Investigating the effect of novel classes in semi-supervised learning

Alex Yuxuan Peng YPEN260@AUCKLANDUNI.AC.NZ
Yun Sing Koh YKOH@CS.AUCKLAND.AC.NZ
Patricia Riddle PAT@CS.AUCKLAND.AC.NZ
University of Auckland

Bernhard Pfahringer BERNHARD@CS.WAIKATO.AC.NZ
University of Waikato

Editors: Wee Sun Lee and Taiji Suzuki

Abstract

Semi-supervised learning usually assumes the distribution of the unlabelled data to be the same as that of the labelled data. This assumption does not always hold in practice. We empirically show that unlabelled data containing novel examples and classes from outside the distribution of the labelled data can lead to a performance degradation for semi-supervised learning algorithms. We propose a 1-nearest-neighbour based method to assign a weight to each unlabelled example in order to reduce the negative effect of novel classes in unlabelled data. Experimental results on MNIST, Fashion-MNIST and CIFAR-10 datasets suggest that the negative effect of novel classes becomes statistically insignificant when the proposed method is applied. Using our proposed technique, models trained on unlabelled data with novel classes can achieve similar performance as ones trained on clean unlabelled data.

Keywords: semi-supervised learning, neural network, novel classes, generalization

1. Introduction

Neural networks have been successfully applied to challenging tasks such as image or speech recognition. However, in order to achieve good performance, it is necessary to have a large amount of labelled training data. This requires humans to painstakingly label many examples. The labelling process can be very costly and time consuming. To address this issue, many different semi-supervised learning (SSL) algorithms have been proposed in recent years [Kingma et al. (2014); Laine and Aila (2017); Lee (2013); Rasmus et al. (2015); Tarvainen and Valpola (2017)]. The reported results on some of the benchmark datasets for semi-supervised learning using only very few labelled examples are approaching the performance of supervised learning with all the examples labelled. For instance, Tarvainen and Valpola (2017) managed to achieve an error rate of 9.11% on ImageNet 2012 with only 10% of the instances being labelled [Tarvainen and Valpola (2017)].

Most research on semi-supervised learning assumes that the distribution of unlabelled data is the same as the distribution of labelled data. However, this assumption does not always hold in practice. We use semi-supervised learning when we do not have the resources to label all of the available data. This means that we cannot check all the data points to make sure they are only from the classes we are interested in. Therefore novel classes might be present in the unlabelled data when we apply semi-supervised learning in practice. We define **novel classes** as classes that are only present in the unlabelled training data, but not in the labelled data (including test data). We assume future unseen data will always be from the same distribution as the labelled training data. Hence, our test datasets only contain classes that are present in the labelled training data. Note that our learning scenario is different to that of novelty detection [Bishop (1994)] and domain shift or transfer [Ruder and Plank (2018)], where the future unseen data can be distributed differently from the training data.

We attempt to answer two research questions. The **first research question** is: *Do novel classes in unlabelled data negatively affect the performance of semi-supervised learning algorithms?* The **second research question** is: *Can we reduce this negative effect of novel classes by assigning individual weights to unlabelled data?* Our experiments show that the presence of novel classes does degrade the performance of semi-supervised algorithms. In order to answer the second question, we propose a 1-nearest-neighbour based method to assign individual weights to unlabelled data. The experimental results suggest that when the proposed method is applied to semi-supervised algorithms, the negative effect of novel classes becomes statistically insignificant.

The rest of the paper is organized as follows. In Section 2, we review related research. In Section 3, we introduce two semi-supervised learning algorithms that are used in our experiments. We propose a method to reduce the effect of novel classes in Section 4. In Section 5, we describe the experiments designed to address the two research questions. We analyze the experimental results in Section 6. We give further discussions about our experiments as well as drawbacks of our proposed method in Section 7. Finally, we conclude our work and propose future research in Section 8.

2. Related Work

There has been a lot of development in semi-supervised learning for neural networks. Pseudo-Label is a simple method that assigns "pseudo-labels" to unlabelled data using the model predictions at each epoch, and then a model is trained on both labelled data and the "pseudo-labelled" data [Lee (2013)]. Kingma et al. (2014) proposed using deep generative models for semi-supervised learning [Kingma et al. (2014)]. They used variational autoencoders [Kingma and Welling (2014)] as their generative models. Inspired by Denoising Autoencoders [Vincent et al. (2010)], Ladder Network [Rasmus et al. (2015)] adds noise to the input of each layer and proposed a denoising function to reconstruct the input. Then the denoising loss is added to the supervised loss. Instead of adding random noise to the input, Virtual Adversarial Training attempts to compute tiny perturbations that are expected to affect the predictions the most by using the idea of adversarial training [Miyato et al. (2016, 2017)]. Temporal Ensemble [Laine and Aila (2017)] is a self-training based method that is similar to Pseudo-Label. However, instead of training the current model to match the predictions of the previous model, it encourages the current model to match the outputs of the previous model directly. Temporal Ensemble also uses an exponential moving average of the model outputs as the "target". This is more stable than simply using the outputs of the model in the last epoch. The problem with Temporal Ensemble is that the storage or memory required to store the exponential moving average values grows linearly with the size of the training data. Mean Teacher [Tarvainen and Valpola (2017)] solves this problem by keeping an exponential moving average of the models instead of model outputs. Now the memory requirements do not change with the size of the data. We limit the scope of this paper to self-training based methods such as Pseudo-Label and Mean Teacher. We leave the experimentation with other types of semi-supervised learning to future research.

