

Learn transferable features for semantic image segmentation in the presence of label noise

First Author
Institution1
Institution1 address
firstauthor@i1.org

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

August 6, 2017

Abstract

The ABSTRACT HERE

1 Introduction

Why noisy labels? This paragraph should discuss the ubiquity of label noise and difficulties of collecting perfect annotations.

The recent success of deep neural networks benefits from the availability of large-scale supervised datasets such as [26, 6, 21, 17]. These datasets allow researchers to develop deep neural network models for object recognition[28], object detection[8], semantic image segmentation[18, 34] and other applications assuming the existence of perfect ground-truth segmentation. However, collecting well-annotated datasets on a large scale can be tremendously expensive and time-consuming in general. When annotators work on their tasks, it is natural for them to make mistakes as a result of lack of expertise, inherent ambiguity of tasks or unconscious bias. Enormous efforts were made in various manners to ensure the correctness of annotations according to reports in [26, 17]. Saving efforts made for correctness will result in a noisy dataset but also potentially more annotated images. Trade-offs need to be made between the impact of by label noise and the gain of a larger dataset. For particular tasks, there exist freely available labels as alternatives to manual annotations. But these labels are often noisy owing to the way they were created. For example, one can use digital maps,

like OpenStreetMap, to segment aerial images. These segmentations constructed from maps suffer from the incomplete annotation as well as registration problems.[20] Besides, Pl@ntNet¹, a crowdsourcing platform, provide millions of images of plants and corresponding labels which may or may not be correct. Strong supervision is therefore required to correct the mistakes, for example double-checking the annotations over and over again and ensembling opinions from multiple annotators. It is therefore sometimes inevitable for deep neural network models to accept the existence of noisy annotations.

Why feature's noise robustness? This paragraph should discuss 1. CNNs arch: hierarchical feature extraction and task-specific layers 2. CNNs based models benefit from transferability of hierarchical feature

Convolutional neural networks (CNNs) based models often contain two principal components: a stack of convolutional layers to extract hierarchical features and a few task-specific layers to fit the specific learning objectives.² The convolutional layers were proved “transferable” not only to another dataset[32] but also to another

¹<https://identify.plantnet-project.org/>

²M: As a non-expert it is rather unclear to me how to identify or design generic layers and task-specific ones, or what it actually means or why it is important to make this distinction. In addition, I wonder to what extent end-to-end training actually turns generic layers into task-specific ones. J: I think the idea to make this distinction here is to highlight that hierarchical features are reusable for many different applications and tasks but task-specific ones are not as they may have different number of neurons due to the different number of classes, or have even completely different architectures because the learning objectives are different for different app/tasks.

application[8, 18]. This feature transferability allows reusing the convolutional features for tasks and applications different from which they were originally trained with in a transfer learning scenario, and it helps to achieve better performance when there are only limited number of training samples[18].

This paragraph should introduce the idea of studying the impact of annotation noise on feature transferability. In cases where only a clean but small dataset is available, an extra noisy but large dataset in a similar domain might be helpful as it can be used to pre-train the convolutional features of the CNNs model. Previous studies[29, 23] have reported a negative impact of label noises on classification performance, but not yet on convolutional feature transferability. In general, optimal classification performance on test set often indicates that the extracted features are also optimal, whereas suboptimal classification performance does not necessarily reflect the convolutional features are also suboptimal, especially concerning feature transferability. Feature transferability describes the *generality of features*, i.e., the category-independence of features. Low-level features were proved to be less dependent to categories and thus more transferable to new tasks than high-level features. [32] We experimented in Section 3 that how much label noises interfere the transferability of convolutional features.

Narrow the problem of discussion down to Segmentation In this paper, we considered three types noises that happen to semantic image segmentation. These segmentation noises are summarized in Section 3.1. We chose to study segmentation errors because

1. It is more difficult to correct noisy segmentation than to correct classification noises for object recognition;
2. Semantic segmentation can be treated as pixel-wise classification so that existing methods to compensate label noise can be applied with corresponding assumptions when necessary.

Table of contents

In the next section, we summarized related works in areas of transfer learning and learning with noisy labels for deep learning. In Section 3 we formulated the segmentation errors into three categories, misannotation, misclassification and incomplete annotation, and tested feature transferability against them separately. In Section 4 we con-

nected training with inexhaustive annotations to Positive and Unlabeled Learning. We tested our hypothesis in Section 5, studying whether the misannotation and misclassification had an impact on learning “transferable” features.

