and increasing the number of labeled data is unfeasible due to the time constraints of this project. Since the semi-supervised algorithms we are experimenting with deal with randomness, we ran every experiment using three different

[2]https://www.cs.toronto.edu/ kriz/cifar.html

seeds, reporting the average results with their respective standard errors. Finally, unless stated otherwise we trained the models for 100 epochs with a batch size of 20.

4.1. MixMatch

For our baseline semi-supervised model, we chose the MixMatch algorithm due to its proven track record in semi-supervised learning (Berthelot et al., 2019). Following implementations found in recent literature, we started by applying MixMatch to a Densenet-121 used by Nawaz et al., with block configuration (6, 12, 24, 16). For a fair comparison with experiments in later sub-sections, we fine-tuned the DenseNet architecture, as well as the hyperparameters of both MixMatch and the optimizer. Prior to fine tuning, for proof of concept, we trained the fully supervised network on both the full dataset and reduced partition of training labels to validate that our architecture can achieve good results, and that our MixMatch implementation does in fact yield an improvement. The hyper-parameters were matched to those found in (Nawaz et al., 2018), and the results which confirm our architectures are in Table 2.

We began by fine-tuning the number of convolutional layers per block coupled with their respective growth rate and initial number of filters. Due to the large number of feature maps utilized throughout DenseNet, we experimented with increasing the dilation of the kernel in each progressive layer of each block (Antoniou et al., 2018). The dilation per convolutional layer i for $i = (0, 1, 2..)$ in block j is $d_{i,j} = 2^i$. This increases the receptive field of some feature maps hence allowing the network to efficiently extract both local and global features. We also experimented with CBAM, to allow the network to dynamically focus on particular spatial areas of classification interest, across the large series of feature maps produced by DenseNet (Woo et al., 2018).

The following are the architectures considered:

- Architectures = (6, 12, 24, 16) with 24 initial filters and 12 growth, (6, 6, 6, 6) with 24 initial filters and 12 growth, (4, 4, 4, 4) with 64 initial filters and 32 growth
- Increasing Dilation = True/False
- CBAM = True/False

For all the above experiments, we used the following settings: Stochastic Gradient Descent optimiser with Nesterov momentum of 0.9. Cosine Annealing scheduler (Loshchilov & Hutter, 2017) with initial learning rate of 0.01. Weight decay (L2) coefficient of 10^{-5}, dropout of 0.2, and batch size of 20 (Srivastava et al., 2014). The maximum unlabeled loss weight λ_U that is used by MixMatch was set to 75. Most of these parameters were set at their recommended defaults and were fine-tuned at a later stage.

After experimentally determining the most suitable architecture configuration by choosing the one with the best validation F_1 score from Table 2, we moved to fine-tuning the optimiser and MixMatch hyperparameters. The following hyperparameters were optimised through a grid search, the best results of which are reported in bold in Table 3.

- Unlabeled loss maximum λ_U = (50, 75)
- Dropout probability = (0, 0.1, 0.2)
- Weight decay value (L2) = (10^{-5}, 10^{-4}, 10^{-3})
- Learning rates = (0.1, 0.01)

After fine-tuning the baseline MixMatch algorithm, we got an average validation F_1 score of 0.817. Examining the training results across epochs in Figure 7a, we noticed that the training process itself is extremely stochastic. We theorized that this is likely due to the multiple randomized steps in MixMatch (augmentation and MixUp). We also conjecture that MixUp does not make much intuitive sense on this dataset. On say Cifar-10, interpolating an image of a car with a horse could yield a result whose label is arguably a distribution over car and horse labels as the image contains both classes. However, interpolating benign and malignant images is counter-intuitive as a tissue sample can never be both at the same time. We suppose this is likely the case and that MixMatch often ends up training with noisy data, yielding the observed stochastic process.

4.2. FixMatch

In researching how to improve on MixMatch, we wanted to find solutions that contain less stochastic elements, while also alleviating our intuitive concerns described earlier. Therefore, we first sought to replace MixMatch's label sharpening with pseudo-labeling (Lee, 2013) since this generates more confident labels. Additionally, we also wanted to replace MixUp with a data augmentation process that fits our use case better.

