Label Refinery: Improving ImageNet Classification through Label Progression

Hessam Bagherinezhad[1,2], Maxwell Horton[1,2], Mohammad Rastegari[1,3], Ali Farhadi[1,2,3]
{hessam,max,mohammad,ali}@xnor.ai

[1]XNOR AI, [2]University of Washington, [3]Allen AI

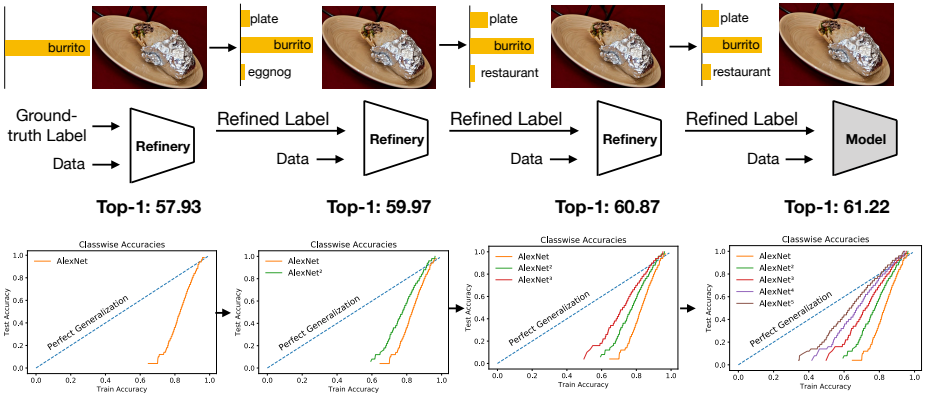


Fig. 1: Current labeling principles impose challenges for machine learning models. We introduce the *Label Refinery*, an iterative procedure to update ground truth labels using a visual model trained on the entire dataset. The Label Refinery produces soft, multi-category, dynamically-generated labels consistent with the visual signal. The training image shown is labelled with the single category "burrito". After a few iterations of label refining, the labels from which the final model is trained are informative, unambiguous, and smooth. This results in major improvements in the model accuracy during successive stages of refinement as well as improved model generalization. These plots show that as models proceed through successive stages of refinement, the gaps between train and test results and approach ideal generalization.

Abstract. Among the three main components (data, labels, and models) of any supervised learning system, data and models have been the main subjects of active research. However, studying labels and their properties has received very little attention. Current principles and paradigms of labeling impose several challenges to machine learning algorithms. Labels are often incomplete, ambiguous, and redundant. In this paper we study the effects of various properties of labels and introduce the *Label Refinery*: an iterative procedure that updates the ground truth labels after examining the entire dataset. We show significant gain using refined labels across a wide range of models. Using a Label Refinery improves the state-of-the-art top-1 accuracy of (1) AlexNet from 59.3 to 67.2, (2) MobileNet[1] from 70.6 to 73.39, (3) MobileNet$^{0.25}$ from 50.6 to 55.59, (4) VGG19 from 72.7 to 75.46, and (5) Darknet19 from 72.9 to 74.47.

Keywords: Label Refinery, Convolutional Neural Networks, Deep Learning

1 Introduction

There are three main components in the typical pipeline of supervised learning systems: the *data*, the *model*, and the *labels*. Sources of data have expanded drastically in past several years. We have observed the impact of large-scale datasets for several visual tasks. A variety of data augmentation methods [1,2,3,4] have effectively expanded these datasets and improved the performance of learning systems. Models have also been extensively studied in the literature. Recognition systems have shown improvements by increasing the depth of the architectures [5,6], introducing new activation and normalization layers [7,8], and developing optimization techniques and loss functions [9,10]. In contrast to the improvements in data and models, little effort has focused on improving labels.

Current labeling principles and practices impose specific challenges on our learning algorithms. 1) Incompleteness: A natural image of a particular category will contain other object categories as well. For example, Figure 2(a) shows an example from ImageNet that is labeled "cat" but the image contains a "ball" as well. This problem is rooted in the nature of how researchers define and collect labels, and is not unique to a specific dataset. 2) Taxonomy Dependency: Categories that are far from each other in the taxonomy structure can be very similar visually. 3) Inconsistency: To prevent overfitting, various loss functions and regularization techniques have been introduced into the training process. Data augmentation [1] is one of the most effective methods employed to prevent neural networks from memorizing the training data. Most modern state-of-the-art architectures for image classification are trained with crop-level data augmentation, in which crops of the image used for training can be as small as 8% of the area of the original image [5]. For many categories, such small crops will frequently result in patches in which the object of interest is no longer visible (Figure 2), resulting in an inconsistency with the original label.

To address the aforementioned shortcomings, we argue that several characteristics should apply to ideal labels. Labels should be *soft* to provide more coverage for co-occurring and visually-related objects. Traditional one-hot vector labels introduce challenges in the modeling stage. Labels should be *informative* of the specific image, meaning that they should not be identical for all the images in a given class. For example, an image of a "dog" that has similar appearance to a "cat" should have a different label than an image of a "dog" that has similar appearance to a "fox". This also suggest that labels should be defined at the instance-level rather than the category-level. Determining the best label for each instance may require observing the entire data to establish intra- and inter-category relations, suggesting that labels should be *collective* across the whole dataset. Labels should also be consistent with the image content when crops are taken. Therefore, labels should be *dynamic* in the sense that the label for a crop should depend on the content of the crop.