2 Related work

Transfer Learning *transfer learning* We sometimes have a learning task in one domain of interest, but we only have sufficient training data in another domain which does not share a feature space with the domain of interest. Transfer learning arises in this scenario to transfer knowledge from one domain to another and to improve the performance of learning by avoiding much expensive data-labeling efforts.[22] Weights of convolutional neural networks (CNNs) show outstanding transferability to another task. For example, weights trained on ImageNet images to perform image classification were shown successfully transferred to new categories and new learning problems[8, 18, 27]. Better performance were achieved for these tasks by using ImageNet pre-trained CNNs as initialization than training full model from scratch. Yosinski et al. discovered that feature transferability is negatively affected by the specialization of higher layer neurons and optimization difficulties caused by breaking co-adapted neurons. Their experiments showed that low-level features, which are less dependent to particular categories, are more transferable than high-level features. **TODO:** Relations to our work. We studied if feature transferability is negatively affected by the presence of label noises.

Unsupervised pre-training Apart from supervised pre-training, one can also obtain pre-trained features in an unsupervised or a semi-supervised way. The most common method is to train a generative model with either *auto-encoder* variants or *deep belief networks*. Vincent et al.[30] trained multiple levels of representation robust to the corrupted inputs with stacked denoising auto-encoders. Masci et al.[19] presented a stacked convolutional auto-encoder unsupervised pre-training for hierarchical feature extraction. Hinton et al.[12] proposed a greedy learning algorithm to train *deep belief nets* one layer at a time to train hierarchical features. Lee et al.[14] presented a *convolutional deep belief network*, to

learn hierarchical convolutional representations. A few studies[5, 4, 1] highlighted the advantage of unsupervised pre-training compared to the random initialization, connecting unsupervised pre-training to a norm of regularization and a method that help disentangle the sample variations. However, better random initialization strategies, for example, xavier initialization[9] and its variants, have shortened the gap between unsupervised pre-training and random initialization. Using unsupervised pre-training or not now becomes a tradeoff between the time and resources invested and the performance gain. Unsupervised deep representation learning is in general not comparable to supervised representation learning especially when large scale dataset is available. A proper method to learn features in the presence of label noise should at least outperform unsupervised pre-training because noisy information is still better than no information.

Deep Learning with Noisy Labels A few studies[29, 23] investigated the impact of label noise on classification performance with convolutional neural networks assuming the labels were randomly transited from one to another given the probabilities fall in a transition matrix. They found a significant decrease in classification performance along with the increase of false label proportion when the total number of examples is fixed. They then proposed methods to handle this label noise at random (NAR)[7] situation by either introducing a linear noise layer on top of the output layer[29] or correcting the loss functions with an estimation of the noise transition matrix[23]. Xiao et al.[31] integrated a probabilistic graphic model to an end-to-end deep learning system to train predicting class labels, either correct or wrong, as well as to correct the wrong labels. Reed & Lee[24] proposed an empirical way of taking into account the *perceptual consistency* for large-scale object recognition and detection when incomplete and noisy labels exist by introducing a bootstrapping modification to the negative log-likelihood, in either a “Hard” or a “soft” favor.

Noise robustness In contrast to the works above, Rolnick et al.[25] argued that deep neural networks can learn robustly from the noisy dataset as long as an appropriate hyper parameters choice was made. They studied instead of replacing the correct labels with noisy labels but diluting correct labels with noisy labels to support their argu-

ment. They then concluded sufficiently large training set is of more importance than lower the level of noise. This work is closely related to our work in Section 3, except that we focus on the label noise robustness regarding the feature transferability instead of the classification performance. Additionally, most of these studies focus on the classification problems, whereas our work inclined more to the semantic segmentation problem.

Positive and Unlabeled Learning If we consider the in-exhaustive annotation issue only, i.e., only a proportion of the target instances were annotated, the problem becomes similar to a so-called *positive and unlabelled learning* (PU learning) setup[16]. In the positive and unlabeled learning setup, the training dataset has two sets of examples: the *positive (P) set*, contained only positive examples, and the *unlabeled (U) set*, contained a mix of positive or negative examples. If we categorize the pixels into either *foreground pixels* or *background pixels*, the correctly annotated instances form the positive set, and the unannotated instances are mixed with the background pixels, forming an unlabeled set. The previous studies about PU learning mainly focus on the binary classification for linear-separable problems[3, 15], whereas we showed in Section 4 that it is possible to train deep neural networks for multiple classes with only “positive” and unlabeled examples.