FixMatch fits the above requirements nicely as it uses pseudo-labeling and standard data augmentation processes. Moreover, it also uses a confidence threshold which discards unlabeled data with high entropy pseudo labels. As a result, the network will start using unlabelled data only when it has reached a sufficient stage of training maturity to make well informed, confident predictions.

Following our earlier methods, we tuned the architecture first, as seen in Table 2, keeping the same parameters static. After choosing the architecture with the highest validation F_1 score, we tuned the following optimization hyperparameters, the results of which are reported in Table 3:

- Dropout probability = (0, 0.1)
- Weight decay value (L2) = (10^{-4}, 10^{-5}, 10^{-6})
- Learning rate = (5×10^{-3}, 10^{-3}, 10^{-4})
- Loss λ_U is always 1

Given the architecture and optimiser with the best F_1 validation, we next tuned the specific hyperparameters of FixMatch, those being:

- Unlabeled factor=(1, 3, 4, 5)
- FixMatch confidence threshold = (0.95, 0.85)
- M Randaug = (5, 10)
- N Randaug = (1, 3)

After having found the best configuration for FixMatch (bolded F_1 score in Table 4), we wanted to research the

Algorithm	Architecture	Val F_1 score	Val Loss
Fully Supervised	(6, 12, 24, 16)	0.901 ± 0.014	0.205 ± 0.012
Fully Supervised (40 labeled)	(6, 12, 24, 16)	0.653 ± 0.019	1.972 ± 0.112
MixMatch	(4, 4, 4, 4)	**0.805 ± 0.013**	**0.384 ± 0.035**
	(4, 4, 4, 4) with CBAM	0.797 ± 0.024	0.454 ± 0.017
	(4, 4, 4, 4) with Dilation	0.799 ± 0.013	0.419 ± 0.021
	(6, 12, 24, 16) (Baseline)	0.770 ± 0.028	0.451 ± 0.007
	(6, 12, 24, 16) with Dilation and CBAM	0.803 ± 0.029	0.451 ± 0.030
FixMatch	(4, 4, 4, 4)	0.725 ± 0.058	0.510 ± 0.049
	(4, 4, 4, 4) with Dilation	0.753 ± 0.008	0.529 ± 0.063
	(4, 4, 4, 4) with Dilation and CBAM	**0.785 ± 0.015**	**0.389 ± 0.010**
	(6, 6, 6, 6) with Dilation	0.753 ± 0.008	0.503 ± 0.036
	(6, 12, 24, 16)	0.713 ± 0.015	0.474 ± 0.030

Table 2. The best 5 architecture results obtained when using MixMatch and FixMatch trained on 5 labeled images per subclass. The fully supervised model was trained on **all** the images in the training set, and hyper-parameters used were dropout of 0, learning rate of 0.1, and weight decay 0.001. These hyper-parameters were obtained using a gridsearch on the 3 hyper-parameters.

Algorithm	Loss lambda u	Learning Rate	Weight Decay	Dropout	Val F_1 score	Val Loss
MixMatch	50	0.01	10^{-4}	0.1	0.810 ± 0.028	0.389 ± 0.058
	75	0.01	10^{-5}	0.1	**0.817 ± 0.022**	0.412 ± 0.008
	75	0.01	10^{-4}	0.2	0.800 ± 0.019	**0.386 ± 0.060**
FixMatch	1	0.001	10^{-5}	0	0.812 ± 0.045	0.460 ± 0.085
	1	0.001	10^{-6}	0	0.812 ± 0.045	0.443 ± 0.099
	1	0.005	10^{-4}	0	**0.829 ± 0.006**	**0.334 ± 0.008**

Table 3. The best 3 results obtained when using MixMatch and FixMatch (fine-tuning parameters). Both MixMatch and FixMatch used block config of (4, 4, 4, 4), but FixMatch had both increasing dilation and CBAM while MixMatch had none.

kinds of augmentations that would best suit histopathological images. For labeled data augmentations as well as weak augmentations, we tested combinations of:

- Random Horizontal Flip (p=0.5)
- Random Vertical Flip (p=0.5)
- Random Affine: Translate = (0.1, 0.1) and degrees = 20
- Color Jitter: Brightness = 0.1, Jitter = 0.1

