Preface

The thesis report contains a preface that explains the topic of the thesis, the context (institute or company), the main findings in a few lines and the names of the members of the thesis committee. The preface may end with a few acknowledgements, and completed with name and date.

The main goal of this thesis is to study if, in a transfer learning scenario, the transferability of pre-trained weights is affected by the existence of segmentation noises for training examples and to explore methods compensating the negative effects if exist. Transfer learning is relevant when segmentations in a domain of interest are difficult to obtain on a large scale. The transferred model is often a classification model trained with a selective subset of images from ImageNet. Another choice of transferred model is a segmentation model trained with another segmentation dataset to pursue more transferable weights. However, various noises may occur in segmentations if not enormous effects were made to correct them. Being able to learn comparable transferable weights even in the presence of these noises can be beneficial to save efforts made to correct every single segmenting error and create more segmentations using the saved efforts. This can be helpful in collecting segmentation datasets on a large scale with less effort and money cost when using the crowd-sourcing power.

We found that mis-segmentation noises had less influence on weights transferability compared to misclassification noises and inexhaustive segmentations. We proposed to categorize or binarize classes so that misclassified labels would not have as much as effects on weights transferability as training with the exact classes. A modification to cross entropy loss was proposed to alleviate the negative influence of unlabeled positive examples.

Members of the thesis committee include Prof. dr. A.Hanjalic (Multimedia Computing Group, TU Delft) as the chair, dr. J.C. van Gemert (Vision Lab, TU Delft) who was the daily supervisor of the student, and Prof.dr. M. Loog (Pattern Recognition Laboratory, TU Delft) and dr. Z. Szlvik (CAS Benelux, IBM).

I sincerely appreciate the magnificent supports provided by dr. J.C. van Gemert, Prof.dr. M. Loog and dr. Z. Szlvik as co-supervisors day to day. I would also like to thank dr. D.M.J. Tax for his expert knowledge in the domain.

Jihong Ju
August 21, 2017

Learn transferable features with noisy segmentation datasets

First Author
Institution1
Institution1 address
firstauthor@i1.org

Second Author
Institution2
First line of institution2 address
secondauthor@i2.org

August 21, 2017

Abstract

The broad existence of noisy datasets motivates us to consider if it is possible to utilize noisy segmentations in training better image segmentation models with limited training samples. We investigated in this work how to pre-train convolutional neural network (CNN) models with potentially incomplete, mislabeled segmentations. The influence of segmentation noises on model transferability was investigated in an experimental setup with synthesized segmentation noises. With the same setup, we discovered that binarizing classes as foreground and background could improve feature transferability for a small training set corrupted heavily with random labels. A class-dependent modification to the cross-entropy loss was proposed to balance precision and recall for training with incomplete segmentations. Compared to simply changing class weights, the proposed sigmoidal loss down-weights the losses for confident predictions and unconfident predictions differently. Experiments demonstrated that replacing the cross-entropy loss with a sigmoidal loss for background class provide an improvement to both pre-training and fine-tuning performance in the presence of incomplete segmentation.

1 Introduction

Most state-or-the-art convolution neural nets[25, 2, 12] benefit from transferring convolutional neural network (CNN) models trained on a subset of images from ImageNet. These ImageNet models[17, 34, 36, 13] were originally trained for an object recognition task, the ILSVRC[32] challenge, using around 1.2 million labeled images. In contrast to object recognition tasks, it is tougher to collect a semantic segmentation dataset on that large-scale. The largest segmentation datasets, Microsoft COCO2014[23], contains 123,287 images of 80 object categories, smaller by a factor of 10 than the ILSRVC dataset. The difficulty of obtaining segmentations is natural because it costs much more efforts for people to segment than to classify an image. Therefore, Semantic segmentation datasets normally have a much smaller scale than object recognition dataset. A commonly used method of improving segmentation performance in the limitation of lacking training samples is to transfer weights from the pre-trained ImageNet models as performed in [25, 2].

However, it can be challenging to adapt the ImageNet models for semantic segmentation. Firstly, ImageNet models cannot be employed directly to RGB-D images or 3D images like CT scans and MRI scans in 3D. Secondly, segmentation models do not necessarily follow the architecture design of classification models. For example, the object recognition models pursue features invariance to better capture semantics regardless the variations in objects. The result translation invariant and resolution-reduced features could reduce the localization accuracy which is not essential for object recognition but is critical for object segmentation.[40, 2] The challenges in adapting ImageNet models can result in CNN architectures different for segmentation models[40]. Lastly, ImageNet models were trained with natural images at relatively low resolution. In some domains of interest, images can be non-

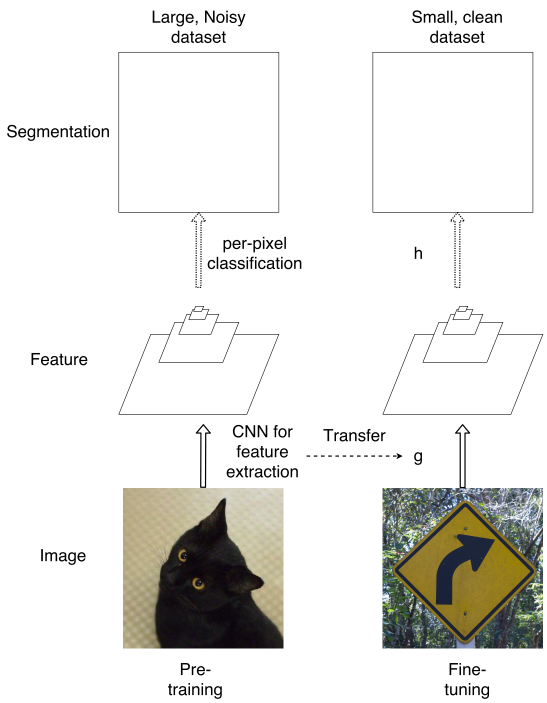

Figure 1: Pre-train segmentation models pre-trained in the presence of mislabeled segmentations. The result convolutional neural network (CNN) for feature extraction is transferred and fine-tuned with a small set of clean segmentation in the domain of interest.

natural, such as aerial images, images from bird's eye view, and medical images; In some other domains, images may have different lighting conditions or have a high resolution than the ImageNet ones. It can be beneficial to further train ImageNet models using a dataset in a similar domain as the one of interest. In any of these three cases, pre-training a segmentation model with another segmentation dataset can be valuable to achieve good segmentation performance with a small training set available.

The segmentation datasets for pre-training may, however, contain label noises. The use of the crowd-sourcing platform like Mechanical Turk is common nowadays to collect datasets on a large-scale. It is natural for crowd-sourcing workers to make mistakes as a result of lack of expertise, inherent ambiguity of tasks or unconscious bias. Enormous efforts are required, according to [23, 7], to ensure the correctness of segmentations. If not requiring "gold standard" segmentations, the efforts saved for correctness can be made to segment more images for a larger dataset. Besides, some labels other than the manual ones may be freely available for particular tasks. But these labels often contain structural noises depending on the way they were created. For example, digital maps, like OpenStreetMap, can be used to segment aerial images. Segmentations constructed from the maps could suffer from incompleteness as well as registration problems.[27]

Ideally, the use of these noisy datasets for pre-training should not affect transferring the learned model to another dataset. But if negative influences introduced by label noises are remarkable, methods of compensating the noises become necessary.

Noises of different kinds can exist in segmentation labels. In particular, we consider mislabeling for the whole segment, assuming the outline of objects is always correct. We categorized object mislabelling into three categories: inexhaustive segmentation, misclassification, and false segmentation. **Inexhaustive segmentation** means that there exist objects left unsegmented. A typical scenario where incomplete segmentation emerges is to segment images containing massive amounts of objects of the same kind, e.g., a flock of sheep or a pile of products; **Misclassification** of objects from one category to another can exist occasionally. For example, the Microsoft COCO dataset[23] contains some misclassified bears and teddy bears even though annotators were asked to segment only one category at a time; **False segmentation** denotes that semantically meaningful objects from an undefined category are wrongly segmented, as objects of interest. It may occur due to the unclear definition of categories, visual similarities between objects, etc. We synthesized these three types of noises with a well-annotated dataset separately and studied their influences to the trained weights transferability.