3 Noise robustness of feature transferability

From feature generality to feature robustness

Convolutional neural networks for images are believed to extract a rich hierarchy of image features. Some neurons capture concepts such as people and text while some units capture texture and material properties, such as dot arrays and specular reflections[8]. The conceptual features are specific for particular categories and the textural features can be shared by various categories. Furthermore, low-level features were found showing less specialization than the high-level features. By training a CNN with half of the ImageNet categories and transfer features to the other half, Yosinski et al.[32] found feature transferability decreases from the bottom layers to the top

layers. The low level features, especially features of the first two layers, presented outstanding transferability between categories with far distance, for example natural categories and man-made classes as reported in [32]. Another evidence for this is that the first convolutional features for images are often observed as Gabor filters or color blobs even training with different datasets and different objectives[33, 14, 13, 27]. Given the evidence that low-level features can be category independent and shared by categories far from each other, we wondered if they are robust to learn with noisy training labels. More specifically speaking, the research question concerned in this section is:

1. Is transferability of convolution neural networks negatively influenced by the label noises?

Table of contents We formulated the learning task and annotation errors of interest in Section 3.1 and described the experiment setups in Section 3.2 to learn how transferable the features were when they were trained with a noisy dataset.

3.1 Problem Formulation

Semantic Segmentation *Segmentation is per-pixel classification* The goal of semantic segmentation tasks is to segment images into semantically meaningful partitions, a.k.a., *segments*. These segments are *target* if they depict instances of pre-defined object categories or *non-target* otherwise. A common interpretation for semantic image segmentation tasks with CNNs is the pixel-wise classification model: Given an image x , a segmenting model $f : R^{h \times w \times c} \rightarrow R^{h \times w}$ predicts a label for each pixel and predict a label map \hat{y} that has the same size as x . h, w are image height and weight respectively, and c is the number of image channels. The corresponding annotation y consist of labels for each pixel. Supposing there are K predefined categories, the label of pixel ij

$$y_{ij} = \begin{cases} k \in [1, K], & \text{for target pixels} \\ 0, & \text{for non-target pixels} \end{cases}$$

where $i \in [1, h], j \in [1, w]$.

Label corruption model *How noises synthesized* A straightforward way to model noisy labels is to corrupt

true labels with a corruption model. The corruption model describes the probability of observed label map conditioning on the the true label map y , the image x and label errorness e :

$$p(\tilde{y}|x, y, e)$$

where the occurrence of errors e for pixels depends on the inputs x and true labels y . Such a noise model is called noisy not at random (NNAR) [7] because the noise depends on not only the true label y but also the inputs x . Given a corruption model and true annotations, one can synthesized the corresponding noisy annotations stochastically.

Clarity for noise considered. In our works, we considered three types of errors for a semantic segmentation problem, *misannotation*, *misclassification* and *inexhaustive annotation*, and modified the NNAR model accordingly. Note that all these label errors apply to the whole segment instead of to individual pixels. That is pixels for the same segments will have the same true labels and observed labels. We excluded segmenting errors such as imprecise boundaries, oversegmenting or undersegmenting the objects from study because they are more difficulty to synthesize than the preceding classification errors. It would be of more value to use a real dataset for such errors than to synthesize.

Misannotation Misannotation denotes the wrongly segmented objects for categories that are semantically meaningful but are not predefined. For example, a toy dog can be misannotated as a dog, assuming that “dog” is predefined and “toy dog” is not. In the presence of misannotation, pixel labels of segment S are transited from 0 to k with probability

$$p(\tilde{y}_{ij} = k | x, y_{ij} = 0), i, j \in S$$

The dependence of observed pixel labels \tilde{y}_{ij} on the original image x interpret the premise that misannotation would only happen to semantically meaningful segments in an image. It is natural to include this premise because semantically meaningless partitions of an image are less likely to be segmented by an annotator. However, it is difficult to interpret the above probability in practice because it is conditioning on the semantic meaning of x . We can synthesize misannotation errors with a perfectly annotated training set by selecting part of the cat-

egories as non-target categories and instances for these categories are misannotated stochastically with probability $p(y_{ij} = k | y_{ij} = 0)$. By doing this, we can get rid of interpreting the pixels semantic meaning in probability.

Misclassification Different from misannotation, misclassification error means labels were misclassified between pre-defined categories. For example, cats may be misclassified as dogs occasionally if both “cat” and “dog” are target classes. In the presence of misclassification, pixel labels of segment S are transited from k to j stochastically with probability:

$$p(y_{ij} = j | y_{ij} = k), i, j \in S, k, j \in [1, K]$$

We assumed misclassification is independent of the exact shape and appearance of the objects so that the transition probability can be independent of the inputs x . This model is often called *noisy at random* [7] because the noisy labels \tilde{y} depends on the true label y but not the inputs x . This assumption does not hold in every cases of practice, for example, some instances can be more likely to be misclassified due to its ambiguity in shapes or appearances. But the difficulty of modeling such ambiguity lead us to simply assume input independences. Given the class transition probabilities, one can easily synthesize noisy annotations including misclassification errors given a well-annotated segmentation dataset.