Oliver et al. (2018) evaluated a few algorithms mentioned above in fair and realistic settings [Oliver et al. (2018)]. The implementations of all the algorithms evaluated used the same architectures, data augmentations, and optimization methods. They showed that the supervised baselines used in some of the previous works were too weak. In their implementation, the gap between semi-supervised training and the supervised baseline was much smaller. They also studied the effect of distribution mismatch between the labelled data and unlabelled data. They found that adding unlabelled data from the novel classes can hurt the performance. This is in agreement with our experimental results. While Oliver et al. (2018) only tested the effect of novel classes on CIFAR-10, we also conducted experiments on MNIST and Fashion-MNIST. In addition to demonstrating the negative effect of novel classes, we also propose the first novel solution for reducing this effect.

3. Background

We use Pseudo-Label and Mean Teacher as two example algorithms in our experiments to investigate the effect of novel classes in semi-supervised learning for neural networks. In this section, we describe how both Pseudo-Label and Mean Teacher work.

3.1. Pseudo-Label

Pseudo-Label is a self-training based semi-supervised learning method [Lee (2013)]. *Pseudo-labels* are defined as predictions made by the current model for unlabeled data. The Pseudo-Label method treats these *pseudo-labels* as true labels for the unlabelled data during training. Pseudo-labels are updated after each epoch. So the training process is similar to that of supervised training, except that pseudo-labels are used for the unlabelled data instead of true labels. Cross-entropy loss is usually used for classification problems in supervised learning. Let us define cross-entropy loss as

$$H(y, f) = -\sum_{c=1}^{C} y_c \log f_c \tag{1}$$

where y is the class label encoded in one-hot encoding, f is the model output and C is the number of classes. Then the loss function for Pseudo-Label is defined as

$$L = \frac{1}{m}\sum_{i=1}^{m} H(y_i, f_i) + a(t)\frac{1}{m'}\sum_{j=1}^{m'} H(y'_j, f_j) \tag{2}$$

where m is the number of labelled examples and m' is the number of unlabelled examples. The class label for the ith labelled example is denoted by y_i while y'_j denotes the *pseudo-label* for the jth unlabelled example. Model outputs for the labelled and unlabelled examples are denoted by f_i and f_j respectively. The first term in Equation 2 corresponds to the cross-entropy loss for the labelled data, and the second term is the cross-entropy loss for the unlabelled data. The importance of the loss related to unlabelled data is controlled by a hyperparameter $a(t)$. Intuitively speaking, the second term of the loss function tries to reduce the difference between the predictions of the current model and that of the model in the previous epoch on unlabelled data. The scheduling of the hyperparameter $a(t)$ is very important when applying Pseudo-Label. If $a(t)$ is set too high in the beginning, the model will fail to learn because the model is likely to be very bad at prediction initially. We use a scheduling function to schedule $a(t)$ as defined in Lee (2013).

$$a(t) = \begin{cases} 0 & \text{if } t < T_1 \\ \frac{t-T_1}{T_2-T_1}a & \text{if } T_1 \leq t < T_2 \\ a & \text{if } T_2 \leq t \end{cases}$$

where a, T_1 and T_2 are set by users, and t is the current training epoch. So $a(t)$ is initially set to be 0, and gradually increased after each epoch before being set to be the user selected a for the remaining epochs. Lee (2013) showed that Pseudo-Label has an effect of entropy regularization on MNIST dataset, which provides an explanation for the success of Pseudo-Label.

3.2. Mean Teacher

Mean Teacher is also a self-training based semi-supervised method. However, instead of trying to reduce the difference in predictions between the current model and the previous model, Mean Teacher attempts to reduce the difference in model outputs. Furthermore, Mean Teacher keeps an exponential moving average (EMA) of the model that is considered as the teacher model. Mean Teacher then tries to reduce the difference between the student model (current model) and the teacher model. The rationale for using a EMA model as the teacher model is that the EMA model provides more stable "target" outputs. The loss function of Mean Teacher is defined as

$$L = \frac{1}{m}\sum_{i=1}^{m} H(y_i, f_i) + a(t)\frac{1}{2(m+m')}\sum_{j=1}^{m+m'}(f_j - f'_j)^2 \tag{3}$$

where f is the output of the student model, and f' is the output of the teacher model (the exponential moving average of the student model). The first term in the loss function is the cross-entropy loss of the labelled data. The second term is the mean squared error (MSE) that measures the distance between the outputs of the student model and the teacher model. Again, a hyperparameter $a(t)$ is used to balance the two losses. Mean Teacher uses a Gaussian ramp-up function to schedule hyperparameter $a(t)$ as defined in Tarvainen and Valpola (2017) and Laine and Aila (2017). It is defined as follows.