With regards to strong augmentations using RandAugment, we hypothesised that having augmentations that heavily distort the colour space would negatively impact the training process. This is because histopathological images rely on the colours obtained from staining to visualise cancerous tissue. Therefore, we removed the following transformations from RandAugment: Equalize, Invert, Posterize, Solarize, SolarizeAdd, Color. The results can be seen in Table 5

τ	μ	M	N	Val F_1 score	Val Loss
0.95	3	10	1	**0.832 ± 0.015**	0.382 ± 0.040
	1	10	3	0.810 ± 0.012	0.404 ± 0.041
	4	10	3	0.799 ± 0.021	0.477 ± 0.052
0.85	4	5	1	0.817 ± 0.027	0.438 ± 0.093
	5	5	3	0.809 ± 0.007	0.431 ± 0.076
	4	10	3	0.805 ± 0.020	**0.342 ± 0.017**

Table 4. FixMatch Algorithm hyper-parameter tuning. τ is the confidence threshold, μ is the ratio of unlabeled data, M is the strength of the distortion of the transformations and N is the total number of transformations done to each training example.

Analyzing the overall hyperparameter tuning results, we can see that the best tuned FixMatch configuration on average performs 0.045 better in validation F_1 score when compared to MixMatch. Interestingly, unlike MixMatch, we note that the best configuration includes the use of both CBAM and increased dilations. Removing the strong color distortions in RandAugment also proved to be a good strategy, thus confirming our hypothesis. Finally, when comparing the FixMatch training process with that of MixMatch in Figure 7b, the stochasticity of the validation loss has improved slightly, however the problem is still apparent.

4.3. Pre-Trained FixMatch

Since we are using only 5 images per sub-class as labeled training examples, we conjecture it is difficult for the network to generalize to new data, especially since FixMatch can only make highly confident predictions on unlabeled data very similar to our labeled ones. To mitigate this, we experimented with pretraining the network weights through an autoencoder. We hoped that learning features from the full training distribution would improve our network generalization and convergence, by giving FixMatch more robust features for its pseudo-label predictions in the early stages of training.

To this end, we built a convolutional autoencoder using our network architecture with previously optimised hyperparameters as the encoder architecture, and approximately mirrored the network using residual connections for the

RandAugment	Dataset	Transformations				Val F_1 score	Val Loss
		Hor. Flip	Vert. Flip	Random Affine	Color Jitter		
Normal	Labeled	$p = 0.5$	$p = 0.5$	-	-	0.835 ± 0.012	0.421 ± 0.092
	Unlabeled	$p = 0.5$	-	-	-		
	Labeled	$p = 0.5$	$p = 0.5$	$deg = \pm 20$	-	0.817 ± 0.001	0.431 ± 0.056
	Unlabeled	$p = 0.5$	-	-	-		
Custom	Labeled	$p = 0.5$	$p = 0.5$	-	-	0.819 ± 0.010	**0.330 ± 0.010**
	Unlabeled	$p = 0.5$	$p = 0.5$	-	-		
	Labeled	$p = 0.5$	$p = 0.5$	-	-	**0.849 ± 0.040**	0.380 ± 0.063
	Unlabeled	$p = 0.5$	$p = 0.5$	$deg = \pm 20$	-		

Table 5. Top 2 results for every transformation for Fixmatch fine-tuning. Labeled dataset indicates that the transformations were performed only on the labeled training set. Similar reasoning for the unlabeled.

decoder. We trained to minimize the reconstruction MSE loss on the entire (labelled and unlabeled) data set of 40X magnification training data. Moreover, we augmented the data with random horizontal flips during training, to provide some translational invariance.

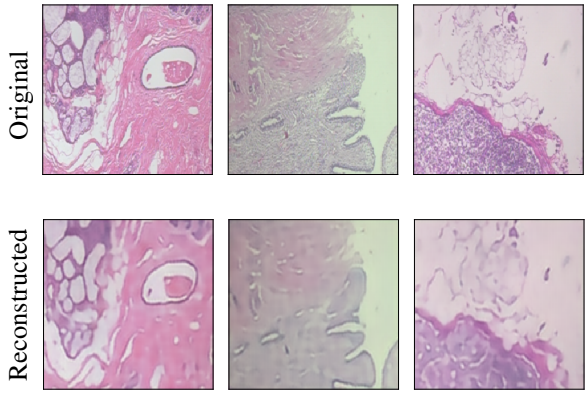


Figure 6. Randomly selected images reconstructed by the Autoencoder with 40X magnification.