Inexhaustive annotation Inexhaustive annotation denotes that there exists unsegmented instances for pre-defined categories. Pixels of an unannotated object S have labels flipped from k to 0 stochastically with probability:

$$p(y_{ij} = 0 | y_{ij} = k), i, j \in S, k \in [1, K]$$

Again we assumed inexhaustive annotations to be noisy at random (NAR).

3.2 Synthesized datasets

Experiment setup In order to investigate the influence of misannotation, misclassification and inexhaustive annotation on feature transferability respectively, we set up experiments with a perfectly annotated dataset, PASCAL VOC2011[6]. 15 out of 20 categories 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 Fully Convolutional Networks with AlexNet (FCN-AlexNet) models 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 model. The fine-tuned models were evaluated by the mean intersection over union ratio (mean IU) achieved with the test set of the fine-tuning dataset. The performance of fine-tuning model comparing to the random-initialized model indicates transferability of the pre-trained weights. In order to avoid the influence of categories-splitting choice, the 20 categories of VOC2011 were divided equally into four folds and each fold was studied separately. The exact partitions of categories are listed in Table 1.

Experiment details The training dataset was enriched with extra manual segmentations by Hariharan et al.[11] To keep the segmentation task simple, we used only single-object images for both pre-training dataset and fine-tuning dataset, resulting in 4000 training images of 20 categories in total. 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 its relatively short training time and the existence of an ImageNet model for AlexNet. Only weights of convolutional filters in AlexNet were transferred from the pre-training phase to fine-tuning phase and the other layers were random initialized, with Xavier Initialization. The ImageNet model and completely random weight initialization are the upper bound and lower bound respectively for various pre-trained weights summarized in Table 1. The default hyperparameters of FCN-AlexNet[18] were kept unchanged. Training run 240,000 iterations for pre-training phase, and 12,000 iterations for fine-tuning phase.

What Table 1 tell us. How annotation errors are synthesized; What are the results compared to noise-free counterparts; How synthesization different from reality; Misannotation, misclassification and inexhaustive annotation were synthesized and studied separately with stochastically corruption of the well-annotated pre-training dataset as described in Section 3.1. The exact label transition probabilities are summarized in the following paragraphs.

Misannotation As discussed in Section 3.1, misannotation errors in this experiment were synthesized by se-

lecting one category as the target category and all the other 14 categories in the pretraining dataset became non-target. In the presence of misannotation errors, instances for the non-target categories were misannotated as if they were also from the target category with probability of $p(\tilde{y}_{ij} = 1 | y_{ij} = 0) = 1$ or 0.5. The two choices of probability result in the two pre-trained models, “AllMisannotated” model and “HalfMisannotated” model, in Table 1 respectively. The noise-free case for misannotation is the datasets with only the selected target category segmented and the other 14 categories unsegmented, denoted as *SingleTarget* in Table 1. TODO results

Misclassification Misclassification errors were synthesized from the well-annotated pre-training dataset as well, as described in Section 3.1. Labels of segmented instances transited stochastically from one category to another with probability $p(\tilde{y}_{ij} = l | y_{ij} = k) = \delta(l - k)$. The resulted trained model was denoted as “AllRandomLabels” in Table 1. Similarly, if half of the objects were assigned to random labels with uniform probability, the corresponding pre-trained model is called “HalfRandomLabels” model. The noise-free counterpart of these two noisy models is the model trained with true labels, “TrueLabels”. TODO results

Inexhaustiv Train data with inexhaustive annotations were synthesized by randomly converting labels of segmented objects to 0 with probability $p(\tilde{y}_{ij} = k | y_{ij} = k) = 0.5$. TODO results

4 Positive and Unlabeled Learning

This part should explain the necessity of Positive and Unlabeled Learning setup, including the importance to make a difference between inexhaustive annot. and misclassification.

We discussed in Section 3 that misclassification and inexhaustive annotation can have negative influences on feature transferability, which is undesirable when we pre-train weights with noisy data. Methods to compensate these errors are therefore necessary to train better representation with a certain number of annotated images. In practice, misclassification error is less likely to *dorminate*, meaning that the misclassified rate is often not larger than the correct rate. By contrast, unannotated instances can be dominant for a large-scale dataset, especially when mis-