In this paper we introduce *Label Refinery*, a solution that uses a neural network model and the data to modify crop labels during training. Refining the labels while training enables us to generate soft, informative, collective, and dynamic labels. Figure 1 depicts an example of a label refinery. As models go

(a)

(b)

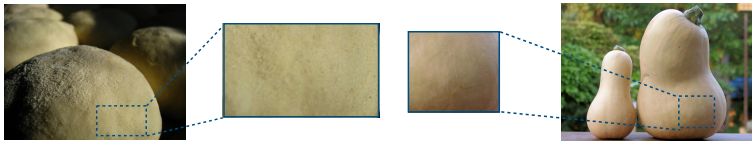
Fig. 2: Figure 2(a) shows a sample image from the "persian cat" category of ImageNet's training set. The standard technique to train modern state-of-the-art architectures is to crop patches as small as 8% area of the original image, and label them with the original image's label. This will often result in inaccurate labels for the augmented data. Figure 2(b) shows a sample crop of the original image where the "persian cat" is no longer in the crop. A trained ResNet-50 labels Figure 2(a) by "persian cat", and labels Figure 2(b) by "golf ball". We claim that using a model to generate labels for the patches results in more accurate labels and therefore more accurate models.

through the stages of the refinery labels are updated based on the previous models. This results in major improvements in the accuracy and generalization. The output of the label refinery is a set of labels from which one can learn a model. The model trained from the produced labels are much more accurate and more robust to over-fitting.

Our experiments show that Label Refining consistently improves the accuracy of object classification networks by a large margin across a variety of popular network architectures. Our improvements in Top-1 accuracy on the ImageNet validation set include: AlexNet from 59.3% to 67.2%, VGG19 from 72.7% to 75.46%, ResNet18 from 69.57% to 72.52%, ResNet50 from 75.7% to 76.5%, DarkNet19 from 72.9% to 74.47%, $\text{MobileNet}_{0.25}$ from 50.65% to 55.59%, and MobileNet_{1} from 70.6% to 73.39%. Collective and dynamic labels enable standard models to generalize better, resulting in significant improvements in image classification. Figure 1 Plots the train versus test accuracies as models go through the label refinery procedure. The gap between train and test accuracies is getting smaller and closer to an ideal generalization.

We further demonstrate that a trained model can serve as a Label Refinery for another model of the same architecture. For example, we iterate through several successions of training a new AlexNet model by using the previously trained AlexNet model as a Label Refiner. Our results show major improvements (from 59.3% to 61.2%) on using AlexNet to refine labels for another AlexNet. Note that the final AlexNet has not seen the actual ground-truth labels in the past few stages. The final AlexNet models demonstrate greatly reduced overfitting compared to the original models (Figure 4(a) and Figure 5). We also experiment with using a model of one architecture as a Label Refiner for a model of another architecture. Further, we have also shown that adversarially modifying image examples improves the accuracy when using label refinery.

Our contributions include: (1) introducing the Label Refinery for crop-level label augmentation, (2) improving state-of-the-art accuracy on ImageNet for



(a) (b)

Fig. 3: Sample training examples from "dough" and "butternut squash" categories of ImageNet. While the two sample images are visually distinctive, their random crops are quiet similar. A trained ResNet-50 labels both cropped patches softly over categories of "dough", "butternut squash", "burrito", "french loaf", and "spaghetti squash". We claim that labelling the crops softly by a trained model makes the training of the same model more stable, and therefore results in more accurate models.

a variety of existing architectures, (3) demonstrating the ability of a network to improve accuracy by training from labels generated by another network of the same architecture, and (4) generating adversarial examples to improve the performance of the Label Refinery method.

2 Related Work

Label Smoothing and Regularization: Softening labels has been used to improve generalization. [11] uniformly redistributes 10% of the weight from the ground-truth label to other classes to help regularize during training. DisturbLabel [10] replaces some of the labels in a training batch with random labels. This helps regularize training by preventing overfitting to ground-truth labels. [12] augments noisy labels using other models to improve label consistency. [13] introduces a notion of local distributional smoothness in model outputs based on the smoothness of the model's outputs when inputs are perturbed. The smoothness criterion is enforced with the purpose of regularizing models. The work of [14] explores penalizing networks by regularizing the entropy of the model outputs. Unlike our method, these approaches can not address the inconsistency of the labels.

Incorporating Taxonomy: Several methods have explored using taxonomy to improve label and model quality. [15] uses cross-category relationships from knowledge graphs to mitigate the issues caused by noisy labels. [16] designs a hierarchical loss to reduce the penalty for predictions that are close in taxonomy to the ground-truth. [17] investigates learning multi-label classification with missing labels. They incorporate instance-level information as well as semantic hierarchies in their solution. Incorporating taxonomic information directly into the model's architecture is explored in [18]. [19] uses the output of existing binary classifiers to address the problem of training models on single-label examples that contain multiple training categories. These methods fail to address the incompleteness of the labels. Instead of directly using taxonomy, our model collectively infer the visual relations between categories to impose these knowledge into the training while capturing a complete description of the image.