Supposing misclassification had a negative influence on feature transferability, as we will show in Section 5.1, simply segmenting the foreground objects against the background can be a straightforward way to correct misclassified objects. It can be regarded as converting precise but potentially inaccurate labels to accurate but imprecise labels. Jain et al.[16] proved the possibility of training a fully convolutional network to generate foreground segmentations. Surprisingly, the model trained on over 1 million images generalizes well to object categories not included in the training. Training transferable features may not require as precise supervision as training classifiers because transferability of features are correlated to feature generality, i.e., if they are independent of particular categories[39].

If we consider inexhaustive segmentation only, the problem becomes similar to a so-called *positive and unlabelled learning* (PU learning) setup[21]. In the positive and unlabeled learning setup, the training dataset has two sets of examples: a *positive (P) set*, containing only positive examples, and an *unlabeled (U) set*, containing a mix of positive or negative examples. The P set in an incompletely segmented dataset is comprised of segmented pixels while the rest of the pixels construct the U set. The main characteristic of the U set is no easy way to generate reliable negative labels out of it. The traditional semi-supervised learning techniques are consequently not applicable as a result of the absence of negative training samples. The set of background pixels containing unsegmented objects can fulfill this property of U set. Learning to segment objects in the presence of inexhaustive segmentation can be therefore considered as a learning problem with only positive examples and unlabeled examples. In this work, we treated the unlabeled set as a set of examples with noisy negative labels, following a branch of previous studies for PU learning as discussed in Section 2.

The main topics discussed in this thesis are:

1. How labels noises influence feature transferability of CNN models for semantic segmentation.

2. If binarizing classes for pre-training could alleviate the negative effects of misclassified objects in segmentations if the negative effect exists.

3. If modifications to the cross-entropy loss could achieve better performance in the presence of inexhaustive segmentations.

The rest of this thesis is organized as follows: In the next section, we summarize related works. In Section 3 we formulate the problem of transferring segmentation models pre-trained with noisy segmentations, and how to synthesize the three types segmentation noises. We introduce a sigmoidal loss for the negative class to compensate noisy negative labels in Section 4. Experiments in Section 5.1 was designed to study the influences of inexhaustive segmentations, misclassification and false segmentation separately. The proposed sigmoidal loss, together with the class-weighted loss and a modified hard bootstrapping loss, was evaluated in synthesized PU learning setups in Section 5.2. Discussions are presented in Section 6 and conclusions are summarized in Section 7.

2 Related work

Transfer Learning Weights of convolutional neural networks were proved "transferable" not only to another dataset[33, 39] but also to other applications[9, 25]. Transferring weights of pre-trained CNN models allows to improve the model performance without engaging in much efforts spent for data-labeling.[28] Yosinski et al.[39] reported that initializing a network with transferred features from almost any number of layers can produce a boost to the fine-tuning performance with better generalization. In other words, transferability is possible between completely dissimilar tasks, for example, from natural objects to human-made objects. If transferring features from dissimilar datasets is possible, it could also be possible to transfer weights from a noisy dataset.

Apart from supervised pre-training, one can also perform unsupervised learning to obtain pre-trained features typically with auto-encoders[38, 26], deep belief networks[14, 19]. Though a few studies[6, 5, 1] discussed the advantage of unsupervised pre-trained features compared to random weights initialization, the distance between the two has been shortened ever since the arises of modern initialization strategies, namely Xavier initialization[10] and its variants. We used random weights initialization as the lower bounds for pre-training with noisy labels. Features learned in the presence of label noises should at least outperform random weights because noisy information should be still better than no information.

Deep Learning with Noisy Labels The impact of randomly flipped labels on classification performance has been investigated by [35, 29] for convolutional neural networks. They both reported decreases in classification performance as the proportion of flipped labels increases for a fixed number of training samples. By contrast, Rolnick et al.[31] argued that deep neural networks can learn robustly from the noisy dataset as long as appropriate choices of hyperparameters were made. They studied the effects of diluting instead of replacing correct labels with noisy labels and concluded that larger training set is of more importance than lower the level of noise. None of these studies explored the influence of label noises on feature transferability.

Methods, including a linear noise layer on top of the model output[35], loss correctness with an estimation of the noise transition matrix[29] were proposed to compensate the negative effect on classification performance introduced by flipped labels. We designed our experiments using noisy segmentations synthesized with perfect segmentation as well.

Positive and Unlabeled Learning Traditional positive and unlabeled learning methods proposed for text classification[24] do not extend well to deep learning models. These traditional methods often follow a two-step strategy: first identifying a set of reliable negative samples (RN set) from U set and then iteratively build a set of classifiers with RN set and P set, while updating the RN set with a selected classifier. It would take tremendously longer time to train a couple of deep learning models iteratively than to train a sequence of naïve Bayesian (NB) models or supported vector machines (SVMs). Therefore, it would be unrealistic to train CNN models in this manner.

Alternatively, one can treat all unlabeled examples as negative and simply reweigh positive and negative examples[20]. Under the assumption of randomly labeled positive examples, Elkan & Noto[4] demonstrated that a classifier trained on positive and unlabeled examples predicts probabilities that differ by only a constant factor from the true conditional probabilities of being positive. These two works considered only binary classification whereas we showed that it is possible to train deep neural networks for multiple classes with only positive and unlabeled examples.

The problem of weighting negative examples is that it assumes the probability for a negative label being wrong is independent of the underlying distribution of inputs. Given a nonrandom model, the confidence of a model prediction conveys information about the underlying inputs distribution. A confident, positive prediction indicates the example is more likely to be positive than to be negative. Therefore, we instead assumed that the probability of mislabeling for a negative label is correlated to the prediction confidence of a trained model. In other words, we assumed the confident contrary prediction is more likely to be mislabeled than unconfident examples. Lin et al.[22] proposed a so-called Focal Loss to down-weight confident predictions and thus focus training on hard negatives for imbalanced learning. The sigmoidal negative loss we proposed in Section 4 follows a similar design as [22] to not over-punish confident positive predictions for samples with negative labels. Additionally, Reed & Lee[30] proposed a bootstrapping loss to emphasize *perceptual consistency* in training when incomplete and noisy labels exist. We modified the hard bootstrapping loss to interpret the prior knowledge that positive labels are reliable.

3 Feature transferability with label noises

3.1 Problem Formulation

Semantic Segmentation A deep learning model for semantic segmentation normally consists of two principal functions: a CNN feature extractor g that extracts hierarchical feature maps F from images I, followed by a classifier h that generates pixel-by-pixel prediction to fit labels S. Together they form a segmentation model f to predict class probabilities for each of the pixels in a given image I:

$$f(I) = h(g(I))$$

Training is to find an optimal f from the space of functions which minimizes a loss function L that measures the distance between S and $f(I)$:

$$f^* = \operatorname*{argmin}_f L(S, f(I))$$

Feature Transferability A pre-trained feature extractor, g_t, can be transferred as an initialization of the feature extractor and is expected to result in a better performed f^* on the fine-tuning test set than a random choice of initialization g_0. The corresponding fine-tuning performance improvement indicates the transferability of the pre-trained feature extractor. Ideally, a feature extractor pre-trained with noisy labels would result in a fine-tuned model with equivalent performance as one pre-trained with true labels. The difference in the result fine-tuning performance improvement between the noisy and clean pre-trained feature extractor can tell the influence of label noises on feature transferability.