After training for 300 epochs using Adam optimizer (Kingma & Ba, 2015) with cosine annealed learning rate of 0.01 and no dropout, we reached an MSE loss of 0.505 ± 0.011. To validate that the autoencoder had successfully learned the data distribution, we compared the input and output of the autoencoder on a random selection of images (as in Figure 6). Noting that the reconstructed images are well produced mitigated our concerns that the loss was not sufficiently minimizing. We can see that the autoencoder successfully reconstructs strong, large features but smooths out some of the finer resolution details in the images. We hypothesized that this was due to the complexity of reconstructing the very fine texture of tissue at 40X magnification. To verify this, we ran another experiment with the exact same configuration on the 400X magnification data, getting an average of 0.144 ± 0.006 MSE loss.

When fine-tuning FixMatch using the pre-trained network we experimented with freezing different sets of network weights. Ultimately we fine-tuned the last dense block, the final batchnorm (Ioffe & Szegedy, 2015), and final fully connected linear, freezing all other network weights. This process involved using the best FixMatch configuration

τ	LR	L2	Val F_1 score	Val Loss
0.95	0.01	10^{-5}	**0.816 ± 0.014**	0.468 ± 0.042
	0.01	10^{-7}	0.797 ± 0.005	**0.456 ± 0.029**
0.85	0.01	10^{-7}	0.588 ± 0.115	0.522 ± 0.021
	0.01	10^{-9}	0.584 ± 0.053	0.518 ± 0.040

Table 6. Pre-trained FixMatch Algorithm hyper-parameter tuning results. τ is the confidence threshold, LR is the learning rate and L2 is the weight decay.

from the previous section, and tuning the following hyper-parameters for 100 epochs:

- Learning Rate: (0.01, 0.001, 10^{-4})
- Weight Decay: (10^{-9}, 10^{-7}, 10^{-5})
- FixMatch Confidence Threshold: (0.85, 0.95)

Analyzing the validation F_1 scores in Table 6 we found that unfortunately the pre-trained FixMatch algorithm did not result in any improvements. We conjecture that this might be due to the encoder's inability to extract fine-grained textural details of the images, which might be important predictors for the classification task. A benefit we noticed from using the pre-trained weights is that the training process as depicted in Figure 7c is much less stochastic than in previous configurations. When comparing the validation F_1 scores across epochs for the three semi-supervised algorithms in Figure 7d, the reduction in stochasticity is even more apparent. It is important to note that this reduction is present despite a high learning rate. We believe this result to be significant as on BreaKHis, even fully-supervised algorithms are extremely stochastic (Toğaçar et al., 2019; Nawaz et al., 2018). Additionally, with a smoother semi-supervised training process, it will be easier for future research to be conducted on this topic as an analysis of network behavior would be much easier.

4.4. Test Results

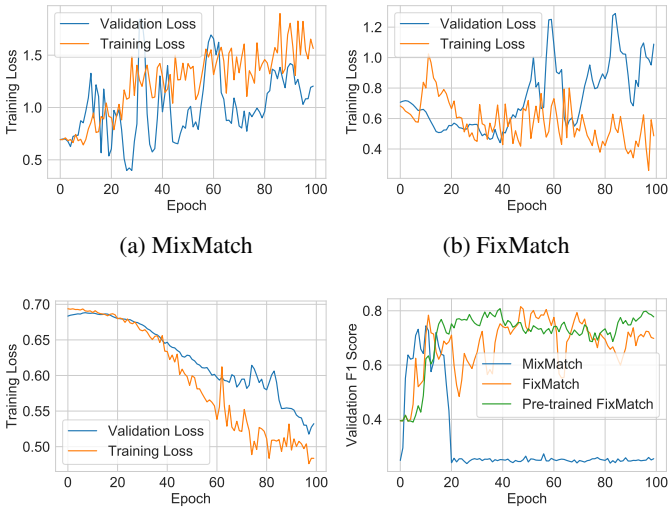
In the previous sub-sections, we ran experiments to find the best configurations for MixMatch, FixMatch, and pre-trained FixMatch. To get the test results for these tuned architectures, we trained them on 3 different seeds to get averages and standard errors.