Data Augmentation: To preserve generalization, several data augmentations such as cropping, rotating, and flipping input images have been applied in training models [8,6,5,20]. [2] proposes data warping and synthetic over-sampling to generate additional training data. [1] and [4] explore using GANs to generate training examples. Most of such augmentation techniques further confuse the model with inconsistent labels. For example, a random crop of an image might not contain the main object the that image We propose augmenting the labels alongside with the data by refining them during training when augmenting the data.

Teacher-Student Training: Using another network or an ensemble of multiple networks as a teacher model to train a student model has been explored in [21,22,23,24,25,26]. [22] explores training a shallow student network from a deeper teacher network. A teacher model is used in [26,24] to train a compressed student network. Most similar to our work is [25], where they introduce distillation loss for training a model from an ensemble of its own. We show that Label Refinery must be done at the crop level, it benefits from being performed iteratively, and models benefit by learning off of the labels generated by the exact same model.

3 Label Refinery

Previous works have shown that data augmentation using cropping significantly improves the performance of classification models [8,27]. Given a dataset $\mathcal{D} = \{(X_i, Y_i)\}$, we can formalize data augmentation by defining a new dataset $\tilde{\mathcal{D}} = \{(f(X_i), Y_i)\}$, where f is a stochastic function that generates crops on-the-fly for the image X_i. The image labels assigned to the augmented crops are often not accurate (Figure 2 and Figure 3). We address this problem by passing the dataset through multiple Label Refiners. The first Label Refinery network C_{θ_1} is trained over the dataset $\tilde{\mathcal{D}}$ with the inaccurate crop labels. The second Label Refinery network C_{θ_2} is trained over the same set of images, but uses labels generated by C_{θ_1}. More formally, we can view this procedure as training C_{θ_2} on a new augmented dataset $\tilde{\mathcal{D}}_1 = \{(f(X_i), C_{\theta_1}(f(X_i)))\}$. Once C_{θ_2} is trained, we can similarly use it to train a subsequent network C_{θ_3}.

We train the first Label Refinery network C_{θ_1} using the cross-entropy loss against the image-level ground-truth labels. We train all subsequent Label Refinery models C_{θ_t} for $t > 1$ by minimizing the KL-divergence between its output and the soft label generated by the previous Label Refinery $C_{\theta_{t-1}}$. Letting $p_c^t(z) \triangleq C_{\theta_t}(z)[c]$ be the probability assigned to class c in the output of model C_{θ_t} on some crop z, our loss function for training model C_{θ_t} is:

$$L_t(f(X_i)) = -\sum_c p_c^{t-1}(f(X_i)) \log\left(\frac{p_c^t(f(X_i))}{p_c^{t-1}(f(X_i))}\right) \tag{1}$$

$$= -\sum_c p_c^{t-1}(f(X_i)) \log p_c^t(f(X_i)) + \sum_c p_c^{t-1}(f(X_i)) \log p_c^{t-1}(f(X_i))$$

The second term is the entropy of the soft labels, and is constant with respect to C_{θ_t}. We can remove it and instead minimize the cross entropy loss:

$$\tilde{L}_t(f(X_i)) = -\sum_c p_c^{t-1}(f(X_i)) \log p_c^t(f(X_i)) \tag{2}$$

Note that training C_{θ_1} using cross entropy loss can be viewed as a special case of our sequential training method using KL-divergence in which C_{θ_1} is trained from the original image-level labels. It's worth emphasizing that the subsequent models do not see the original ground truth labels Y_i. The information in the original labels is propagated by the sequence of Label Refinery networks.

If any of the Label Refinery networks have Batch Normalization [7], we put them in training mode even at the label generation step. That is, their effective mean and standard deviation to be computed from the current training batch as opposed to the saved running mean and running variance. We have observed that this results in more accurate labels and, therefore, more accurate models. We believe that this is due to the fact that the Label Refinery has been trained with the Batch Normalization layers in the training mode. Hence it produces more accurate labels *for the training set* if it's in the same mode.

It is possible to use the same network architecture for some (or all) of the Label Refinery networks in the sequence. We have empirically observed that the dataset labels improve iteratively even when the same network architecture is used multiple times (Section 4). This is because the same Label Refinery network trained on the new refined dataset becomes more accurate that its previous versions over each pass. Thus, subsequent networks are trained with more accurate labels.

The accuracy of a trained model heavily depends on the consistency of the labels provided to it during training. Unfortunately, assessing the quality of crop labels quantitatively is not possible because there crop level labels are not provided. Asking human annotators to evaluate individual crops is infeasible both due to the number of possible crops and due to the difficulty of evaluating soft labels to a large number of categories for a crop in which there may not be a single main object. We can use a network's validation set accuracy as a measure of its ability to produce correct labels for crops. Intuitively, this measurement serves as an indication of the quality of a Label Refinery network. However, we observe that models with higher validation accuracy do not always produce better crop labels if the model with higher validation accuracy is severely overfit to the training set. Intuitively, this is because the model will reproduce the ground-truth image labels for training set images. We explore this more in Section 4.1.

One popular way to augment ImageNet data is to crop patches as small as 8% of the area of the image [27]. In the presence of such aggressive data augmentation, the original image label is often very inaccurate for the given crop. Whereas traditional methods only augment the image input data through cropping, we additionally augment the labels using Label Refinery networks to produce labels for the crops. Smaller networks such as MobileNet [28] usually aren't trained with such small crops. Yet, we observe that such networks can

benefit from small crops if a Label Refinery is used. This demonstrates that a primary cause in accuracy degradation of such networks is inaccurate labels on small crops.