3.2 Label noise synthesization

It is challenging to find a dataset with both clean and noisy labels available, so we tried to synthesize segmentation noises with correct labels. A straightforward way to synthesize noisy labels is to corrupt true labels stochastically for each segment with a corruption model. The corruption model simply describes the probability of the observed labels given the true labels.

For segmentation problems, each pixel (or voxel for 3D segmentation) of a training image has a label assigned to one of the pre-defined categories. Supposing there are K pre-defined categories, the label of a pixel in the i-th row and j-th column from an image of size $h \times w$:

$$y_{ij} = \begin{cases} 1 \leq k \leq K, & \text{for foreground pixels} \\ 0, & \text{for background pixels} \end{cases}$$

where $1 \leq i \leq h, 1 \leq j \leq w$ and $i, j, k \in \mathrm{Z}^+$.

Inexhaustive segmentation

Given the true labels, which pixels belong to the same object is known. Inexhaustive segmentation can be synthesized as flipping all pixels for an object of the k-th category from k to 0 with probability p_{0k}. The probability for these pixels to have correct labels is then: $1 - p_{0k}$.

Misclassification

To synthesize misclassification, we can stochastically convert pixel labels of segment for an object of the k-th category from k to j with probability p_{jk}, where $j, k \leq K$ and $j, k \in \mathrm{Z}^+$. Probabilities for all possible combinations of j and k form a *confusion matrix* in which probabilities in each column sum up to 1.

False segmentations

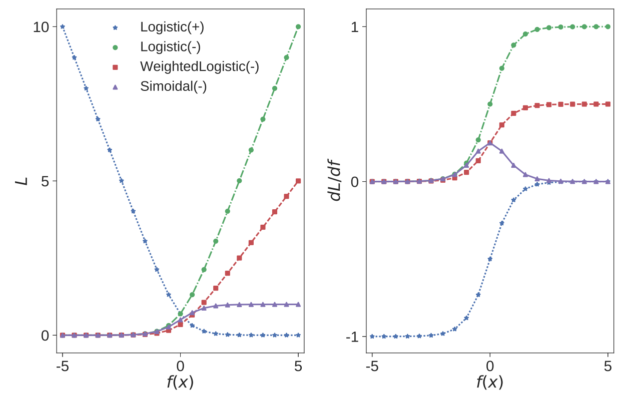

Figure 2: The differences in losses (top figure) and derivatives (bottom figure) with repect to logits between the weighted logistic negative loss and the sigmoidal negative loss. The **+** sign represents loss for positive samples and the **-** sign stands for loss for negative samples. The sigmoidal loss for negative examples reaches a plataeu and the derivative drops to zero in the very positive region, whereas the weighted logistic loss for negative is just a linearly scaled logistic loss.

The presence of false segmentations can be synthesized in two steps: excluding classes except one from the foreground categories, forming the clean labels set, and assigning pixels of the excluded categories with foreground labels with a probability of p_{11}, constructing the noisy labels set.

4 Modifications to the cross-entropy loss for PU Learning

Learning with only positive and unlabeled examples A training set of the positive and negative (PU) learning problems contains only a set of positive examples (P set) and a set of unlabeled samples (U set). Unlabeled examples can be either positive or negative, meaning there is no reliable negative examples available One straightforward way to generate negative examples for training is to treat all the unlabeled examples as a set of negative examples with noises. The problem then converts to learning with clean positive and noisy negative labels. The ultimate goal of solving such a problem is to learn a model that predicts as many positives as possible while keeping the false positive rate low, regardless the influence of false negative labels.

Weighted negative loss Clearly, it is possible to compensate the negative influence of false negative labels by simply weighing the positive and negative examples differently, namely, let the positive and negative examples have different rates of contribution to the total loss. Suppose a logistic loss is used, the corresponding weighted losses for positive and negative samples are:

$$l_+ = -\log p(y_i = +1|x_i)$$
$$l_- = -q \log p(y_i = -1|x_i), 0 < q < 1$$

where $p(y_i|x_i) = \sigma(f(x))$ denotes the probablistic predictions for the i-th example given by model $f(\cdot)$, with the sigmoid function $\sigma(\cdot)$ as the activation function. This loss is called **weighted negative loss** in this paper. Empirically, the choice of q can be made based on the highest precision and recall achieved on a validation set.

Sigmoidal Negative Loss We used a class-dependent loss to down-weight the loss contribution of very positive predictions negative labels and still making full use of the clean positive labels. The loss for positive examples is still a normal logistic loss and the loss for negative examples is replaced with a sigmoidal loss [37]:

$$l_+ = -\log p(y_i = +1|x_i)$$
$$l_- = 1 - p(y_i = -1|x_i)$$

where p again denotes the sigmoid-activated model output. We called this class-dependent loss **sigmoidal negative loss** in a sense it uses a sigmoidal function as the loss for negative samples.

Figure 2 shows the differences in losses and derivatives with respect to model output between weighted negative logistic loss and sigmoidal negative loss. The main feature of sigmoidal loss for negative examples is its relatively small changes in the region of confident positive, compared to the logistic negative loss and 0.5 weighted negative loss. As a consequence of this feature, the corresponding derivative decreases to zero as the model prediction increases in the positive direction.

Modification to the bootstrapping loss for PU learning We have also modified the hard bootstrapping loss by Reed et al.[30] to encourage confident positive predictions and compensate noisy negative labels:

$$l_- = -\beta \log p(y_i = -1|x_i) - (1-\beta) \log p(y_i = \hat{y}_i|x_i)$$

where $\hat{y} = \mathrm{argmax}_{j\in\{-1,+1\}}\, p(y_i = j|x_i)$ is the model prediction and $0 < \beta < 1$. The first term of the objective is a weighted negative loss and the second term can be considered as a regularization term to encourage consistent predictions. This loss is referred as **bootstrapping negative loss** for the rest of this paper.

Extend PU learning to multiclass problems For classification problems with multiple positive classes and one negative class, the weighted negative loss, sigmoidal negative loss and bootstrapping negative loss can be extended to train deep learning models in the presence of negatively labeled examples from various positive classes. The logistic loss naturally extends to the multi-class scenario as the sigmoid activation extends to the softmax function. Similar modifications can be then made to the cross-entropy loss, namely, keeping the losses for positive classes unchanged and applying the modifications to the loss for the negative class. We used the manner of extending the losses in this work. Alternatively, one can apply a one-vs-all strategy, with which the normal logistic loss is used for positive classes while the weighted, sigmoidal and bootstrapping loss can be used for the negative class.

Extend PU learning to segmentation By assuming the probability of missing positive label for a pixel is independent of its neighbor pixels, the weighted negative loss, sigmoidal negative loss and bootstrapping negative loss are all applicable to per-pixel classification problems if classification for each pixel is made independently.

Implementation details We introduced the sigmoidal negative loss after training with the normal cross entropy loss for a few epochs. The sigmoidal loss for negative examples saturates for very positive outputs, meaning that the confident, positive prediction has little contribution to the total loss. This can introduce problems at the beginning of the training procedure when the confident predictions are likely to be made at random. Additionally, optimization could reach the plateau where the model made all positive predictions with high confidence. We applied a similar "fade-in" mechanism to the modified hard bootstrapping loss as well because it also relies on a nonrandom model for sufficiently reliable prediction $\hat{y}$.

Another problem encountered in the PU learning setup is the class imbalance introduced by negatively labeled positive samples. A balanced dataset can become imbalanced in the presence of false negative labels, especially if only a small portion of positive samples are correctly labeled. We reweighed positive and negative samples based on their occurrences of the observed labels to alleviate the influence of imbalance for training. Note that the class-weighted logistic loss reweighed the classes in addition to this frequency balancing class weight.