$$a(t) = \begin{cases} a * exp[-5(1-\frac{t}{T})^2] & \text{if } t \leq T \\ a & \text{if } t > T \end{cases}$$

where t is the current epoch, T is the ramp-up period chosen by user, and a is the base parameter also set by user. We applied the same ramp-up function in our implementation of Mean Teacher. Apart from a, Mean Teacher introduced another hyperparameter, the EMA decay rate η, that controls the coefficient λ in EMA.

$$\lambda(s) = min(1 - \frac{1}{s+1}, \eta)$$

where s is the training step (number of weight updates) and η is chosen by user. $\lambda(s)$ controls the importance of the EMA model relative to the current model when updating the teacher model. Note that in Mean Teacher, the teacher model is updated for each training step instead of each epoch. The above function indicates that we use the true average of the models learned so far as the teacher model in the beginning, before we start using the EMA model.

4. Distance Based Weighting Framework

As described in Section 3, Pseudo-Label and Mean Teacher attempt to train the current model (student model) to match the predictions (outputs) of the model in the last epoch (teacher model) for unlabelled data. The existence of novel classes in the unlabelled data can potentially push the decision boundary away from the one learned from using clean unlabelled data. The experimental results in Section 5.1 suggest that the presence of novel classes can hurt the performance of the algorithms. In this section, we propose a distance based weighting framework and our implementation of it to reduce the negative effect of novel classes.

We attempt to reduce the negative effect of novel classes by assigning individual weights to unlabelled data. The intuition is very simple, if we can lower the weights of the novel examples in the training loss, these novel examples will have less impact on the training. Our proposed framework works as follows.

1. Compute the distance d_j for each unlabelled data point to the known classes.
2. Apply a function $g(d_j)$ to transform distance d_j into weight w_j.
3. Assign weight w_j to each unlabelled data point in the loss function.

The specific implementation of this framework depends on the definitions of distance d_j and transformation function $g(d_j)$. The distance to the known classes should be higher for an example from the novel classes. The transformation function should transform high-distance values into low-weight values, and ideally be bounded to a certain range. Next, we describe our implementation of this framework.

4.1. 1-Nearest-Neighbour Based Weighting

We define the distance d_j as the Euclidean distance between an unlabelled example and its closest neighbour in the labelled training data. Hence, we essentially run a 1-Nearest-Neighbour (1NN) algorithm with the labelled data being the training set, and compute the 1NN distance d_j for each unlabelled example. Note that we basically have n clusters, where

Table 1: Architecture A. The architecture used for MNIST and Fashion-MNIST.

Layer	**Parameters**
Input	28 × 28 grayscale images
Convolutional	20 filters, 5 × 5, valid padding, relu
Pooling	Maxpool 2 × 2
Convolutional	50 filters, 5 × 5, valid padding, relu
Pooling	Maxpool 2 × 2
Dense	800 → 500, relu
Softmax	500 → 5

n is the number of labelled examples in the training set. The centroid for each cluster is one of the labelled examples. We then need to define a function $g(d_j)$ to transform all the distances into weights.

A naive way to transform distance into a weight can be defined as follows.

$$g(d_j) = 1 - scale(d_j)$$

where *scale* normalizes d_j to the range [0, 1] within its cluster. This transformation is unlikely to work well. Because it always assigns 0 to the unlabelled example that is furthest away from its centroid, regardless of the value of the distance. However, it is possible that every unlabelled example is very close to the centroid (labelled example) and belongs to the same class, hence all of the unlabelled examples should be assigned large weights. We use the following transformation instead:

$$g(d_j) = \frac{1}{exp(\beta d_j)} \tag{4}$$

where β is a hyperparameter that controls how quickly the weight approaches zero as the distance increases. Using this transformation we can avoid the problem described above.

Lastly, we apply the computed weights to the loss function. For instance, the loss function for Pseudo-Label now becomes

$$L = \frac{1}{m}\sum_{i=1}^{m} H(y_i, f_i) + a(t)\frac{1}{m'}\sum_{j=1}^{m'} w_j H(y'_j, f_j) \tag{5}$$

where w_j is weight for the jth unlabelled example. And the loss function for Mean-Teacher is now

$$L = \frac{1}{m}\sum_{i=1}^{m} H(y_i, f_i) + a(t)\frac{1}{2(m+m')}\sum_{j=1}^{m+m'} w_j(f_j - f'_j)^2 \tag{6}$$

where w_j is the weight for the jth example. The experiments designed to evaluate the proposed method are explained in Section 5.2, and the experimental results are analysed in Section 6.2.

5. Experiments

In this section, we describe the experiments that investigate the effect of novel classes on Pseudo-Label and Mean Teacher, and test the effectiveness of the proposed method in reducing the effect of novel classes. All the experiments were implemented in Pytorch [Paszke et al. (2017)][1]. The experimental results are discussed in Section 6.