3.1 Adversarial Jittering

Using a Label Refinery network allows us to generate labels for any set of images. Our training dataset $\tilde{\mathcal{D}}_t = \{(f(X_i), C_{\theta_t}(f(X_i)))\}$ depends only on the input images X_i, and labels are generated on-the-fly by the Refinery network C_{θ_t}. This means that we are no longer limited to using images in the training set $\mathcal{D}$. We could use another unlabeled image dataset as a source of X_i. We could even use synthetic images. We experiment with using the Label Refinery in conjunction with the network being trained in order to generate adversarial examples on which the two networks disagree.

Let $C_{\theta_{t-1}}$ and C_{θ_t} be two of the networks in a sequence of Label Refinery networks. Given a crop $f(X_i)$, we define $\alpha_t(f(X_i))$ to be a modification of $f(X_i)$ for which $C_{\theta_{t-1}}$ and C_{θ_t} output different probability distributions. Following the practice of [29] for generating adversarial examples, we define α_t as

$$\alpha_t(X) = X + \eta \frac{\partial L_t}{\partial X} \tag{3}$$

, where L_t is the KL-divergence loss defined in Equation 1. This update performs one step of gradient ascent in the direction of increasing the KL-divergance loss. In other words, the input is modified to exacerbate the discrepancy between the output probability distributions. In order to prevent the model being trained from becoming confused by the unnatural inputs $\alpha_t(f(X_i))$, we batch the adversarial examples with their corresponding natural crops $f(X_i)$.

4 Experiments

We evaluate the effect of label refining for a variety of network architectures on the standard ImageNet, ILSRVC2012 [30] classification challenge. We first explore the effect of label refining when the Label Refinery network architecture is identical to the architecture of the network being trained. We then evaluate the effect of label refining when the Label Refinery uses a more accurate network architecture. Finally, we present some ablation studies and analysis to investigate the source of the improvements. Note that all experiments are done with a single model over a single validation crop.

Implementation Details: All models are trained using PyTorch [31] on 4 GPUs for 200 epochs to ensure convergence. The learning rate is constant for the first 140 epochs. It is divided by 10 after epoch 140 and again divided by 10 after epoch 170. We use an initial learning rate of 0.01 to train AlexNet and an initial learning rate of 0.1 for all other networks. We use image cropping and horizontal flipping to augment the training set. When cropping, we follow the data augmentation practice of [27] in which the crop areas are chosen uniformly

Model	Top-1	Top-5
AlexNet	57.93	79.41
AlexNet^2	59.97	81.44
AlexNet^3	60.87	82.13
AlexNet^4	61.22	**82.56**
AlexNet^5	**61.37**	**82.56**

Model	Top-1	Top-5
ResNet50	75.7	92.81
ResNet50^2	**76.5**	**93.12**

Model	Top-1	Top-5
MobileNet	68.51	88.13
MobileNet^2	**69.52**	**88.7**

Model	Top-1	Top-5
VGG16	70.1	88.54
VGG16^2	71.85	90.07
VGG16^3	**72.49**	**90.76**

Model	Top-1	Top-5
VGG19	71.39	89.44
VGG19^2	72.66	90.75
VGG19^3	**73.32**	**91.30**

Model	Top-1	Top-5
Darknet19	70.6	89.13
Darknet19^2	72.74	90.73
Darknet19^3	**73.01**	**90.92**

Table 1: Self-Refining results on the ImageNet 2012 validation set. Each model is trained using labels refined by the model right above it. That is, AlexNet^3 is trained by the labels refined by AlexNet^2, and AlexNet^2 is trained by the labels refined by AlexNet. The first row models are trained using the image level ground-truth labels.

from 8% to 100% of the area of the image. We use a batch size of 256 for all models except the MobileNet variations, for which we use batch size of 512. Except for adversarial inputs experiments, we train models from refined labels starting from a random initialization. Our source code is available at `http://github.com/hessamb/label-refinery`.

Self-Refinement: We first explore using a Label Refinery to train another network with the same architecture. Table 1 shows the results for self-refinement on various architectures. Each row represents a randomly-initialized instance of the network architecture trained with labels refined by the model directly one row above it in the table. All six network architectures improve their accuracy through self-refinement. For AlexNet the self-refining process must be repeated 4 times before convergence, whereas MobileNet and ResNet-50 converge much faster. We argue that this is because AlexNet is more overfit to the training set. Therefore, it takes more training iterations to forget the information that it has memorized from training examples. One might argue that the accuracy improvements are due to the extended training time of models. However, we experimented with training models for an equal number of total epochs and the model accuracies did not improve further. This is discussed further in Section 4.1.