5 Experiments

5.1 The influence of synthesized segmention noises on feature transferability

Experiment setup To investigate the influence of inexhaustive segmentations, misclassifications and false segmentations on feature transferability, we set up experiments with synthesized label noises from a well-annotated dataset, the PASCAL VOC2011 segmentation dataset[7]. Fifteen out of twenty categories of the VOC2011 dataset were selected to form a *pre-training dataset* and the other categories formed a *fine-tuning dataset*. The pre-training dataset was used to train a Fully Convolutional Network with AlexNet (FCN-AlexNet) model[25] for segmentation in the presence or absence of synthesized segmentation errors. The fine-tuning dataset was used to fine-tune the weights of convolutional layers from the pre-trained FCN-AlexNet models. Inexhaustive segmentations, misclassifications, and false segmentations were synthesized separately with stochastical corruptions to the well-annotated pre-training dataset, followed the descriptions in Section 3.1. Fine-tuned models were evaluated by mean intersection over union ratio (mean IU) achieved on the fine-tuning test set, referring to as the *fine-tuning performance*. Performance improvement of fine-tuning transferred models compared to an randomly initialized model indicates the transferability of pre-trained weights.

Experiment details To avoid that the choice of pre-training and fine-tuning splitting for categories influence the results, the 20 categories of VOC2011 were divided equally into four folds. Each fold was studied separately, and the exact partitions of each fold are listed in Table 1. The training dataset was enriched with extra segmentations by Hariharan et al.[11] To keep the segmentation task simple, we used only single-object images, resulting in totally 4000 training images for 20 categories available for pre-training, fine-tuning and evaluation. In order to accelerate the training process, we subsampled the original images by four times. Fully Convolutional Networks with AlexNet was used for experiments because of its relatively small capacity and thus short training time. The existence of an ImageNet model for AlexNet can be beneficial to set a performance reference. The non-transferred layers were randomly initialized with Xavier Initialization. The ImageNet model and completely random weight initialization were considered as the upper bound and lower bound, respectively, for various pre-trained weights summarized in Table 1. The default hyperparameters of FCN-AlexNet in [25] were kept unchanged. Training run 240,000 iterations for pre-training phase, and 12,000 iterations for fine-tuning phase. Snapshots for trained models were taken every 4,000 iterations. Each experiment was repeated three times, mean and standard deviation were computed over the last five snapshots for all repetitions.

Inexaustive Segmentation Fine-tuning performances of transferring models pre-trained with complete labels and half of the objects unsegmented respectively are summarized in Table 1. The HalfUnsegmented model transferred into a fine-tuned model with an average mean IU 0.04 worse than the CompleteLabels model, almost the same as training a model with random weight initialization. This result indicates that inexhaustive segmentation can have a negative impact on weights transferability. By applying the sigmoidal negative loss to pre-train models with half of the objects segmented, we were able to achieve a comparable fine-tuning performance as using the complete segmentation to pre-train.

Misclassifications Models pre-trained with random labels are listed in Table 1. Compared to the CompleteLabels model (the one trained with true labels), both the model trained with all random labels or half true half random labels led to worse fine-tuning performances. Fine-tuned performances of the AllRandomLabels model and the HalfRandomLabels model were no better than randomly initializing model weights, indicating poor weights transferability of a trained model in the presence of random labels to segmentations. In other words, misclassification noises in segmentation can impact the transferability of convolutional weights negatively.

However, binarizing labels for pre-training was able to achieve equivalent fine-tuning performance as using true labels. It achieved better performance than training to segment the exact classes but with random labels. Binarized labels are accurate but imprecise because pixel labels were randomized among foreground classes and binarizing labels lead to correct segmentation for foreground and background. This observation indicates that inaccurate labels may have more significant negative influence than imprecise labels for feature transferability.

Additionally, we studied the influence of categorizing classes for pre-training to keep the labels precise to some extent. We categorized the fifteen classes in the pre-training set into person, animal, vehicle, indoor according to [7] and trained a model to transfer, shown as the error bars on lines in figure 3 at categories 4. As a comparison, the fifteen classes were also randomly categorized into 4, 7, 11 categories and shown as isolated error bars in figure 3 at categories 4, 7, 11 respectively. Figure 3 shows that categorizing pre-training classes had little effect to the fine-tuning performance of transferred models. Even categorizing classes at random without explicit meaning could pre-train weights better than random initialization (shown as error bar at categories 0). It can be beneficial to binarize or group classes into hyper-categories to learn better transferable weights in the presence of misclassifications.

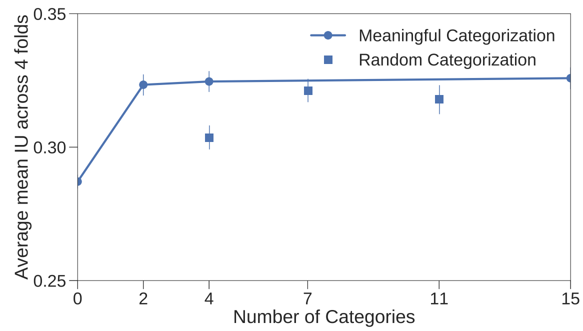

Figure 3: Test performance for models fine-tuned from pre-trained weights using data of categorized pre-training classes. Isolated error bars denote random categorizations (RC) of the 15 classes, while error bars located on lines denote the meaningful categorization. Zero categorize means no pre-trained weights used and the model was random initialized. The displayed mean IU/mean accuracies and standard deviations were averaged over four folds listed in Table 1. The trend shows that binarizing and categorizing classes had little negative effect on feature transferability.

False segmentations To synthesize false segmentations, we selected one category, either cat or dog depending on the folds, as the target category and all the other 14 categories in the pre-training dataset became non-target, as discussed in Section 3.1. In the presence of false segmentations, instances from non-target categories can be misannotated as the target category with a probability of $p_{10} = 0.5$. The result pre-trained models is named as FalseSegmentHalf in Table 1 respectively. The noise-free counterpart of is the model pre-trained with segmentations of the selected target category only and the other 14 categories remained unsegmented, denoted as NoFalseSegmented.

Compared to the NoFalseSegmentated model, transferring the HalfFalseSegmented model introduced no significant decrease in fine-tuning performance. This observation indicates that false segmented meaningful objects may have a little negative impact when they are used for pre-training transferable weights.

5.2 Learn to classify with positive and unlabeled samples

2-dimensional moons dataset Figure 4 shows the decision boundaries of a 2-layer multilayer perceptron, with 6 neurons per layer, trained with the weighted loss, the sigmoidal loss and the bootstrapping loss. Four hundred samples per class were drawn randomly from two inter-

	Initial Feature Extractor	Fine-tuning mean IU per pretraining-finetuning fold				Average mean IU across four folds
Fine-tuning categories		aeroplane, bicycle, bird, boat, bottle	bus, car, cat, chair, cow	dining table, dog, horse, motorbike, person	potted plant, sheep, sofa, train, TV	
Upper bound & lower bound	ImageNetModel	0.42 ± 0.01	0.51 ± 0.01	0.49 ± 0.01	0.47 ± 0.01	0.47 ± 0.01
	RandomWeights	0.29 ± 0.01	0.29 ± 0.03	0.27 ± 0.01	0.30 ± 0.02	0.29 ± 0.02
Inexaustive Segmention	CompleteLabels	0.29 ± 0.01	0.36 ± 0.01	0.29 ± 0.01	0.37 ± 0.01	$\mathbf{0.33 \pm 0.01}$
	HalfUnsegmented	0.26 ± 0.01	0.30 ± 0.03	0.28 ± 0.03	0.32 ± 0.02	0.29 ± 0.02
	SigmoidalLoss	0.30 ± 0.01	0.37 ± 0.01	0.31 ± 0.02	0.34 ± 0.02	$\mathbf{0.33 \pm 0.02}$
Misclassification	AllRandomLabels	0.29 ± 0.01	0.33 ± 0.03	0.26 ± 0.01	0.28 ± 0.01	0.29 ± 0.01
	HalfRandomLabels	0.27 ± 0.01	0.33 ± 0.02	0.25 ± 0.01	0.29 ± 0.01	0.29 ± 0.01
	BinarizedLabels	0.30 ± 0.02	0.35 ± 0.01	0.29 ± 0.02	0.35 ± 0.03	$\mathbf{0.32 \pm 0.02}$
False Segmentaion	NoFalseSegmented	0.26 ± 0.01	0.37 ± 0.03	0.27 ± 0.01	0.33 ± 0.04	$\mathbf{0.31 \pm 0.02}$
	FalseSegmentedHalf	0.27 ± 0.01	0.34 ± 0.01	0.30 ± 0.01	0.32 ± 0.01	$\mathbf{0.31 \pm 0.01}$