Table 2: Architecture B. The architecture used for CIFAR-10.

Layer	**Parameters**
Input	32×32 RGB images
Convolutional	32 filters, 3×3, same padding, relu
Convolutional	32 filters, 3×3, valid padding, relu
Pooling	Maxpool 2×2
Convolutional	64 filters, 3×3, same padding, relu
Convolutional	64 filters, 3×3, valid padding, relu
Pooling	Maxpool 2×2
Dense	$2304 \rightarrow 512$, relu
Softmax	$512 \rightarrow 5$

5.1. Experiments to Demonstrate the Effect of Novel Classes

We conducted experiments on three popular image-recognition datasets: MNIST [LeCun et al. (1998)], Fashion-MNIST [Xiao et al. (2017)] and CIFAR-10 [Krizhevsky and Hinton (2009)]. The characteristics of these datasets and the preprocessing are explained below.

- **MNIST** is a 28×28 grayscale image dataset of handwritten digits. The dataset contains 60,000 training images and 10,000 test images. There are 10 classes in total (0 to 9). The classes are distributed evenly. In our experiments, a validation set containing 5,000 images is created from the training images using stratified sampling. This leaves 55,000 images for training. We treat classes $\{5, 6, 7, 8, 9\}$ as novel classes. This means the labelled training set, validation set and test set only contain classes $\{0, 1, 2, 3, 4\}$, while the unlabelled data includes all classes. The validation set and test set are around half of their original sizes after taking out the novel classes. The number of labelled examples in the labelled training set is set to 50.

- **Fashion-MNIST** follows the same format and structure as MNIST, but it contains fashion items instead of handwritten digits. The ten classes are t-shirt/top, trousers, pullover, dress, coat, sandal, shirt, sneaker, bag and ankle boot. Again, we created a validation set containing 5,000 examples (including novel classes) from the 60,000 training images. Classes {sandal, shirt, sneaker, bag, ankle boot} are considered as novel classes. The number of labelled examples in the labelled training set is set to 100.

1. The source code can be found on GitHub: https://github.com/superRookie007/novel-classes-ssl. Supplementary materials and additional experimental results are also published there.

- **CIFAR-10** is a 32×32 RBG image dataset. There are 50,000 training images and 10,000 test images. There are 10 balanced classes: airplane, automobile, bird, cat, deer, dog, frog, horse, ship and truck. We split the training images into a validation set of 5,000 examples (including novel classes) and a training set of 45,000 examples. Classes {dog, frog, horse, ship, truck} are treated as novel classes. The labelled training set has 3000 examples.

An unlabelled dataset is **pure** if it does not contain any example from the novel classes. A **dirty** unlabelled dataset includes all the examples from the novel classes. For each dataset, we experimented with three training scenarios as follows.

- *Exclude*: supervised training without unlabelled data.
- *Include_pure*: semi-supervised training with pure unlabelled data.
- *Include_dirty*: semi-supervised training with dirty unlabelled data.

Each experiment was run 50 times for MNIST and Fashion-MNIST, and 20 times for CIFAR-10. The seed for splitting the training data into labelled and unlabelled data was different in each run. The seeds were kept the same for each of the three training scenarios.

Both Pseudo-Label and Mean Teacher are model or architecture agnostic. This means that we can use whatever architecture we want with these algorithms without changing the rest of the code. We used Architecture A defined in Table 1 for MNIST and Fashion-MNIST. Architecture B defined in Table 2 is used for CIFAR-10. We use the standard mini-batch stochastic gradient descent (SGD) to train our models. We use a batch size of 100 for all our experiments. The following learning rate scheduler is used:

$$lr(t) = lr \times \gamma^t$$

where lr is the base learning rate set by the user, t is the current epoch and γ is the multiplicative factor of the learning rate decay process. All hyperparameters were set using validation sets. The specific hyperparameters used in each experiment can be found in the supplementary materials. Experimental results are discussed in Section 6.1. All results were obtained from the test sets.

Note that the architectures used in this paper are different from the ones used in the original papers. We used simple architectures to limit the computing power and running time for our experiments. It is possible that different architectures can potentially change how novel classes affect these algorithms. However, this is a different research question and we will leave it to future research. Lee (2013) applied unsupervised pretraining in addition to the Pseudo-Label algorithm in order to improve the performance further. Data augmentation was applied in the original implementation of Mean Teacher [Tarvainen and Valpola (2017)]. We do not apply any data augmentation or unsupervised pretraining in our implementation, because our goal is not to achieve state-of-the-art results on benchmark datasets. The additional complexity only makes it more difficult to interpret the experimental results. We only intend to test if novel classes have any negative effect on the performance of semi-supervised algorithms, while holding other factors constant.

5.2. Evaluation of the Proposed Method

We evaluated the proposed method on both Pseudo-Label and Mean Teacher on MNIST [LeCun et al. (1998)], Fashion-MNIST [Xiao et al. (2017)] and CIFAR-10 [Krizhevsky and Hinton (2009)]. The data processing is exactly the same as described in Section 5.1. The only difference is that we have to compute the weight for each unlabelled data point using the proposed method. And we now have an additional hyperparameter β to tune.