Cross-Architecture Refinement: The architecture of a Label Refinery network can be different from that of the trained network. A high-quality Label Refinery should not overfit on training data even if its validation accuracy is high. In other words, under the same validation accuracy, a network with lower training accuracy is a better Label Refinery. Intuitively, this property allows the refinery to generate high-quality crop labels that are reflective of the true content of the crops. This property prevents the refinery from simply predicting the training labels. We observe that a ResNet-50 model trained to 75.7% top-1 validation accuracy on ImageNet can serve as a high-quality refinery. Table 2 shows that a variety of network architectures benefit significantly from training with refined labels. All network architectures that we tried using Label Refiner-

Model	Paper Number		Our Impl.		Label Refinery	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
AlexNet [8]	59.3	81.8	57.93	79.41	**66.28**†	**86.13**†
MobileNet [28]	70.6	N/A	68.53	88.14	**73.39**	**91.07**
MobileNet0.75 [28]	68.4	N/A	65.93	86.28	**70.92**	**89.68**
MobileNet0.5 [28]	63.7	N/A	63.03	84.55	**66.66**†	**87.07**†
MobileNet0.25 [28]	50.6	N/A	50.65	74.42	**54.62**†	**77.92**†
ResNet-50 [5]	N/A	N/A	75.7	92.81	**76.5**	**93.12**
ResNet-34 [5]	N/A	N/A	73.39	91.32	**75.06**	**92.35**
ResNet-18 [5]	N/A	N/A	69.7	89.26	**72.52**	**90.73**
ResNetXnor-50 [32]	N/A	N/A	63.1	83.61	**70.34**	**89.18**
VGG16 [6]	73	91.2	70.1	88.54	**75**	**92.22**
VGG19 [6]	72.7	91	71.39	89.44	**75.46**	**92.52**
Darknet19 [33]	72.9	91.2	70.6	89.13	**74.47**	**91.94**

Table 2: Using refined labels improves the accuracy of a variety of network architectures to new state-of-the-art accuracies. The Label Refinery used in these experiments is a ResNet-50 model trained with weight decay.
† These models can be further improved by training with adversarial inputs (Table 3).

ies gained significant accuracy improvement over their previous state-of-the-art. AlexNet and ResNetXnor-50[1] achieve more than a 7 point improvement in top-1 accuracy. Efficient and compact models such as MobileNet benefit significantly from cross-architecture refinement. VGG networks have a very high capacity and they overfit to the training set more than the other networks. Providing more accurate training set labels helps them to fit to more accurate signals and perform better at validation time. Darknet19, the backbone architecture of YOLOv2 [33], improves almost 4 points when trained with refined labels.

Adversarial Inputs: As discussed in Section 3.1 we can adversarially augment our training set with patches on which the refinery network and the trained model disagree. We used a gradient step of $\eta = 1$, as defined in Equation 3 to augment the dataset. We batch each adversarially modified crop with the original crop during training. This helps to ensure the trained model does not drift too far from natural images. We observe in Table 3 that smaller models further improve beyond the improvements from using a Label Refinery alone.

[1] ResNetXnor-50 is the XNOR-net [32] version of ResNet-50 in which layers are binary.

Model	GT Labels		Label Refinery		Adversarial	
	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
AlexNet	57.93	79.41	66.28	86.13	**67.2**	**86.92**
MobileNet0.5	63.03	84.55	66.66	87.07	**67.33**	**87.4**
MobileNet0.25	50.65	74.42	54.62	77.92	**55.59**	**78.58**

Table 3: Smaller models are further improved by training over adversarial inputs. The Adversarial Label Refinery is ResNet-50.

Model	Top-1	Top-5
AlexNet – no refinery	57.93	79.41
AlexNet – soft static refinery	63.55	84.16
AlexNet – hard dynamic refinery	64.41	84.53
AlexNet – soft dynamic refinery	66.28	86.13

Table 4: AlexNet benefits from both soft labeling and dynamic labeling. When combined the improvement is increased over both, suggesting that they capture different aspects of label errors. Label Refinery is ResNet-50.

4.1 Analysis

We explore the characteristics of models trained using a Label Refinery. We first explore how much of the improvement comes from the dynamic labeling of the image crops and how much of it comes from softening the target labels. We then explore the overfitting characteristics of models trained with a Label Refinery. Finally, we explore using various loss functions to train models against the refined labels. Most of the analyses are performed on AlexNet architecture because it trains relatively fast ($\sim$ 1 day) on the ImageNet dataset.

Dynamic Labels vs. Soft Labels: The benefits of using a label refinery are twofold: (1) Each crop is dynamically re-labeled with more accurate labels for the crop (Figure 2), and (2) images are softly labeled according to the distribution of visually similar objects in the crop (Figure 3). We find that both aspects of the refinement process improve performance. To assess the improvement from dynamic labeling alone, we perform label refinement with hard dynamic labels. Specifically, we assign a one-hot label to each crop by passing the crop to the Label Refinery and choosing the most-likely category from the output. To observe the improvement from soft labeling alone, we perform label refinement with soft static labels. To compute these labels for a given crop, we pass a center crop of the original image to the refiner rather than using the training crop. We compare the results for soft static labels and hard dynamic labels in Table 4. Both dynamic labeling and soft labeling significantly improve the accuracy of AlexNet.

Model	Top-1	Top-5
AlexNet – no refinery	57.93	79.41
AlexNet – taxonomy based refined categories	56.73	77.69
AlexNet – visually refined categories	58.54	80.77
AlexNet – visually refined images	62.69	83.46

Table 5: Comparing refining labels at category level vs. image level. Note that "AlexNet – visually refined images" is trained over image level refined labels as opposed to crop level. For fairness, we fixed the batch normalization layers of label refinery (which harms the quality of label refinery) in all visually refined labels experiments. Label Refinery is ResNet-50.