Table 1: Segmentation performance for FCN-AlexNet models pre-trained with 15 categories from the PASCAL VOC2011 dataset and fine-tuned with the other 5 categories. **ImageNetModel** represents the pre-trained ImageNet model (upper bound); **RandomWeights** indicates that the randomly initialized weights (lower bound); All the other extractors were pre-trained in the presence or absence of the corresponding label noises listed in the leftmost column. Half of the objects unsegmented result in pre-trained models not significantly better than random weight initialization. Introducing the sigmoidal negative loss in the pre-training phase was able to improve the fine-tuning performance to be comparable to pre-trained model with complete segmentations. Random labels interfered the fine-tuning performance of transferred models and binarizing classes as foreground and background for in the pre-training dataset help overcome the negative effects of random labels. Including false segmentations had little influence on transferred models.

leaving half circles with noises added with a minor standard deviation. The leftmost column in the figure shows the multilayer perceptron trained with complete positive labels and the normal logistic loss while the other three columns show the classifier trained with half of the positive examples labeled as negative. For sigmoidal negative loss, the mislabeled positive examples faraway from the decision boundary have smaller derivatives the ones closer to but still distant from the decision boundary. As a consequence, optimization performed as counting the weights update contributions differently for confident and unconfident prediction instead of simply decreasing the overall estimation for the negative loss. The result decision boundary becomes more distant from the positive cluster, and confident predictions are made in more areas of the space.

CIFAR dataset Images from the CIFAR10 dataset were used as positive (P) examples, and images from the CIFAR100 dataset were considered as the negative (N) examples. We trained a CNN model to classify images into eleven classes: ten positive classes from CIFAR10 and a negative class for images from CIFAR100. To synthesize a PU learning setup, we correctly labeled only part of positive images from CIFAR10 and the rest were assigned negative labels together with CIFAR100 images, forming the unlabeled (U) set. An eight layer CNN model was trained with different choices of losses and with the labeled positive examples and unlabeled examples. Model performances were then evaluated on a separate test set of combined CIFAR10 and CIFAR100 with true labels.

The architecture of the CNN model can be found in Appendix C. Each model was trained from scratch with Adam optimizer and base learning rate 0.0001. Experiments were repeated three times with random split of P

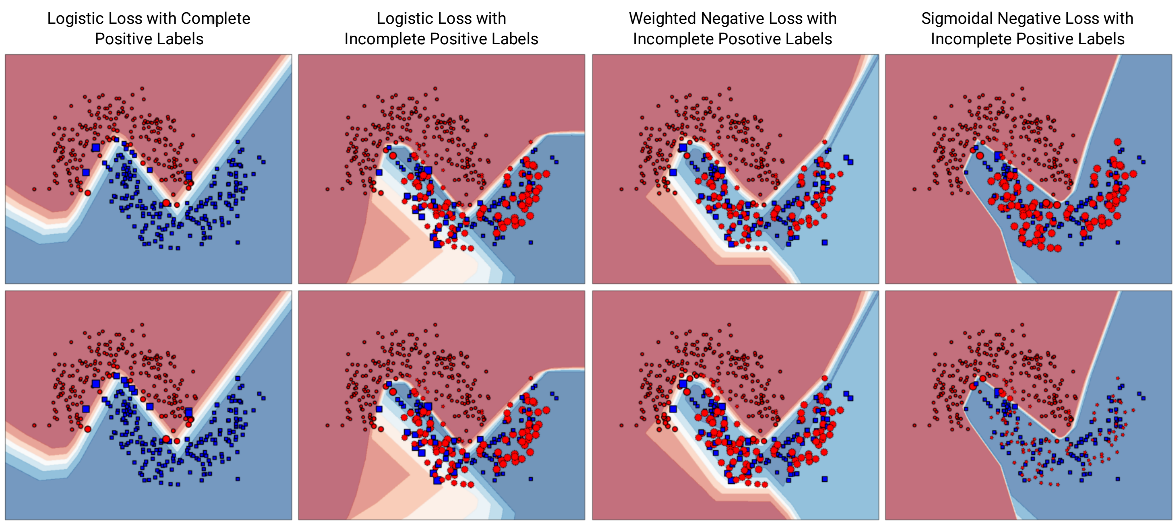

Figure 4: Decision boundaries, optimal losses, and derivatives of a 2-layer multilayer perceptron trained with different losses on a 2D moons dataset with the unlabeled positive. The **top row** demonstrates the per-class normalized training losses with **markers sizes** and the **bottom row** displays the per-class normalized loss derivatives w.r.t the output with marker sizes. A **red circle** indicates an example labeled as positive and a **blue square** indicates the example has a negative label. The **background colors** represent the probability for the area to be positive given by the classifier trained with the given samples and labels: **red** for high probability areas, **blue** for low probability areas and **white** for the class transition areas, i.e.decision boundaries. Compared to the normal logistic loss and weighted logistic loss, the decision boundary optimized with sigmoidal negative loss is more distant from the positive cluster. The sigmoidal negative loss has saturated losses and low derivatives for predictions farther from the decision boundary.

set, and U set and standard deviations were around 0.01 if not explicitly mentioned. Note that there is no category overlap between CIFAR10 dataset and CIFAR100 dataset.

Table 2 summarizes the test precisions and recalls of using different losses to learn with true positive labels and noisy negative labels, compared to training with the normal cross-entropy loss. With 50% of the positive example correctly labeled and the rest assigned negative labels, the normal cross-entropy loss led to an imbalanced model with high precision but low recall, and therefore with a low f1-score. By reweighing the negative loss by a factor of 0.5, we were able to balance precision and recall and improve the resulting f1-score significantly. Sigmoidal negative loss and hard bootstrapping loss were able to achieve even slightly better f1-scores than the weighted negative loss, though not as good as training with clean labels with the same number of positive examples. It is noteworthy that the weighted negative loss and hard bootstrapping loss achieved more balanced precision and recall than the sigmoidal negative loss. That is because the two former losses were weighted by classes frequency of observed labels, around 0.67 for the negative class and 2 for positive classes, whereas the sigmoidal negative loss was not. Reweighing the sigmoidal negative loss by observed label frequencies would trade too much precision for recall (0.74 and 0.83 respectively), resulting in a worse f1-score than not reweighing losses. The sigmoidal negative loss seems to be easier over-balancing with the same choice of class weights.