Note that when we apply the 1-Nearest-Neighbour algorithm, we can use raw images or low-dimensional vector representations of the images. Since distance based methods suffer from *the curse of dimensionality*, we postulate that we can get better performance if dimensionality reduction is applied beforehand. We tested using Variational Autoencoder (VAE) [Kingma and Welling (2014)] to learn low-dimensional representations from the images, and then applied the 1-Nearest-Neighbour to these representations. For MNIST and Fashion-MNIST, we used a simple multilayer perceptron (MLP) based encoder and decoder as introduced in Kingma and Welling (2014). The dimensionality of the hidden representation was set to 10. For CIFAR-10, we used convolutional encoder and transposed convolution for up-sampling in the decoder. The dimensionality of the vector representation for CIFAR-10 was set to 20.

To test the effectiveness of the proposed method, we apply the computed weights in Pseudo-Label and Mean Teacher, and check if this improves the performance on include_dirty where novel classes are present in unlabelled data. We add two additional training scenarios to our experiments as follows.

- *With_weights_raw*: weights are computed using raw images, and model is trained with dirty unlabelled data.
- *With_weights_vae*: weights are computed using low-dimensional representations learned using VAE, and model is trained with dirty unlabelled data.

Both with_weights_raw and with_weights_vae were trained using the same dirty unlabelled data on which include_dirty was trained. Similar to the experiments in Section 5.1, we ran each experiment 50 times for MNIST and Fashion-MNIST, and 20 times for CIFAR-10. The data seed was different for each run. We used the same set of seeds across all experiments. Again, all hyperparameters were set using validation sets, the hyperparameters used in each experiment can be found in the supplementary materials. Experimental results are reported in Section 6.2. All results reported were obtained from the test sets.

6. Experimental Results

In this section, we discuss the experimental results obtained from the experiments described in the last section. Test accuracy results for Pseudo-Label and Mean Teacher are reported in Tables 3 and 4 respectively. Since the differences in test accuracies across different settings are small, we provide more detailed analysis of the results using the *Friedman test* and *Nemenyi post-hoc test* in the next two subsections. Demšar (2006) provides a great discussion on Friedman and Nemenyi tests along with other statistical tests for comparing classifiers.

Table 3: All the experimental results in the paper for Pseudo-Label. It shows the mean and standard deviation of test accuracy for each dataset and experiment setting.

	MNIST	Fashion-MNIST	CIFAR-10
exclude	93.63% ±1.51%	77.15% ±1.56%	63.85% ±1.20%
include_pure	94.52% ±1.71%	77.02% ±1.99%	65.37% ±1.19%
include_dirty	92.15% ±2.27%	76.14% ±2.02%	64.45% ±1.11%
with_weights_raw	92.74% ±2.11%	77.01% ±1.55%	65.14% ±1.01%
with_weights_vae	93.79% ±1.39%	76.86% ±1.76%	65.88% ±1.22%

Table 4: All the experimental results in the paper for Mean Teacher. It shows the mean and standard deviation of test accuracy for each dataset and experiment setting.

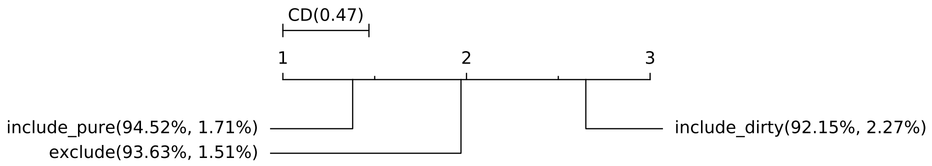
	MNIST	Fashion-MNIST	CIFAR-10
exclude	93.63% ±1.51%	77.15% ±1.56%	63.85% ±1.20%
include_pure	94.06% ±1.68%	78.58% ±1.93%	66.56% ±0.96%
include_dirty	92.97% ±2.03%	76.95% ±2.04%	64.94% ±1.32%
with_weights_raw	93.11% ±1.99%	77.19% ±1.83%	66.04% ±1.11%
with_weights_vae	93.42% ±1.46%	77.14% ±1.85%	66.24% ±0.84%

6.1. Effect of Novel Classes

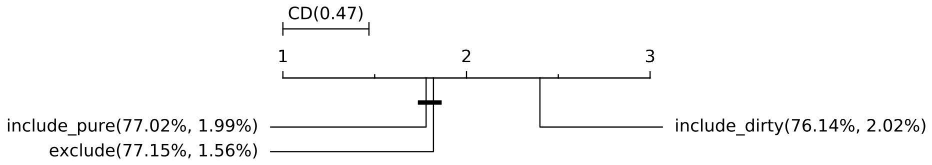
To find out if novel classes really affect the performance, we used the Friedman test ($p = 0.05$) to test if the test accuracies under the three different settings (include_pure, include_dirty and exclude) are statistically different for each dataset. The details of different settings can be found in Section 5. The test suggested that the distributions of test accuracies are indeed different. Then we performed the Nemenyi post-hoc test for pairwise comparisons. We used 0.05 as the confidence level. The results for the Nemenyi tests are shown as CD graphs in Figure 1 for Pseudo-Label and Figure 2 for Mean Teacher. The CD graph shows the average ranking of each setting. The lower the ranking the better. The difference in average ranking is statistically significant if there is no bold line connecting the two settings. The mean and standard deviation of test accuracy are shown in parenthesis.