Initial Representation	mean IU (aeroplane, bicycle, bird, boat, bottle)	mean IU (bus, car, cat, chair, cow)	mean IU (dining table, dog, horse, motorbike, person)	mean IU (potted plant, sheep, sofa, train, TV)
ImageNetModel	0.42 ± 0.01	0.51 ± 0.01	0.49 ± 0.01	0.47 ± 0.01
RandomWeights	0.29 ± 0.01	0.29 ± 0.03	0.27 ± 0.01	0.30 ± 0.02
SingleTarget	0.26 ± 0.01	0.37 ± 0.03	0.27 ± 0.01	0.33 ± 0.04
AllMisannotated	0.30 ± 0.02	0.35 ± 0.01	0.29 ± 0.02	0.35 ± 0.03
HalfMisannotated	0.30 ± 0.00	0.35 ± 0.00	0.29 ± 0.00	0.35 ± 0.00
TrueLabels	0.29 ± 0.01	0.36 ± 0.01	0.29 ± 0.01	0.37 ± 0.01
AllRandomLabels	0.29 ± 0.01	0.33 ± 0.03	0.26 ± 0.01	0.28 ± 0.01
HalfRandomLabels	0.27 ± 0.01	0.33 ± 0.02	0.25 ± 0.01	0.29 ± 0.01
InexhaustiveLabels	0.26 ± 0.01	0.30 ± 0.3	0.28 ± 0.03	0.32 ± 0.02

Table 1: Performances of fine-tuned FCN-AlexNet models with different representation initializations. *ImageNetModel* represents the pre-trained ImageNet model; *RandomWeights* indicates that the weights were randomly initialized; All the other weights were first trained with the pre-training dataset in the presence or the absence of different types of label noises. Each experiment was repeated three times, the mean and the standard deviation were computed over the last five snapshots for all repetitions.

annotated objects are included for training. For example, the existence of one misannotated toy dog does not mean that all toy dogs are annotated as dogs. If misannotated objects are treated the same as correctly annotated ones as proposed in Section 3, the unannotated “toy dogs” may cause the problem of inexhaustive annotations and lead to worse fine-tuning performance.³ Therefore, we explored methods to compensate inexhaustive annotations for segmentation in this section.

Formulation *This part should formulate the problem with a probability model and make clear difference between model for synthesizing and model to fit.* When we synthesized errors in Section 3, we assumed that all label errors occurred to each segment instead of each pixel. This is helpful when we already know the However, which pixels belong to the same segment is unknown when a model is trained with noisy labels only. prediction for each pixel is made independently though the corresponding receptive fields is overlapped. Therefore, we instead assume label errors all occurred independently Reasons:

$$p(\tilde{y}_{ij} | x, y_{ij}).$$

³J: Extra experiment to support this: 50% misannotated + 50% unannotated.

Annotation	Loss	acc.	prec.	rec.	F_1
Complete	CrossEntropyU.	0.87 ± 0.01	0.88 ± 0.01	0.82 ± 0.01	0.85 ± 0.01
50%(P+N)	CrossEntropyU.	0.83 ± 0.01	0.84 ± 0.01	0.78 ± 0.01	0.80 ± 0.01
50%P+U	CrossEntropyU.	0.64 ± 0.04	0.93 ± 0.08	0.34 ± 0.02	0.44 ± 0.06
50%P+U	WeightedU.	0.78 ± 0.01	0.75 ± 0.01	0.75 ± 0.01	0.76 ± 0.01
50%P+U	ExponentialU.	0.82 ± 0.01	0.86 ± 0.01	0.73 ± 0.01	0.78 ± 0.01
50%P+U	BootstrapHard	0.74	0.81	0.60	0.67

Table 2: Image classification with positive examples partially annotated. The complete dataset contains images from CIFAR10 as the **positive** (P) set and images from CIFAR110 as the **negative** (N) set. The unannotated positive examples from P set construct the **unlabeled** (U) set together with the N set.

This is a strong assumption for practical problems ,and may lead to future works

This model is easy to apply when we have the perfect annotations and want to generate noisy labels, but it is not as easy when we observe noisy labels and want to recover the correct ones.

$$p(\tilde{y}_{ij}|y_{ij})$$

Weighted Logistic Regression *This part should discuss the linear model for observing positive conditioning on true positive and its relationship to changing the class weight.*

Exponential Loss for unlabeled examples *This part should explain why the exponential loss could perform better than the cross-entropy loss, potentially with a figure of 2D Gaussians.* The “soft” bootstrapping loss in [24] is actually equivalent to a softmax regression with *minimum entropy regularization*[10] which was originally proposed for semi-supervised learning. Minimum entropy regularization encourages the model to have a high confidence in predicting labels.

Implementation details *This paragraph should explain fade-in was introduced to avoid all-positive initial prediction;*

This part should explain the influence of the imbalanced problem and how to overcome.

Annotation	Loss	pixel acc.	mean acc.	mean IU	f.w. IU
Complete	CrossEnt.U				
50%(P+N)	CrossEnt.U				
50%P+U	CrossEnt.U				
50%P+U	WeightedU				
50%P+U	ExponentialU				
50%P+U	BootstrapHard				
50%P+U	DropoutReg.				