By varying the percentage of labeled positive examples, we compared learning with the labeled positive examples and noisy negative examples with the three different losses as shown in Figure 5. Similarly as a result at 50% positive examples labeled, the sigmoidal negative loss and hard bootstrapping loss had only little improvement compared to the weighted negative loss. The assumption we made for the sigmoidal negative loss was that the probability of a negative example being wrong is dependent on the confidence. However, the synthesized mislabeled positive examples were distributed at random when we synthesized them, meaning that the probability for a negative label being wrong is independent of the underlying distribution of the inputs. No matter where the decision boundary is, such uniformly distributed incorrect negative labels are independent of the prediction confidence. Therefore, it is expected for the sigmoidal negative loss

Annotation	Loss	acc.	mean prec.	mean rec.	mean F_1
P+N	CrossEntropyU.	0.87	0.88	0.82	0.85
50%(P+N)	CrossEntropyU.	0.83	0.84	0.78	0.80
50%P+U	CrossEntropyU.	0.66	0.94	0.38	0.49
50%P+U	WeightedNeg.	0.78	0.75	0.75	0.76
50%P+U	SigmoidalNeg.	**0.81**	**0.85**$\pm$0.03	0.72$\pm$0.03	0.77
50%P+U	BootstrapHard	0.80	0.76	**0.81**	**0.78**

Table 2: Overall accuracy, the average of precision, recall, and f1-score over classes on a test set of the CIFAR dataset with true labels. CIFAR10 images were used as positive images (P), and CIFAR100 images were used as negative images (N). The top columns summarize the upper bound, lower bound and reference for learning with positive and unlabeled examples. **Upper bound** Model trained with the complete positive and negative labels; **Reference** Model trained with 50% of the positive and negative examples; **Lower bound** Model trained with 50% labeled positive examples and unlabeled examples; The proposed losses should achieve as high recall as possible while keeping the precision high as well. The modified hard bootstrap loss achieved the highest average f1-score with 50% positive examples unlabeled.

and bootstrapping loss to have similar performance as the weighted negative loss.

Figure 5 also demonstrates that learning with a subset of reliable negative examples (PN setup) outperformed learning with positive and unlabeled examples (PU setup) for any percentage of labeled positives and any losses being used. This result was expected because the PN setup delivers extra information about which images in the unlabeled set are negative. The PU setup is therefore only relevant when it is impossible to easily construct a subset of negative examples from the unlabeled data. Otherwise, training with true positive and negative labels, and potentially with semi-supervised learning, would be superior to training with positive and unlabeled examples only.

Pascal VOC2011 segmentation task To study the effect of sigmoidal negative loss on compensating incomplete segmentations, we used again the PASCAL VOC2011 dataset with extra segmentation[11]. We synthesized inexhaustive segmentations the same way as described in Section 5.1. The same AlexNet-FCN model was trained together with the different loss functions for class 0 to predict binary segmentation, determining

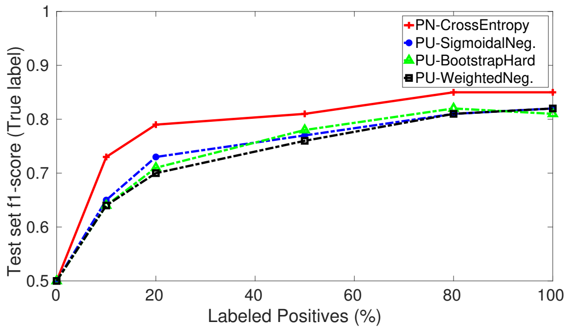

Figure 5: Comparing the f1-score achieved for weighted negative loss, sigmoidal loss and hard bootstrapping loss with varying percentage of labeled positive images **P+N** represents training with the percentage of images with reliable positive and negative labels, and **P+U** stands for training with the positive (P) and unlabeled (U) sets. The test performances of the three different losses did not show significant difference with the labeled positive percentage varying from 0% to 100%.

Annotation	Loss	overall acc.	mean acc.	f.w. IU	mean IU
Complete	CrossEntropy	0.90	0.85	0.82	0.75
50%Unseg.	CrossEntropy	0.85	0.68	0.73	0.60
50%Unseg.	WeightedNeg.	0.84	0.71	0.73	**0.62**
50%Unseg.	SigmoidalNeg.	0.83	**0.75**	0.72	**0.62**

Table 3: The best foreground/background segmentation performance achieved on the test set of PASCAL VOC2011 segmentation dataset with the normal cross entropy loss, weighted negative loss, and sigmoidal negative loss in the presence of inexhaustive segmentation. A class weight of 0.7:1.75 was used to balance the sample frequency differences of the two classes. The negative loss was further weighted by a factor of 0.5 for the weighted negative loss. Mean accuracy is equivalent to the average recall over the two classes. Mean IU is the average intersection over union ratio (IU) over two classes and f.w. IU is the frequency weighted average of IU over the two classes. Experiments were repeated twice and standard deviations were approximately 0.02. The sigmoidal negative loss achieved a better mean accuracy than weighted negative loss and a similar mean IU as the weighted negative loss. Both losses performed better than the normal cross entropy loss in the presence of 50% objects unsegmented.

whether a pixel belongs to an object or not. Only single-object images were used for training and testing to avoid the influence of two adjacent objects joining as one object because of binary segmentation. The same hyperparameters for optimization were used as in Section 5.1. The trained models were evaluated with the test set of PASCAL VOC2011 segmentation dataset with binary segmentations.

As shown in Table 3, the sigmoidal negative loss achieved the highest mean accuracy, approximately 0.07 better than training with the normal cross entropy loss. In contrast to the improvement of mean accuracy, mean IU for models trained with either weighted negative loss or sigmoidal negative loss did not show significant improvement compared to the normal cross entropy loss. The increase in the mean accuracy was caused by an increase in foreground accuracy and a decrease in background accuracy. In other words, the sigmoidal loss balances precision for recall, similarly as observed in training the CIFAR dataset. The decrease in precision, however, can interfere the improvement mean IU since the mean IU counts for both low false positive rate and low false negative rate.

We then applied the sigmoidal negative loss to the pre-training phase of the experiment for inexhaustive segmentations in Section 5.1. We were able to recover the negative influence of inexhaustive segmentations on the fine-tuning performance of the pre-trained models. Note that we learned foreground/background segmentation instead of multi-class segmentation when applied the sigmoidal negative loss because we discovered binarizing pre-training classes had little effect to feature transferability in this experimental setup.

Selective predictions made by models trained with the sigmoidal negative loss and the cross entropy loss were presented in Figure 6. For these two example images, the model trained with the cross entropy loss failed to segment objects from images whereas sigmoidal negative loss predicted segmentations on the position of the objects. The coarse outlines were mainly due to the limited compacity of the FCN-AlexNet model. The third column shows predictions given by model trained with complete training segmentation, and it did not produce more accurate outlines.

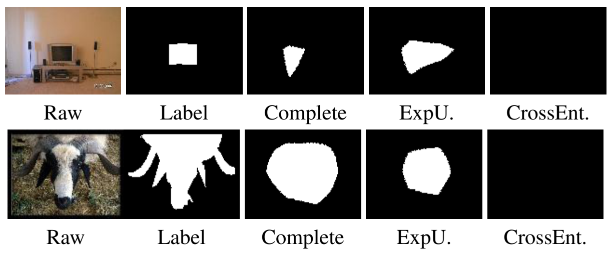

Figure 6: Selective predictions made by models trained with the sigmoidal negative loss and the cross-entropy loss. This figure presented two selective images for which the model trained with the cross entropy loss failed to segment objects, and the one with sigmoidal negative loss succeed.

6 Discussion

We assumed incomplete segmentation, misclassification and false segmentation of objects had occurred independently to the exact shape and appearance of the objects when we synthesized the segmentation noises stochastically in our experiments. However, some categories can be more likely to be misclassified due to its ambiguity in shapes or appearances in practice, such as bear and teddy bear. Similarly, ambiguous objects can have a higher probability of being missing than easily recognizable objects. A real dataset with both clean and noisy segmentation would be valuable to evaluate the influence of noises on feature transferability further.

We proposed to binarize or categorize classes to produce accurate but imprecise labels when misclassification of segments dominates a small pre-training dataset. The result feature transferability depends on whether the inaccurate labels or the imprecise labels is more severe in the decrease of feature transferability. In our experiment, the limited amounts of training images, the low-resolution of images and the constraint capacity of the FCN-AlexNet model may explain that more precise labels cannot improve the fine-tuning performance for the pre-trained models. In general, a trade-off between precise but less ccurate labels and more accurate but less precise labels need to be made to train better transferable features, given a particular dataset.