Figure 1 shows that include_dirty is statistically significantly worse than exclude and include_pure on both MNIST and Fashion-MNIST for Pseduo-Label. For CIFAR-10, there is no significant difference between include_dirty and exclude, but include_dirty is still significantly worse than include_pure. The results for Mean Teacher are shown in Figure 2. For all three datasets, include_dirty is not significantly different from exclude, but it is significantly worse than include_pure.

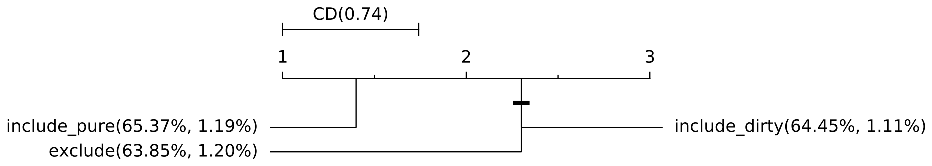
Overall, we can clearly see that novel classes in unlabelled data have a negative effect on the performance of Pseudo-Label and Mean Teacher. When novel classes are present in the unlabelled data, the performance of these algorithms is significantly lower than that trained using clean unlabelled data without novel classes.



(a) Pseudo-Label on MNIST



(b) Pseudo-Label on Fashion-MNIST



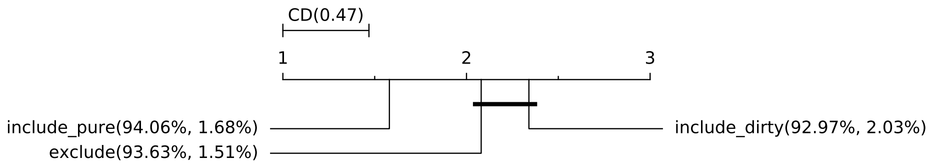
(c) Pseudo-Label on CIFAR-10

Figure 1: Comparison of supervised training without unlabelled data, Pseudo-Label with clean unlabelled data and Pseudo-Label with dirty unlabelled data on MNIST (a), Fashion-MNIST(b) and CIFAR-10 (c).

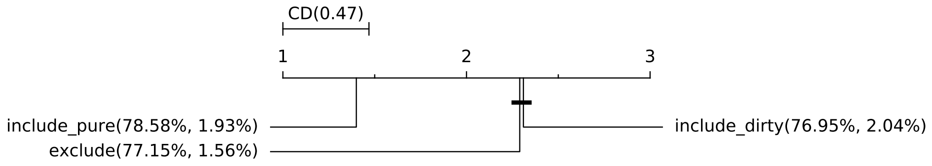
6.2. Effectiveness of 1NN Based Weighting

We have shown that the presence of novel classes lower the performance of Pseudo-Label and Mean Teacher, we now test if our proposed weighting method can reduce this negative effect. Figure 3 shows the Nemenyi test results for Pseudo-Label. For all three datasets, with_weights_vae consistently outperformed include_dirty. This means that when the low-dimensional representations are used for weight computation, our proposed method improved the performance of Pseudo-Label when novel classes are present. When raw images were used to compute the weights, our proposed method did improve the average ranking, but the improvement is not statistically significant for all datasets. There is no significant difference between with_weights_vae and include_pure for all three datasets. This suggests that with our proposed weighting method, the performance of Pseudo-Label does not suffer significantly when novel classes are present in unlabelled data.

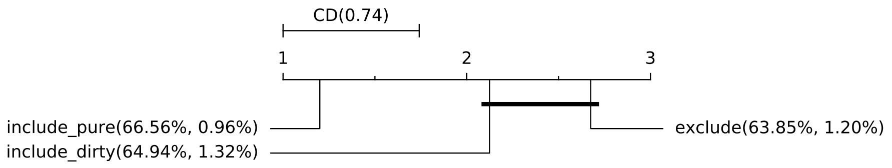
The experimental results for Mean Teacher are shown in Figure 4. For MNIST, with_weights_vae is not significantly better than include_dirty, but is statistically the same as include_pure. Include_pure is significantly better than all the other three settings on Fashion-MNIST. Include_dirty performed significantly worse than the other settings on CIFAR-10 and include_pure is statistically the same as with_weights_vae and with_weights_raw.