Looking at Table 7, the overall results are consistent with

the validation results we saw earlier. The semi-supervised algorithm with the best F_1 score was FixMatch with a customised RandAugment, achieving an average of 0.822. MixMatch and pre-trained FixMatch got very similar F_1 scores, however MixMatch was slightly more confident with a lower test loss.

Algorithm	Test F_1 score	Test Loss
MixMatch	0.802 ± 0.010	0.480 ± 0.024
FixMatch	**0.822 ± 0.001**	**0.424 ± 0.065**
Pre-Trained FixMatch	0.803 ± 0.003	0.511 ± 0.006

Table 7. The test results obtained from the best settings of all the algorithms.



(a) MixMatch

(b) FixMatch

(c) Pre-Trained FixMatch

(d) Validation F_1-Score

Figure 7. Training and validation loss for MixMatch, FixMatch and Pre-trained FixMatch. Figure 7d shows the validation F_1 score for the three algorithms.

5. Related work

BreastNet is a DenseNet based neural network architecture that was developed for fully supervised classification on BreaKHis (Toğaçar et al., 2019). The authors publish accuracies in excess of 95% across all data magnifications for binary classification. These results served to motivate some of our early architectural choices when establishing our baseline model as well as demonstrating some upper bounds for state of the art performance on the dataset.

To mitigate the redundancy of many feature maps when applying a generic, pre-trained network (trained on say ImageNet) to a highly specified task such as medical image classification, Ahn et al. propose a fully unsupervised method by placing a convolutional autoencoder layer on top of the pretrained network architecture. This allows one to extract fine tuned, task specific features from an unlabeled image distribution while also leveraging the rich and diverse features contained in the pretrained model. This differs from our method as we do not utilize pretrained network weights, instead we treat our network itself as the encoder layer to a convolutional autoencoder and train randomly initialized weights. The advantage of our technique is that it allows us to have full control of all network parameters when designing our task specific network architecture, instead of relying on off the shelf designs.

Sun et al. provide a three stage semi-supervised learning process applied to breast cancer mammogram data. Their process is similar to ours in that it involves a feature extraction step from the full data distribution, and using high confidence pseudo-labelling to label the unlabeled training data at each training step. The advantages of our technique are that our feature extraction step of pre-training weights using an autoencoder is automatic, while they manually extracted features. Additionally, our data augmentation methods provide important regularization when leveraging the unlabeled data.

6. Conclusions and Future Work

In this work we explored the use of data augmentation to perform semi-supervised classification of benign and malignant breast cancer tumors, using only 5 labeled data points for every subclass in the BeaKHis dataset ultimately achieving a test F_1 score of 0.822 ± 0.001. We began with MixMatch, a semi-supervised model relying on randomly interpolating images to learn from unlabelled data. We determined that MixMatch has a very erratic training process on BreaKHis, which makes analysis and improvements much more complex. We hypothesized that this was due to its many layers of stochasticity, along with some intuitional concerns regarding MixUp applied to this use case. To address these points we implemented FixMatch. After some success in altering the network architecture using increasing dilations and CBAM; we surpassed the MixMatch F_1 score by tailoring the augmentation process to our use case.

Finally, we experimented with pre-training the FixMatch architecture using an autoencoder, so as to leverage features from the entire training distribution for improved pseudo-labeling. Unfortunately, this approach did not yield any global improvements. We did however notice that the stochasticity during training was significantly reduced. This quality is valuable as it is much easier to fine-tune a model with a smoother training curve, and this reduces the variance across different random seeds.

There are many ideas to explore in this project area. One research direction following this work would be to extend our work to the multi-class problem of BreaKHis. Alternatively, it would be interesting to test the pre-training method on a much larger image distribution, or to improve our autoencoder architecture and training objective, e.g. construct a decoder architecture that is more optimized for a DenseNet encoder. Finally, following the work of S4L (Zhai et al., 2019), we could perform further research on having a second stage of fine-tuning the encoder. After finishing the first stage of fine-tuning the encoder on the labeled and unlabeled data, we would then pick the model checkpoint with the best validation score, and perform supervised fine-tuning, using only the labeled data.