We argued that not over-punishing confident, positive predictions for samples with negative labels can be beneficial by assuming the confident predictions are more likely to be mislabeled given a nonrandom model. However, there can also be disadvantageous for over-excusing losses for confident predictions. One typical problem is that a model made bad predictions with confidence will keep emphasizing its wrong predictions. Therefore, the threshold between punishing and excusing a certain model for make confident contrary predictions need to be tuned appropriately to achieve both high precision and high recall. Unfortunately, the sigmoidal negative loss has no such hyperparameter to tune, which can be a direction to improve it in the future. For example, the Focal Loss[22] could be a parametrizable alternative of the sigmoidal negative loss.

Besides, applying the sigmoidal negative loss to segmentation assumes that the foreground-to-background mislabeling for each pixel occurred independently. This assumption made it possible to implement sigmoidal negative loss directly to classification for individual pixels. However, there is normally a spatial dependence for noises in pixel labels which may help improve the spatial consistency of predictions. Therefore, future works may be possible to make use of the spatial dependence of pixel label noises.

7 Conclusion

We studied how to pre-train transferable convolutional weights in the presence of inexhaustive segmentation, misclassification and false segmentation. We discovered that including false segmentations of meaningful objects for pre-training had little impact on the fine-tuning performance of transferred weights By contrast, misclassification noises can have negative impacts on feature transferability, but binarizing classes to foreground and background can produce accurate but not necessarily precise labels to train better transferable features. We presented that for a small pre-training set, binarizing classes can recover the negative influence of misclassification of object segments on the fine-tuning performance of the transferred models. Inexhaustive segmentation can also negatively affect feature transferability of the pre-trained model. The decrease of the fine-tuning performance due to the unsegmented objects in the pre-training set can be compensated by modifying the loss function for deep learning models. We then proposed a class-dependent loss to not over-punish the confident positive predictions for examples with negative labels. The proposed sigmoidal negative loss was demonstrated to improve both the pre-training and fine-tuning performance of models pre-trained in the presence of inexhaustive segmentations.

References

[1] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML Workshop on Unsupervised and Transfer Learning*, pages 17–36, 2012.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *arXiv preprint arXiv:1606.00915*, 2016.

[3] Vincent Dumoulin and Francesco Visin. A guide to convolution arithmetic for deep learning. *arXiv preprint arXiv:1603.07285*, 2016.

[4] Charles Elkan and Keith Noto. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 213–220. ACM, 2008.

[5] Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

[6] Dumitru Erhan, Pierre-Antoine Manzagol, Yoshua Bengio, Samy Bengio, and Pascal Vincent. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *Artificial Intelligence and Statistics*, pages 153–160, 2009.

[7] Mark Everingham, SM Ali Eslami, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International journal of computer vision*, 111(1):98–136, 2015.

[8] Benoît Frénay and Michel Verleysen. Classification in the presence of label noise: a survey. *IEEE transactions on neural networks and learning systems*, 25(5):845–869, 2014.

[9] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

[11] Bharath Hariharan, Pablo Arbeláez, Lubomir Bourdev, Subhransu Maji, and Jitendra Malik. Semantic contours from inverse detectors. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 991–998. IEEE, 2011.

[12] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. *arXiv preprint arXiv:1703.06870*, 2017.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[14] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

[15] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

[16] Suyog Dutt Jain, Bo Xiong, and Kristen Grauman. Pixel objectness. *arXiv preprint arXiv:1701.05349*, 2017.

[17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

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

[19] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th annual international conference on machine learning*, pages 609–616. ACM, 2009.

[20] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pages 448–455, 2003.

[21] Xiao-Li Li and Bing Liu. Learning from positive and unlabeled examples with different data distributions. *Machine Learning: ECML 2005*, pages 218–229, 2005.

[22] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *arXiv preprint arXiv:1708.02002*, 2017.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[24] Bing Liu, Yang Dai, Xiaoli Li, Wee Sun Lee, and Philip S Yu. Building text classifiers using positive and unlabeled examples. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 179–186. IEEE, 2003.

[25] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[26] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. *Artificial Neural Networks and Machine Learning–ICANN 2011*, pages 52–59, 2011.

[27] Volodymyr Mnih and Geoffrey E Hinton. Learning to label aerial images from noisy data. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 567–574, 2012.

[28] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

[29] Giorgio Patrini, Alessandro Rozza, Aditya Menon, Richard Nock, and Lizhen Qu. Making neural networks robust to label noise: a loss correction approach. *arXiv preprint arXiv:1609.03683*, 2016.

[30] Scott Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. Training deep neural networks on noisy labels with bootstrapping. *arXiv preprint arXiv:1412.6596*, 2014.

[31] David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.

[32] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[33] Hoo-Chang Shin, Holger R Roth, Mingchen Gao, Le Lu, Ziyue Xu, Isabella Nogues, Jianhua Yao, Daniel Mollura, and Ronald M Summers. Deep convolutional neural networks for computer-aided detection: Cnn architectures, dataset characteristics and transfer learning. *IEEE transactions on medical imaging*, 35(5):1285–1298, 2016.

[34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[35] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.

[36] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[37] David MJ Tax and Feng Wang. Class-dependent, non-convex losses to optimize precision. In *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pages 3314–3319. IEEE, 2016.

[38] 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.

[39] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.

[40] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip HS Torr. Conditional random fields as recurrent neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1529–1537, 2015.

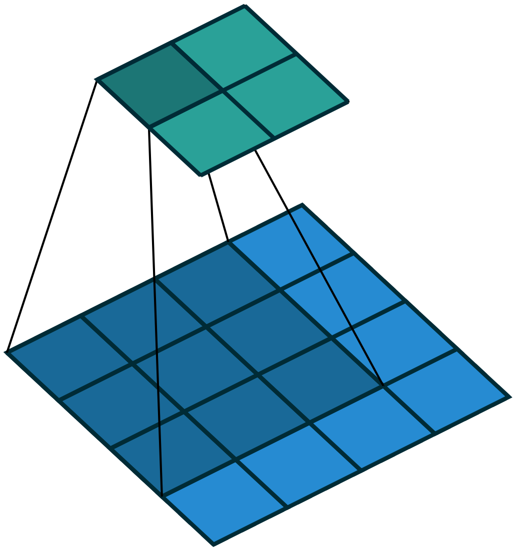

Figure 7: A basic convolution operation. A 3x3 convolutional filter convolves with a 3x3 window sliding over the image (bottom). The output of convolutions at each sliding position form a feature map (top). This figure was drawn by Dumoulin and Visin[3]

A Convolutional Networks for Semantic Segmentation

A.1 Convolutional Neural Networks

The primary components of a typical convolutional neural network (CNN) are several layers of convolution and sub-sampling, followed by a few fully-connected layers. Convolutions

Convolutional layer Subsampling

An example, LeNet-5 (1998) [18], is shown in Figure 8. The first convolutional layer of LeNet contains 6 convolutional kernels of size 5x5 and each convolutional kernels convolve with small windows sliding over the images and produce a feature map of size 28x28. Each output in the produced feature map is corresponding to a small subregion of the visual field (the image), called a *receptive field*. A following max pooling layer subsamples the feature maps by a factor of two by extracting the maximum values for every two adjacent pixels literally and vertically. The result feature map S2 has a shape of 14 by 14 and a receptive field of 6 by 6. Another sequence of convolutional and pooling layers generate feature maps of size 5x5 with receptive field 16x16. Neurons in the last three layers of LeNet are fully connected to the layer before and the layer after if exists, creating the final prediction for 10 classes.