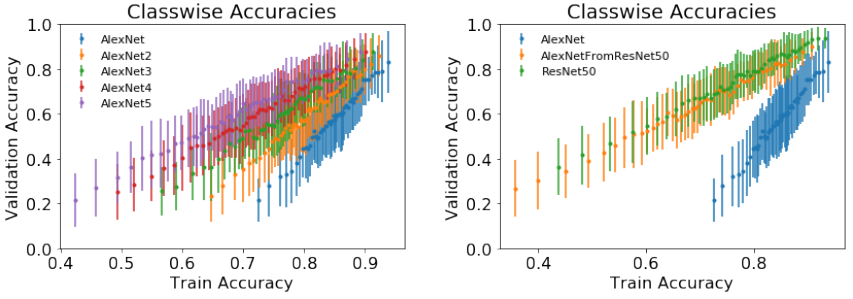


Fig. 4: Per category train and test accuracy. For each model, labels were sorted according to training set accuracies and divided into bins. each point in the plot shows the average validation set accuracy and the associated standard deviation for each bin. These figures show that training with a refinery results in models with less overfitting.

When they are combined we observe an additional improvement, suggesting that they address different issues with labels in the dataset.

Category Level Refining vs. Image Level Refining: Labels can be refined at the category level. That is, all images in a class can be assigned a unique soft label that models intra-category similarities. At the category level, labels can be refined either by visual cues (based on the visual similarity between the categories) or by semantic relations (based on the taxonomic relationship between the categories). Since ImageNet categories are drawn from WordNet, we can use taxonomy-based distances to refine the labels. We experiment with using the Wu-Palmer similarity [34] of the WordNet [35] categories to refine the category labels. Table 5 compares refining labels at the category level with refining at the image level. We observe larger improvements when the labels are refined

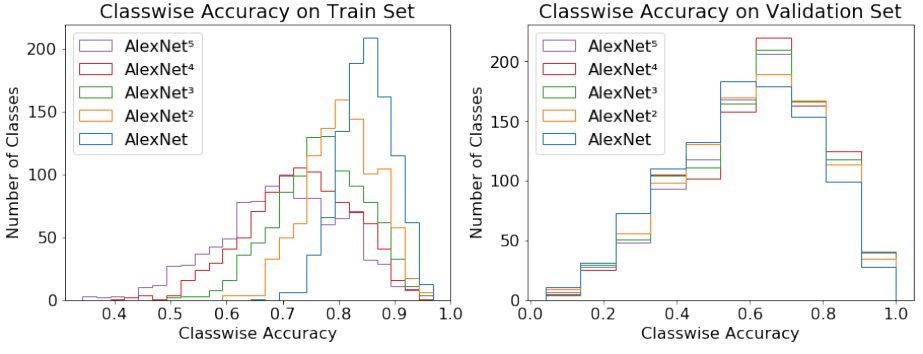


Fig. 5: The train and validation accuracy distribution of AlexNet models trained sequentially. AlexNet is trained off of the ground-truth labels, and the successive models AlexNet^{i+1} are trained off of the labels generated by AlexNet^{i}.

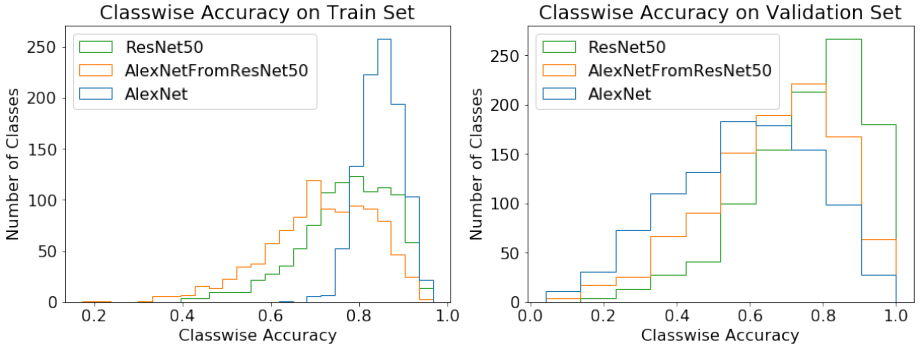


Fig. 6: The train and validation accuracies for AlexNet, ResNet, and AlexNet trained off of labels generate by ResNet50. `AlexNetFromResNet50` has a train accuracy profile that more closely resembles ResNet50 than AlexNet.

at the image level. Our experiment shows that taxonomy-based refinement does not improve training. We believe this is because WordNet similarities do not correlate well with visual similarities in the image space. Refining category labels based off of their WordNet distance can confuse the target model.

Model Generalization: Figure 4(a) and 4(b) show the per-category train and validation accuracies of ImageNet categories for models trained with a Label Refinery. Each point in the plot shows the average and standard deviation of the accuracies for a set of categories. Figure 4(a) shows the accuracies of a sequence of AlexNet models (in different colors). AlexNet trained using the ground-truth labels has much higher train accuracy. Successive models demonstrate less overfitting as shown by the decrease in the ratio between train accuracy and validation accuracy. Figure 4(b) shows the per-category accuracies of AlexNet and ResNet-50, as well as an AlexNet model trained with a ResNet50 Label Refinery. ResNet-50 trained with weight decay generalizes better compared to AlexNet, which has two fully connected layers. Intuitively, the generalization of ResNet-50 enables it to generate accurate per-crop labels for the training set. Thus, training AlexNet with a ResNet-50 Label Refinery allows AlexNet to perform well on the test set without overfiting to the original ground-truth labels.