(a) Mean Teacher on MNIST



(b) Mean Teacher on Fashion-MNIST



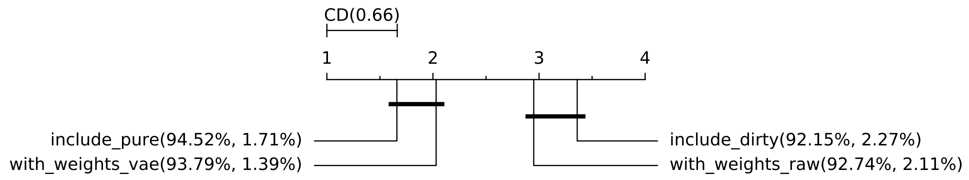
(c) Mean Teacher on CIFAR-10

Figure 2: Comparison of supervised training without unlabelled data, Mean Teacher with clean unlabelled data and Mean Teacher with dirty unlabelled data on MNIST (a), Fashion-MNIST(b) and CIFAR-10 (c).

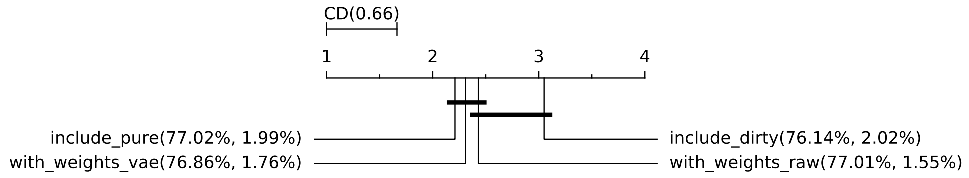
Overall, when our proposed method is applied to Pseudo-Label and Mean Teacher, the performance does not suffer as much when novel classes are present. There is actually no significant difference in performance regardless of the presence of novel classes except for one out of the 6 cases, when our proposed method is used. The advantage of using low-dimensional representations over raw images to compute the weights is not statistically significant, according to our experimental results.

7. Discussion

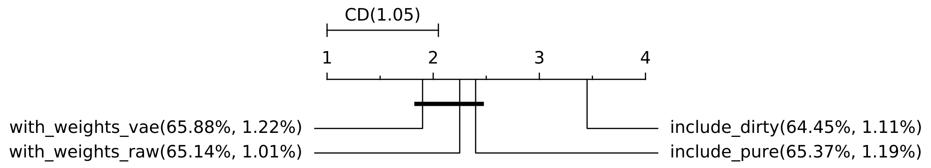
The experimental results in Section 6.1 suggest that novel classes in unlabelled data can hurt the performance of semi-supervised learning algorithms. This is in agreement with the findings by Oliver et al. (2018). They did not re-tune the hyperparameters for experiments that include novel classes, however, we did tune the a hyperparameter while holding other parameters unchanged. The hyperparameter a controls the importance of the regulating term in the loss function. It is essentially a universal weight applied to all the unlabelled data. If we lower a, apart from reducing the negative effect of novel classes, it will also lower the positive effect of the rest of the unlabelled data. Figure 5 shows the comparison of Mean Teacher models with different a values (holding other hyperparameters constant) trained on CIFAR-10 with novel classes. The a value of 50 was used in our experiments



(*a*) Pseudo-Label on MNIST



(*b*) Pseudo-Label on Fashion-MNIST

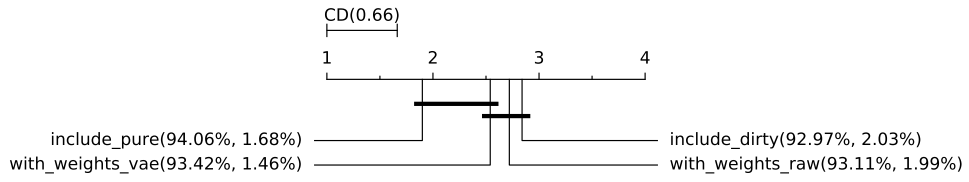


(*c*) Pseudo-Label on CIFAR-10

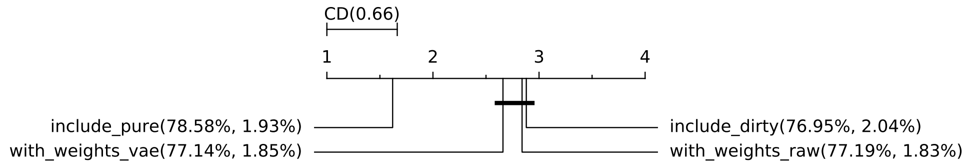
Figure 3: Comparison of Pseudo-Label models trained in the following conditions: dirty unlabelled data without weights, dirty unlabelled data with weights computed from raw images, dirty unlabelled data with weights computed from low-dimensional representations and clean unlabelled data without weights.

in Section 6.1. Lowering the hyperparameter a did not improve the performance of Mean Teacher. Hence, we need to assign individual weights to each unlabelled example to keep the benefit of the clean unlabelled data and at the same time weaken the impact of the novel classes.