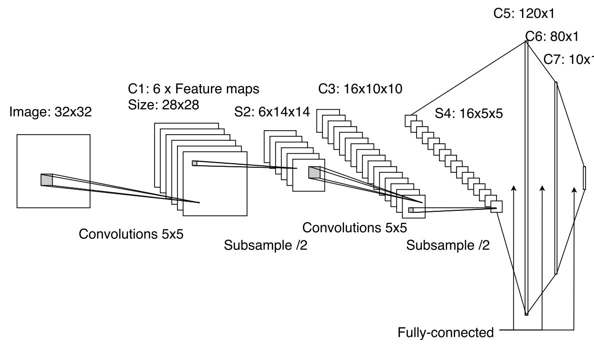

Figure 8: An example convolutional neural network, LeNet-5[18]

The bottom layers in the convolutional layer stack have smaller receptive fields while the top layers have larger receptive fields. A small receptive field means that the filter have access to information only in a local sub-region of the image while a large receptive field can convey more global information. This trend of varying pattern responses from local to global, from simple to complex for stacked convolutional layers is a reflect of emulating animals visual cortex. In cat's visual cortex[15], two basic cell types of visual cortex have been identified: Simple cells respond maximally to specific edge-like patterns within their receptive field. Complex cells have larger receptive fields and are locally invariant to the exact position of the pattern. The shallower convolutional layers play a similar functionlity as simple cells while the deeper layers maps are similar to complex cells.

The main benefit of CNN compared to the traditional multilayer perceptron is that it is easier to optimize due to the local connectivy pattern of convolutional layers and spatial weights sharing. Because of the design choice of convolutional neurons and maximum pooling, translation invariance as well as scaling invariance and distortion invariance to some extent are achievable for convolutional neural networks.[18] Different from the traditional hand-crafted features, learnable convolutional features normally generalize well and can achieve better performance for dataset with a complex input distribution.[17] By increasing the number of convolution layers and number of filters in each layer, one can create CNN models with high capacity, meaning a large space of representable functions. This can be beneficial for datasets of immense complexity, for example, ILSVRC[32], Microsoft COCO[23], as long as there are sufficient training samples with an appropriate optimization strategy.

A.2 Semantic image segmentation

Semantic image segmentation is to segment images into semantically meaningful partitions, a.k.a.,*segments*. It can be operated as classifying pixels into the corresponding pre-defined categories. The success of CNN models on object classification tasks can be extended for semantic image segmentation tasks.[25] As we discussed in the previous session, convolutional layers can extract hierarchical features, which from low-level to high-level encode information from local to global. In contrast to object classification tasks, which normally only need global information to resolve semantics, segmentation tasks also require local information to resolve locations. One of the primary difficulties of applying CNN model to segmentation tasks is how to combine global information and local information to solve semantics and locations altogether. Long et al.[25] proposed a so-called skip architecture to aggregate information from the local low-level features in the hierarchy with global information from the high-level features. The low-level features are fine, presenting appearances and the high-level features are coarse, revealing semantics. By combining them together, it becomes possible to create accurate and detailed segmentation.

Figure 9 shows the architecture of so-called fully convolutional networks (FCN) by Long et al. (2015).

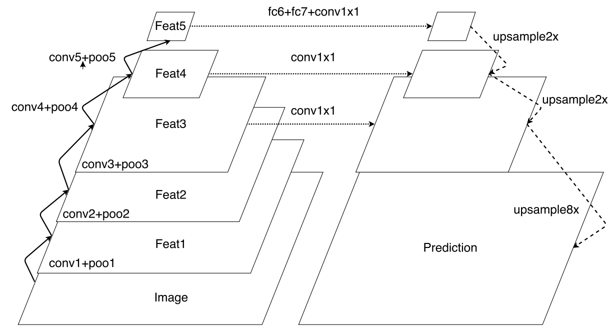

Figure 9: Fully convolutional network (FCN) by Long et al. (2015) [25].

A.3 Transferring convolutional neural nets

B Deep Learning with Label Noise

B.1 Learning in the presence of label noise

NNAR, NAR, NCAR

Such a noise model is called noisy not at random (NNAR) [8] because the noise depends on not only the true label y but also the inputs x.

B.2 Deep learning models robust to label noise

C Supportive information

C.1 An 8-layer Convolutional neural network

C.2 Evaluation metrics

$$\text{accuracy} = \frac{\text{true pos. + true neg.}}{\text{true pos. + false pos. + true neg. + false neg.}}$$

$$\text{precision} = \frac{\text{true pos.}}{\text{true pos. + false pos.}}$$

$$\text{recall} = \frac{\text{true pos.}}{\text{true pos. + false neg.}}$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

layer name	output size	8-layer
conv1	16 × 16	3 × 3, 32, LeakyReLU(0.2)
		3 × 3, 32, LeakyReLU(0.2)
		2 × 2 max pool, dropout(0.2)
conv2	8 × 8	3 × 3, 64, LeakyReLU(0.2)
		3 × 3, 64, LeakyReLU(0.2)
		2 × 2 max pool, dropout(0.2)
conv3	4 × 4	3 × 3, 128, LeakyReLU(0.2)
		3 × 3, 128, LeakyReLU(0.2)
		2 × 2 max pool, dropout(0.2)
fc	1 × 1	flatten, 512-d fc, ReLU, dropout(0.5)
		11-d fc, softmax
Parameters		1,341,739

Table 4: 8-layer Convolutional Neural Networks used for the CIFAR dataset classification.

$$\mathrm{IU} = \frac{\text{true pos.}}{\text{true pos.} + \text{false pos.} + \text{false neg.}}$$

D Additional Results

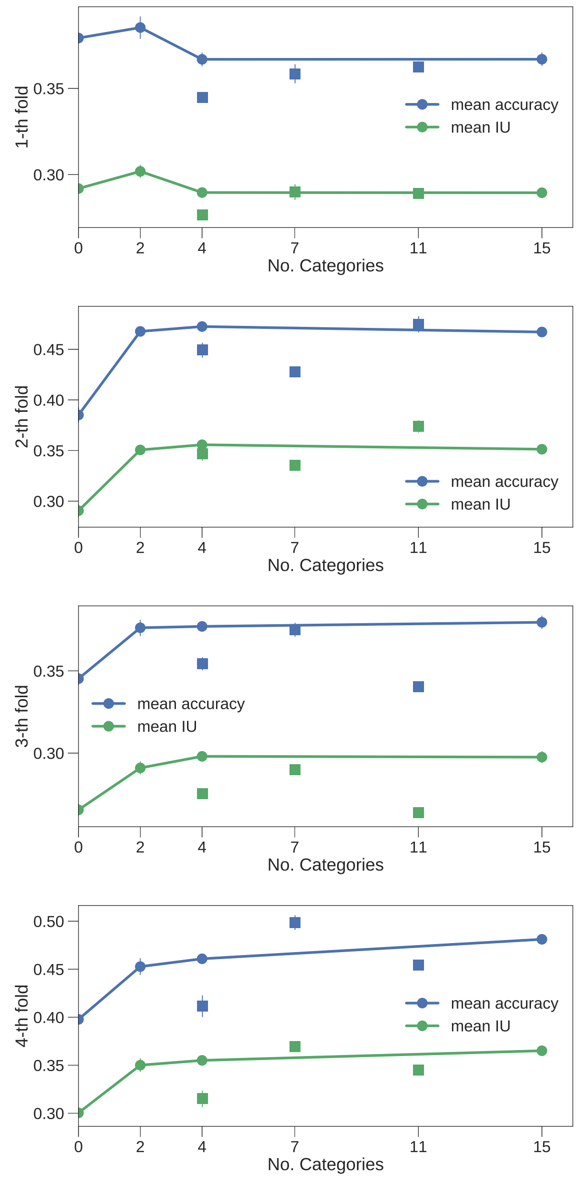

Figure 10: The influence of categorizing classes on test performance of fine-tuned models for each fold, addition to Figure 3.