References

Ahn, Euijoon, Kumar, Ashnil, Feng, Dagan, Fulham, Michael, and Kim, Jinman. Unsupervised deep transfer feature learning for medical image classification. In *Proceedings - International Symposium on Biomedical Imaging*, volume 2019-April, pp. 1915–1918, 3 2019. ISBN 9781538636411. doi: 10.1109/ISBI.2019.8759275. URL http://arxiv.org/abs/1903.06342.

Antoniou, Antreas, Słowik, Agnieszka, Crowley, Elliot J., and Storkey, Amos. Dilated DenseNets for Relational Reasoning. 2018. URL http://arxiv.org/abs/1811.00410.

Araujo, Teresa, Aresta, Guilherme, Castro, Eduardo, Rouco, Jose, Aguiar, Paulo, Eloy, Catarina, Polonia, Antonio, and Campilho, Aurelio. Classification of breast cancer histology images using convolutional neural networks. *PLoS ONE*, 12(6), 6 2017. ISSN 19326203. doi: 10.1371/journal.pone.0177544.

Bayramoglu, Neslihan, Kannala, Juho, and Heikkila, Janne. Deep learning for magnification independent breast cancer histopathology image classification. In *Proceedings - International Conference on Pattern Recognition*, pp. 2440–2445, 2016. ISBN 9781509048472. doi: 10.1109/ICPR.2016.7900002.

Berthelot, David, Carlini, Nicholas, Goodfellow, Ian, Papernot, Nicolas, Oliver, Avital, and Raffel, Colin. MixMatch: A Holistic Approach to Semi-Supervised Learning. *Advances in Neural Information Processing Systems*, 5 2019. URL http://arxiv.org/abs/1905.02249.

Chen, Min, Shi, Xiaobo, Zhang, Yin, Wu, Di, and Guizani, Mohsen. Deep Features Learning for Medical Image Analysis with Convolutional Autoencoder Neural Network. *IEEE Transactions on Big Data*, pp. 1–1, 6 2017. ISSN 2332-7790. doi: 10.1109/tbdata.2017.2717439.

Cubuk, Ekin D., Zoph, Barret, Shlens, Jonathon, and Le, Quoc V. RandAugment: Practical automated data augmentation with a reduced search space. 2019. URL http://arxiv.org/abs/1909.13719.

de Bruijne, Marleen. Machine learning approaches in medical image analysis: From detection to diagnosis. In *Medical Image Analysis*, volume 33, pp. 94–97. Elsevier B.V., 10 2016. doi: 10.1016/j.media.2016.06.032.

Gurcan, Metin N., Boucheron, Laura E., Can, Ali, Madabhushi, Anant, Rajpoot, Nasir M., and Yener, Bulent. Histopathological Image Analysis: A Review. *IEEE Reviews in Biomedical Engineering*, 2:147–171, 2009. ISSN 19411189. doi: 10.1109/RBME.2009.2034865.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, and Sun, Jian. Deep residual learning for image recognition. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2016-Decem, pp. 770–778, 12 2016. ISBN 9781467388504. doi: 10.1109/CVPR.2016.90. URL https://arxiv.org/abs/1512.03385.

Hu, Jie, Shen, Li, and Sun, Gang. Squeeze-and-Excitation Networks. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 9 2018. ISBN 9781538664209. doi: 10.1109/CVPR.2018.00745. URL http://arxiv.org/abs/1709.01507.

Huang, Gao, Liu, Zhuang, Van Der Maaten, Laurens, and Weinberger, Kilian Q. Densely connected convolutional networks. In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, volume 2017-Janua, pp. 2261–2269, 8 2017. ISBN 9781538604571. doi: 10.1109/CVPR.2017.243. URL http://arxiv.org/abs/1608.06993.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *32nd International Conference on Machine Learning, ICML 2015*, volume 1, pp. 448–456, 2015. ISBN 9781510810587.

Jiang, Yun, Chen, Li, Zhang, Hai, and Xiao, Xiao. Breast cancer histopathological image classification using convolutional neural networks with small SE-ResNet module. *PLoS ONE*, 14(3), 3 2019. ISSN 19326203. doi: 10.1371/journal.pone.0214587.