Figure 5 shows the training set and validation set accuracies of a sequence of AlexNet models trained with a Label Refinery. The AlexNet trained with

Model	**Refinery**		**AlexNet**	
	Top-1	Top-5	Top-1	Top-5
AlexNet – no refinery	`N/A`	`N/A`	57.93	79.41
AlexNet – refinery: VGG16	70.1	88.54	60.78	81.80
AlexNet – refinery: MobileNet	68.53	88.14	65.22	85.69
AlexNet – refinery: ResNet-50	75.7	92.81	66.28	86.13

Table 6: Different architecture choices for the refinery network.

Model	Top-1	Top-5
AlexNet – no refinery	57.93	79.41
AlexNet – l_2 loss	63.16	85.56
AlexNet – KL-divergence from output to label	65.36	85.41
AlexNet – KL-divergence from label to output	66.28	86.13

Table 7: Different loss function choices. Label Refinery is ResNet-50.

ground-truth labels achieves $\sim 86\%$ training accuracy for the majority of classes, but achieves much lower validation set accuracies. By contrast, AlexNet[5] has a training accuracy profile more closely resembling its validation accuracy profile. Figure 6 shows a similar phenomena training AlexNet with a ResNet-50 refinery. It's interesting to note that the training and validation profiles of AlexNet trained with a ResNet50 Label Refinery more closely resemble the refinery than the original AlexNet.

Choice of Label Refinery Network: A good Label Refinery network should generate accurate labels for the *training set* crops. A Label Refinery's validation accuracy is an informative signal of its quality. However, if the Label Refinery network is heavily overfitted on the training set, it will not be helpful during training because it will produce the same ground-truth label for all image crops. Table 6 compares different architecture choices for refinery network. VGG16 is a worse choice of Label Refinery than MobileNet, even though VGG16 is more accurate. This is because VGG16 severely overfits to the training set and therefore produces labels too similar to the ground-truth.

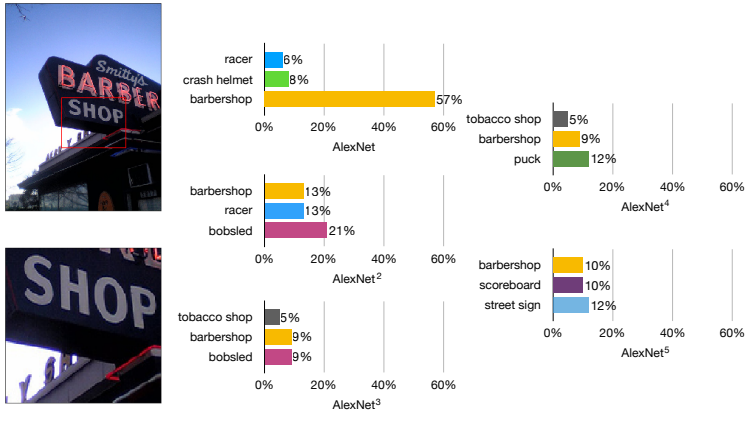


Fig. 7: The top three predictions for a crop of an image labelled "barber shop" in the ImageNet training set. AlexNet trained on the ground-truth labels is overfit towards the image level label. Successive AlexNet models overfit less, reducing the weight of the "barber shop" category and eventually assigning more probability to other plausible categories such as "street sign" and "scoreboard".

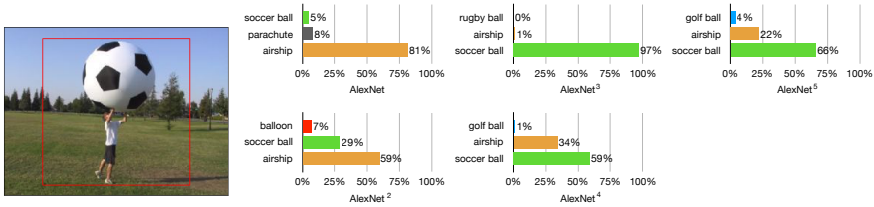


Fig. 8: The top three predictions for an image labelled "soccer ball" in the ImageNet validation set. Successive models learn to avoid overfitting an object surrounded by patches of sky to the "airship" category.

Choice of Loss Function: We can use a variety of loss functions to train our target networks to match the soft labels. The KL-divergence loss function that we use is a generalization of the standard cross-entropy classification loss. Note that KL-divergence is not a symmetric function (i.e. $D_{KL}(P||Q) \neq D_{KL}(Q||P)$). Table 7 shows the model accuracy if other standard loss functions are used.

Qualitative Results: Using a refinery to produce crop labels reduces overfitting by providing more accurate labels during training. In Figure 7, we see an example in which a training image crop does not contain enough information to identify the image category as "barbershop". In spite of this, AlexNet assigns the crop a label of barbershop with high confidence. This is due to overfitting on the training set. By using an AlexNet as a refinery, AlexNet^2 learns to generalize better. It produces a lower score for "barbershop", and a higher score for other categories. Generalization behavior improves with successive rounds of label refining until AlexNet^5 produces a smooth distribution over plausible categories. In Figure 8, we see an example of a "soccer ball" from the validation set of ImageNet. AlexNet incorrectly predicts "airship" with high confidence. This prediction is most likely because the main object is surrounded by blue sky, which is common for an airship but uncommon for a soccer ball. By using AlexNet as a refinery to train another AlexNet model we achieve a reduced score for "airship" and a higher score for "soccer ball". After several rounds of successive refining we achieve an AlexNet model that makes the correct prediction without completely forgetting the similarities between the soccer ball in the sky and an airship.