Table 3: Image semantic segmentation with images contain single instance only from the PASCAL VOC2011 segmentation dataset. The complete **positive** (P) set denotes the foreground instances and the **negative** (N) set consists of the background. The unannotated instances from P set construct the **unlabeled** (U) set together with the N set.

5 Results

5.1 Impact of label noise on feature transferability

6 Conclusion

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] 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.
- [4] 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.
- [5] Dumitru Erhan, Pierre-Antoine Manzagol, Yoshua Bengio, Samy Bengio, and Pascal Vincent. The difficulty of training deep architectures and the effect of unsupervised

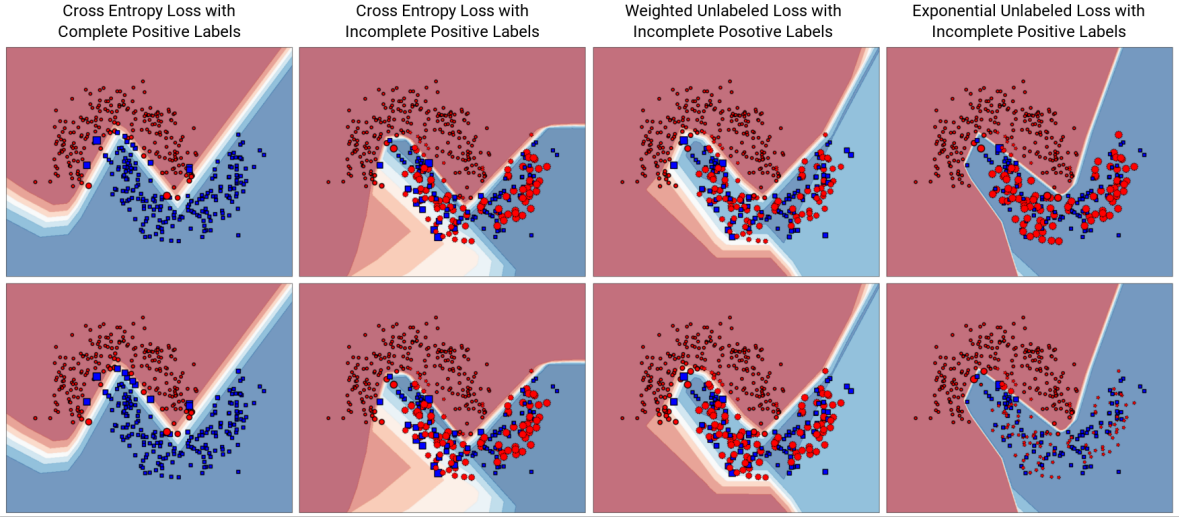


Figure 1: 2D moons dataset with non-linear separable decision boudary. Four hundreds samples per class were drawn randomly from two interleaving half circles with noises added with a minor standard deviation. A **red circle** indicates an example labelled as positive whilst a **blue square** indicates the example has a negative label. The **leftmost** figures have complete positive labels, meaning the positive and negative labels are all correct, whereas, in **the other figures** only half of the positives were correctly labelled and the rest were mixed with the negative samples. 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. The **size of the markers** in the top row denotes the per-class normalized training losses and the **size of the markers** in the bottom row the per-class normalized derivatives w.r.t the output of the last layer for the trained Multilayer Perceptron (MLP) with the different losses.

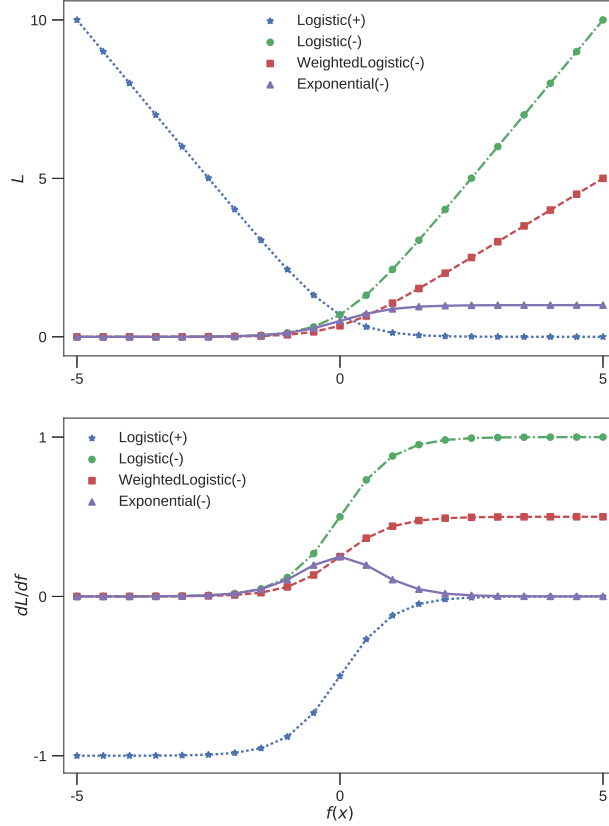


Figure 2: The Logistic Loss, Weighted Logistic Loss, Exponential Loss and their derivatives with respect to the model output.

pre-training. In *Artificial Intelligence and Statistics*, pages 153–160, 2009.