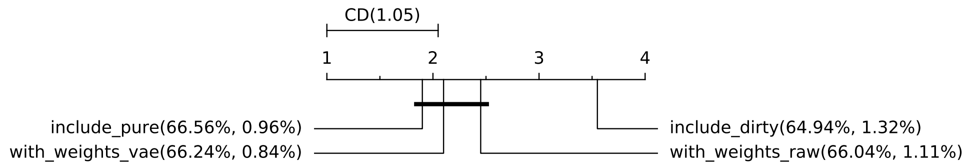
According to the experimental results in Section 6.2, the advantage provided by learning low-dimensional representations of images when computing weights over using raw images is not significant. It could be due to the hyperparameter tuning or the choice of architectures. The advantage can potentially be bigger when more advanced architectures are used in training the VAE. It would be interesting to explore other unsupervised learning algorithms to learn the latent representations. A major downside of using raw images to compute weights and distances is the running time due to the big dimensionality of raw images. This is an obstacle if we have to compute the weights frequently. Learning latent representation by training an autoencoder takes time. However, this training time can be amortized since we can reuse the encoder every time we compute weights. We only tested the proposed method on Pseudo-Label and Mean Teacher, however, it can also be applied to other semi-supervised learning algorithms.



(a) Mean Teacher on MNIST



(b) Mean Teacher on Fashion-MNIST



(c) Mean Teacher on CIFAR-10

Figure 4: Comparison of Mean Teacher models trained in the following conditions: dirty unlabelled data without weights, dirty unlabelled data with weights computed from raw images, dirty unlabelled data with weights computed from low-dimensional representations and clean unlabelled data without weights.

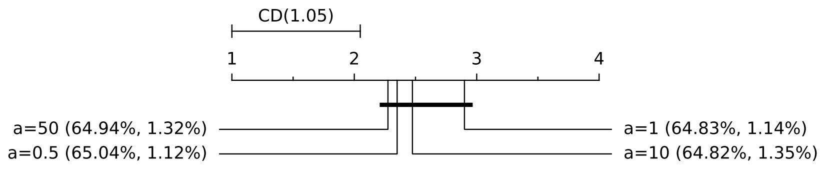


Figure 5: Mean Teacher models with different a values were trained on CIFAR-10 with dirty unlabelled data. We used a value of 50 in our experiments in Section 5.1.

Lastly, we introduced a hyperparameter β in our proposed method. A validation set is needed to tune β along with other hyperparameters in semi-supervised algorithms. In order to be confident that the tuned hyperparameters will work well on future unseen data, the validation set has to be sufficiently large. We only use semi-supervised algorithms when labelled data is scarce. It is possible that we will have larger gain in performance by using most of the labelled data as training data instead of validation set. However, in most of the literature on semi-supervised learning the validation set used is usually larger than the labelled training set. This problem has been mentioned in Oliver et al. (2018).

8. Conclusion

Even though it is a common practice to assume unlabelled data has the same distribution as the labelled data in semi-supervised learning literature, this assumption is not always true. In this work, we empirically showed that novel classes in unlabelled data can hurt the performance of semi-supervised learning algorithms. We then proposed a 1-nearest-neighbour based method to assign weights to unlabelled data, in order to reduce the negative effect of novel classes. The experimental results show that when our proposed method is applied to Pseudo-Label and Mean Teacher, the decrease in performance due to the presence of novel classes becomes statistically insignificant. This suggests that assigning individual weights to unlabelled data is a promising approach to dealing with novel classes in semi-supervised learning.

An important work in the future is to investigate when novel classes can have a larger impact on the performance of semi-supervised learning algorithms. Synthetic data can be very useful for this type of work, because we can control the properties of the data. Another interesting research question is if the representation capacity of a model can affect the magnitude of the effect of novel classes. Finally, we will explore applying outlier detection and novelty detection techniques to deal with novel classes in unlabelled data.

References

Christopher M Bishop. Novelty detection and neural network validation. *IEE Proceedings-Vision, Image and Signal processing*, 141(4):217–222, 1994.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, 7(Jan):1–30, 2006.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *The Second International Conference on Learning Representations*, 2014.

Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pages 3581–3589, 2014.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *The Fifth International Conference on Learning Representations*, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2, 2013.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, and Shin Ishii. Distributional smoothing with virtual adversarial training. In *The Fourth International Conference on Learning Representations*, 2016.

Takeru Miyato, Andrew M Dai, and Ian Goodfellow. Adversarial training methods for semi-supervised text classification. In *The Fifth International Conference on Learning Representations*, 2017.

Avital Oliver, Augustus Odena, Colin A Raffel, Ekin Dogus Cubuk, and Ian Goodfellow. Realistic evaluation of deep semi-supervised learning algorithms. In *Advances in Neural Information Processing Systems*, pages 3235–3246, 2018.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch, 2017.

Antti Rasmus, Mathias Berglund, Mikko Honkala, Harri Valpola, and Tapani Raiko. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, pages 3546–3554, 2015.

Sebastian Ruder and Barbara Plank. Strong baselines for neural semi-supervised learning under domain shift. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*, pages 1044–1054, 2018.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *Advances in Neural Information Processing Systems*, pages 1195–1204, 2017.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(Dec):3371–3408, 2010.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.