Kingma, Diederik P. and Ba, Jimmy Lei. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 2015.

Lee, Dong-Hyun. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. *ICML 2013 Workshop: Challenges in Representation Learning*, pp. 1–6, 2013. URL https://www.kaggle.com/blobs/download/forum-message-attachment-files/746/pseudo_label_final.pdf.

Litjens, Geert, Kooi, Thijs, Bejnordi, Babak Ehteshami, Setio, Arnaud Arindra Adiyoso, Ciompi, Francesco, Ghafoorian, Mohsen, van der Laak, Jeroen A.W.M., van Ginneken, Bram, and Sánchez, Clara I. A survey on deep learning in medical image analysis, 12 2017. ISSN 13618423.

Loshchilov, Ilya and Hutter, Frank. SGDR: Stochastic gradient descent with warm restarts. In *5th International Conference on Learning Representations, ICLR 2017 - Conference Track Proceedings*, 2017. URL https://github.com/loshchil/SGDR.

Nawaz, Majid, Sewissy, Adel A, and Soliman, Hassan A. Multi-Class Breast Cancer Classification using Deep Learning Convolutional Neural Network. Technical Report 6, 2018. URL www.ijacsa.thesai.org.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. Learning Internal Representations by Error Propagation. In *Readings in Cognitive Science: A Perspective from Psychology and Artificial Intelligence*, pp. 399–421. MIT Press, 2013. ISBN 1558600132. doi: 10.1016/B978-1-4832-1446-7.50035-2.

Sohn, Kihyuk, Berthelot, David, Li, Chun-Liang, Zhang, Zizhao, Carlini, Nicholas, Cubuk, Ekin D., Kurakin, Alex, Zhang, Han, and Raffel, Colin. FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. 1 2020. URL http://arxiv.org/abs/2001.07685.

Spanhol, Fabio A., Oliveira, Luiz S., Petitjean, Caroline, and Heutte, Laurent. A Dataset for Breast Cancer Histopathological Image Classification. *IEEE Transactions on Biomedical Engineering*, 63(7):1455–1462, 7 2016. ISSN 15582531. doi: 10.1109/TBME.2015.2496264.

Srivastava, Nitish, Hinton, Geoffrey, Krizhevsky, Alex, Sutskever, Ilya, and Salakhutdinov, Ruslan. Dropout: A simple way to prevent neural networks from overfitting. In *Journal of Machine Learning Research*, volume 15, pp. 1929–1958, 2014.

Sun, Wenqing, Tseng, Tzu Liang Bill, Zhang, Jianying, and Qian, Wei. Computerized breast cancer analysis system using three stage semi-supervised learning method. *Computer Methods and Programs in Biomedicine*, 135:77–88, 2016. ISSN 18727565. doi: 10.1016/j.cmpb.2016.07.017.

Toğaçar, Mesut, Özkurt, Kutsal Baran, Ergen, Burhan, and Cömert, Zafer. BreastNet: A novel convolutional neural network model through histopathological images for the diagnosis of breast cancer. *Physica A: Statistical Mechanics and its Applications*, 2019. ISSN 03784371. doi: 10.1016/j.physa.2019.123592.

Weese, Jürgen and Lorenz, Cristian. Four challenges in medical image analysis from an industrial perspective. In *Medical Image Analysis*, volume 33, pp. 1339–1351. Elsevier B.V., 10 2016. doi: 10.1016/j.media.2016.06.023. URL http://www.ncbi.nlm.nih.gov/pubmed/27344939.

Woo, Sanghyun, Park, Jongchan, Lee, Joon Young, and Kweon, In So. CBAM: Convolutional block attention module. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 11211 LNCS, pp. 3–19, 2018. ISBN 9783030012335. doi: 10.1007/978-3-030-01234-2{_}1.

Zhai, Xiaohua, Oliver, Avital, Kolesnikov, Alexander, and Beyer, Lucas. S4L: Self-Supervised Semi-Supervised Learning. *Proceedings of the IEEE international conference on computer vision*, 2019. URL http://arxiv.org/abs/1905.03670.

Zhang, Hongyi, Cisse, Moustapha, Dauphin, Yann N., and Lopez-Paz, David. MixUp: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings*, 10 2018. URL http://arxiv.org/abs/1710.09412.