- [6] 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.
- [7] 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.
- [8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the*

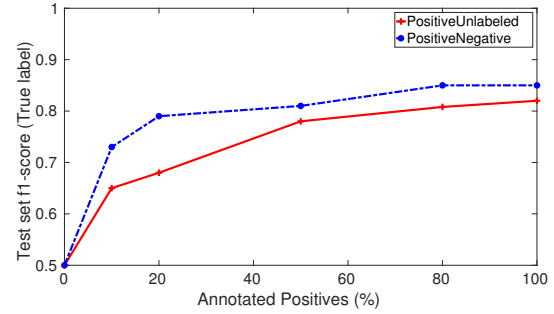


Figure 3: Varying percentage of annotated positives 10%, 20%, 50%, 80% and 100% with images from CIFAR10 as the positives and images from CIFAR110 as the negatives.

IEEE conference on computer vision and pattern recognition, pages 580–587, 2014.

- [9] 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.
- [10] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Advances in neural information processing systems*, pages 529–536, 2005.
- [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] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.
- [13] 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.
- [14] 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.
- [15] Wee Sun Lee and Bing Liu. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pages 448–455, 2003.

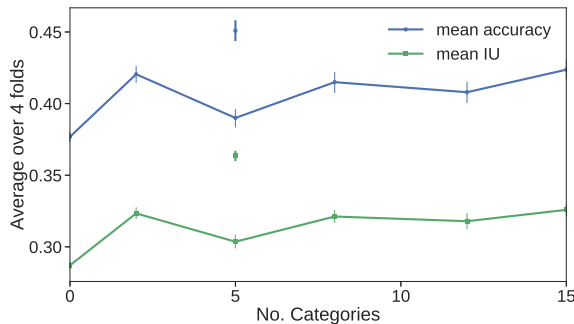


Figure 4: The influence of number of categories in the pre-training dataset on performance of fine-tuned models. Varying the number of categories while pre-training the representation and the pre-trained weights were fine-tuned to segment 5 categories from the PASCAL VOC2011 dataset

- [16] Xiao-Li Li and Bing Liu. Learning from positive and unlabeled examples with different data distributions. *Machine Learning: ECML 2005*, pages 218–229, 2005.
- [17] 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.
- [18] 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.
- [19] 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.
- [20] 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.
- [21] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Wan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 891–898, 2014.
- [22] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.
- [23] 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.
- [24] 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.
- [25] David Rolnick, Andreas Veit, Serge Belongie, and Nir Shavit. Deep learning is robust to massive label noise. *arXiv preprint arXiv:1705.10694*, 2017.
- [26] 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.
- [27] 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.
- [28] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [29] Sainbayar Sukhbaatar, Joan Bruna, Manohar Paluri, Lubomir Bourdev, and Rob Fergus. Training convolutional networks with noisy labels. *arXiv preprint arXiv:1406.2080*, 2014.
- [30] 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.
- [31] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2691–2699, 2015.
- [32] 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.

- [33] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.
- [34] 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.

A Convolutional Networks for Semantic Segmentation

The different levels of hierarchical features in CNNs are believed to play different roles in extracting information from the images. The low-level features process the local information within small neighborhood and the high-level features ensemble information from lower-level features to extract abstract information. The high-level features were found to significantly dependent on the exact categories compared to the low-level features which show extraordinary category independence.[32] Previous studies[29, 23] have shown that training with noisy labels can lead to significantly higher classification errors than training with clean labels if the total number of training samples are fixed. It is nevertheless unclear how the annotation errors would influence the learned multiple level features. We made a hypothesis that annotation errors do not necessarily lead to a “bad representation” because the “generality” of low-level features may contribute to a robustness to the errors when we transfer the learned features to a new dataset with new categories.⁴

Narrow the problem of discussion down to Segmentation

The state-of-art DCNN based semantic image segmentation models relies on transferring the pre-trained convolutional filters as well.[18] A typical method to pre-train the convolutional filters is to train a classification model with the large-scale ILSVRC dataset [26] However, this method constrains the semantic image segmentation models to have the same CNN architecture as the image classification models. The CNN design for semantic image segmentation does not necessarily follow the design of image classification architectures. The segmentation models need both global and local information to predict the category and give a fine segmentation, whereas the classification models care less about local information for object localization. For instance, the presence of the max-pooling layers enable the following convolutional filters