5 Conclusion

In this paper we address shortcomings commonly found in the labels of supervised learning pipelines. We introduce a solution to refine the labels during training in order to improve the generalization and the accuracy of learning models. The proposed Label Refinery allows us to dynamically label augmented training crops with soft targets. Using a Label Refinery, we achieve a significant gain in the classification accuracy across a wide range of network architectures. Our experimental evaluation shows improvement in the state-of-the-art accuracy for popular architectures including AlexNet, VGG, ResNet, MobileNet, and XNOR-Net.

References

1. Wang, J., Perez, L.: The effectiveness of data augmentation in image classification using deep learning. Technical report, Technical report (2017)
2. Wong, S.C., Gatt, A., Stamatescu, V., McDonnell, M.D.: Understanding data augmentation for classification: when to warp? In: Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on, IEEE (2016) 1–6
3. Xu, Y., Jia, R., Mou, L., Li, G., Chen, Y., Lu, Y., Jin, Z.: Improved relation classification by deep recurrent neural networks with data augmentation. arXiv preprint arXiv:1601.03651 (2016)
4. Shrivastava, A., Pfister, T., Tuzel, O., Susskind, J., Wang, W., Webb, R.: Learning from simulated and unsupervised images through adversarial training. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (July 2017)
5. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. (2016) 770–778
6. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
7. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: International conference on machine learning. (2015) 448–456
8. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. (2012) 1097–1105
9. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
10. Xie, L., Wang, J., Wei, Z., Wang, M., Tian, Q.: Disturblabel: Regularizing cnn on the loss layer. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 4753–4762
11. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2818–2826
12. Reed, S., Lee, H., Anguelov, D., Szegedy, C., Erhan, D., Rabinovich, A.: Training deep neural networks on noisy labels with bootstrapping. arXiv preprint arXiv:1412.6596 (2014)
13. Miyato, T., Maeda, S.i., Koyama, M., Nakae, K., Ishii, S.: Distributional smoothing by virtual adversarial examples. stat **1050** (2015) 2
14. Pereyra, G., Tucker, G., Chorowski, J., Kaiser, Ł., Hinton, G.: Regularizing neural networks by penalizing confident output distributions. arXiv preprint arXiv:1701.06548 (2017)
15. Li, Y., Yang, J., Song, Y., Cao, L., Luo, J., Li, J.: Learning from noisy labels with distillation. arXiv preprint arXiv:1703.02391 (2017)
16. Wu, C., Tygert, M., LeCun, Y.: Hierarchical loss for classification. arXiv preprint arXiv:1709.01062 (2017)
17. Wu, B., Lyu, S., Ghanem, B.: Ml-mg: Multi-label learning with missing labels using a mixed graph. In: Proceedings of the IEEE international conference on computer vision. (2015) 4157–4165

18. Cai, L., Hofmann, T.: Exploiting known taxonomies in learning overlapping concepts. In: IJCAI. Volume 7. (2007) 708–713
19. McAuley, J.J., Ramisa, A., Caetano, T.S.: Optimization of robust loss functions for weakly-labeled image taxonomies: An imagenet case study
20. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI. Volume 4. (2017) 12
21. Buciluǎ, C., Caruana, R., Niculescu-Mizil, A.: Model compression. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '06, New York, NY, USA, ACM (2006) 535–541
22. Ba, J., Caruana, R.: Do deep nets really need to be deep? In: Advances in neural information processing systems. (2014) 2654–2662
23. Li, J., Zhao, R., Huang, J.T., Gong, Y.: Learning small-size dnn with output-distribution-based criteria. In: Fifteenth Annual Conference of the International Speech Communication Association. (2014)
24. Shen, J., Vesdapunt, N., Boddeti, V.N., Kitani, K.M.: In teacher we trust: Deep network compression for pedestrian detection
25. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
26. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets. arXiv preprint arXiv:1412.6550 (2014)
27. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., et al.: Going deeper with convolutions, Cvpr (2015)
28. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
29. Nguyen, A., Yosinski, J., Clune, J.: Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2015) 427–436
30. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, IEEE (2009) 248–255
31. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in pytorch. (2017)
32. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: Xnor-net: Imagenet classification using binary convolutional neural networks. In: European Conference on Computer Vision, Springer (2016) 525–542
33. Redmon, J., Farhadi, A.: Yolo9000: Better, faster, stronger. In: Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on, IEEE (2017) 6517–6525
34. Wu, Z., Palmer, M.: Verbs semantics and lexical selection. In: Proceedings of the 32nd annual meeting on Association for Computational Linguistics, Association for Computational Linguistics (1994) 133–138
35. Miller, G.A., Beckwith, R., Fellbaum, C., Gross, D., Miller, K.J.: Introduction to wordnet: An on-line lexical database. International journal of lexicography **3**(4) (1990) 235–244