⁴M: We hypothesize? For the rest I find the actual hypothesis pretty vague. Firstly, you use quotation marks quite often, which doesn’t make “things” clearer. Either don’t use that and make sure that the words you use are indeed the words you like to say or expand the part between “”s and use some more sentences to really explain what you have in mind. Secondly, for me the part “contribute to a robustness to the errors” needs further explaining. You might not have to get mathematically precise here, but I don’t understand what you want to say here...

to have larger receptive fields but, at the same time, reduce the resolution of the features.⁵ Additional upsampling layers can recover the shape of the output segmentation but cannot fully recover the information dropped by subsampling. This first-pooling-and-then-upsampling pipeline can result in coarse segmentation output [2] with non-shape boundaries and blob-like shapes.⁶ Alternatively, one can also extract convolutional features with semantic segmentation tasks. But it is more difficult to collect well-annotated dataset for semantic image segmentation than for object recognition. A large number of annotated images are of great value to train sufficiently generalized representation and avoid overfitting, given the “data-hungry” nature of DCNNs.⁷ Allowing noisy annotations to exist could help significantly increase the number of annotated images and the number of training samples could compensate the impact of annotation errors. We will further discuss this in the related works.

B Deep Learning with Label Noise

C

⁵M: Is it the presence of max-pooling or the presence of subsampling that enables larger receptive fields? And what do you mean by resolution? Should low resolution always mean reduced / “thrown away” information? J: Max-pooling is a particular type of subsampling. Actually both the conv layers and the pooling layers lead to gradually larger receptive field.

⁶M: Also for this claim, I think we need a reference or something like that. Or other proof indeed... In addition, I wonder whether we can explain why this may be the case? J: Yes, I can refer to a paragraph in the intro of CRFasRNN and enrich the discussion a bit.

⁷M: For the rest a bit vague... what do we really mean by suffer? Maybe this becomes clear later in the intro...? J: Refrased. M: This is still quite unclear to me. I would interpret “data-hungry” as an NN that overtrains even with large amount of data, but one would typically try to fix this by introducing some form of regularization or just using smaller networks. All in all [as a maybe-too-stubborn non-deep learner ;-)] I could still wonder what is really the problem...

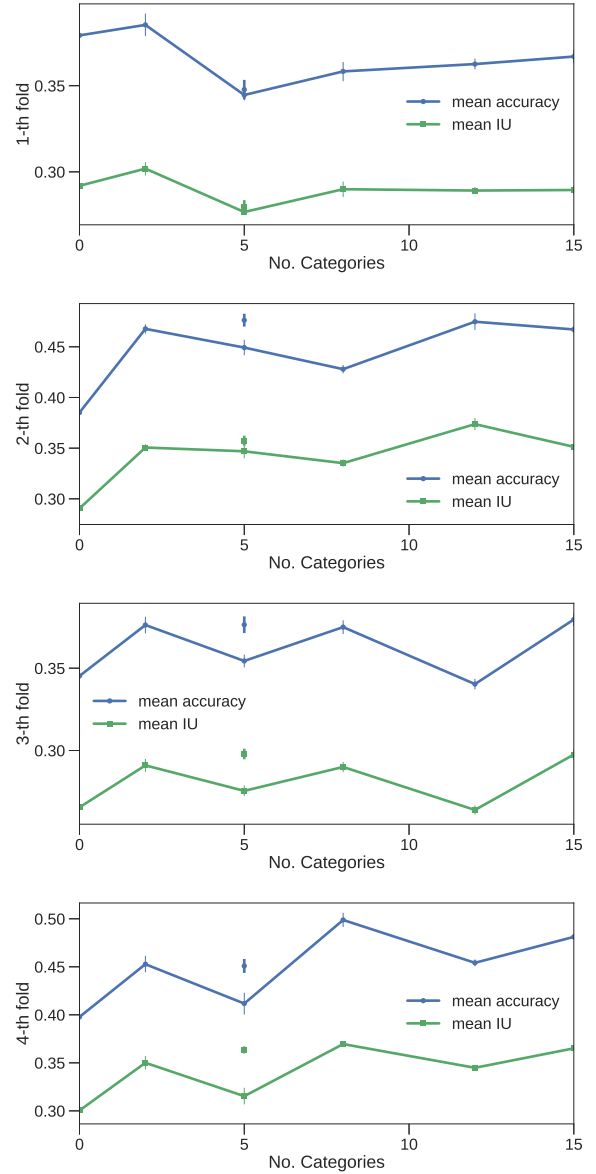


Figure 5: The influence of number of pre-training categories on performance of fine-tuned models for each fold, addition to Figure 4.