LANCET: Labeling Complex Data at Scale

Huayi Zhang
Worcester Polytechnic Institute
Worcester, MA
hzhang4@wpi.edu

Lei Cao*
Massachusetts Institute of Technology
Cambridge, MA
lcao@csail.mit.edu

Samuel Madden
Massachusetts Institute of Technology
Cambridge, MA
madden@csail.mit.edu

Elke Rundensteiner
Worcester Polytechnic Institute
Worcester, MA
rundenst@wpi.edu

ABSTRACT

Cutting-edge machine learning techniques often require millions of labeled data objects to train a robust model. Because relying on humans to supply such a huge number of labels is rarely practical, automated methods for label generation are needed. Unfortunately, critical challenges in auto-labeling remain unsolved, including the following research questions: (1) which objects to ask humans to label, (2) how to automatically propagate labels to other objects, and (3) when to stop labeling. These three questions are not only each challenging in their own right, but they also correspond to tightly interdependent problems. Yet existing techniques provide at best isolated solutions to a subset of these challenges. In this work, we propose the first approach, called LANCET, that successfully addresses all three challenges in an integrated framework. LANCET is based on a theoretical foundation characterizing the properties that the labeled dataset must satisfy to train an effective prediction model, namely the Covariate-shift and the Continuity conditions. First, guided by the Covariate-shift condition, LANCET maps raw input data into a semantic feature space, where an unlabeled object is expected to share the same label with its near-by labeled neighbor. Next, guided by the Continuity condition, LANCET selects objects for labeling, aiming to ensure that unlabeled objects always have some sufficiently close labeled neighbors. These two strategies jointly maximize the accuracy of the automatically produced labels and the prediction accuracy of the machine learning models trained on these labels. Lastly, LANCET uses a distribution matching network to verify whether both the Covariate-shift and Continuity conditions hold, in which case it would be safe to terminate the labeling process. Our experiments on diverse public data sets demonstrate that LANCET consistently outperforms the state-of-the-art methods from Snuba to GOGGLES and other baselines by a large margin – up to 30 percentage points increase in accuracy.

PVLDB Reference Format:
Huayi Zhang, Lei Cao, Samuel Madden, and Elke Rundensteiner. LANCET: Labeling Complex Data at Scale. PVLDB, 14(11): 2154-2166, 2021.
doi:10.14778/3476249.3476269

*Corresponding Author

Proceedings of the VLDB Endowment, Vol. 14, No. 11 ISSN 2150-8097.
doi:10.14778/3476249.3476269

1 INTRODUCTION

Motivation. Labeling is a key bottleneck that limits the success of modern machine learning architectures, such as deep learning, in enterprise deployments. Many AI-based applications from digital health to autonomous cars [4] require hundreds of thousands or even millions of labels to train an accurate and robust machine learning model. As a motivating example, we have been developing a large scale labeling system in collaboration with the Neurology Department of Massachusetts General Hospital, that is used to label EEG segments (450 million segments, 30TB data size) with 6 classes representing different types of seizures. These labeled EEG segments are used to train a seizure classifier that automatically detects seizures based on EEG signals collected during the clinical observation of patients. Our neurologist collaborators expect that well over 20 million labeled EEG segments are needed to cover the full range of variations in seizures. Relying on the domain experts to manually provide this large quantity of labels is impractical, given medical experts' time is extremely limited.

In this work, we tackle this challenge by building a labeling system that produces a sufficient number of labels with minimal human effort. To achieve this, several problems have to be solved:

(1) *Which objects to ask human experts to label?* To minimize the labeling efforts, a subset of objects must be selected for the human annotators to label. Compared to other possible subsets of objects, labeling of this chosen subset should *maximally* improve the performance of the machine learning model subsequently trained on it. This selection is critical for minimizing the human effort, especially when annotating data requires strong domain knowledge.

(2) *How to automatically generate additional labels?* Given that human experts will not be able to provide all necessary labels, a mechanism is needed to auto-generate labels by propagating manually produced labels to similar but still unlabeled objects. This label propagation is difficult, particularly when dealing with EEG/EKG data signals, video clips from autonomous vehicles or other such complex data types. First, when measuring the similarity of complex and thus often high-dimensional objects, we cannot simply rely on raw features of the data to distinguish between objects of different classes. Second, we need an appropriate distance function to measure the closeness of pairs of objects. Third, a distance threshold is required to determine if two objects are close enough to be labeled by the same class.

(3) *When to stop the labeling process?* To avoid wasting valuable human labeling effort, an effective labeling system must determine

whether the labels acquired so far are sufficient to achieve high training accuracy of the machine learning model. Although acquiring more labels may not hurt the performance of the machine learning model, the performance improvement brought by new labels is expected to diminish as the number of the labels increases. Therefore, an effective labeling system should automatically determine when the promise of additional performance improvement diminishes sufficiently so to safely terminate the labeling process.

State-of-the-Art. Although previous research has introduced some techniques for reducing human labeling efforts, in particular by weak supervision [7, 30, 38] and active learning [10–12, 32, 34, 34, 40], none of these works solve all three problems outlined above.

First, weak supervision methods such as Snorkel [30], Snuba [38], and GOGGLES [7] automatically generate labels using some labeling seeds. However, they assume the labeling seeds are provided either via some user defined labeling functions or are a priori user-supplied labeled examples. Although these labeling seeds impact the number and the error rate of the automatically produced labels, weak supervision does not address this critical question of which seeds are most effective at producing quality labels and thus should be initially acquired. That is, weak supervision addresses Problem 2 (Automatic Label Generation), but does not solve Problem 1 (Label Candidate Selection).

The problem of label candidate selection (Problem 1) is the focus of active learning [10–12, 32, 34, 34, 37, 40]. Active learning, however, does not tackle the challenge of automatically generating labels (Problem 2). Further, active learning techniques typically were designed to serve specific machine learning models such as Convolutional Neural Networks [32] or SVM [37]. Even if the labeling candidates recommended by active learning were potentially effective at improving the accuracy of specific prediction models, they are not guaranteed to be effective when used for automatic label generation.

One might think it would be possible to combine weak supervision and active learning to address our overall problem. However, our experiments (Sec. 7.2) show that combining the state-of-the-art weak supervision [38] to solve problem (2) and active learning techniques to solve problem (1) together does not improve the correctness of the produced labels compared to simply using randomly sampled labels in weak supervision. This implies that the two problems of which objects to label and how to automatically produce labels from such initially chosen candidate sets are *interdependent* and thus must be *solved jointly*, which is the approach we follow.

Finally, to the best of our knowledge, the above works do not offer a safe criteria for terminating the labeling process (Problem 3). Instead they leave this task to the users, for example by setting a labeling budget.

Proposed Approach. In this work, we propose the first end-to-end solution that effectively labels complex data at scale, called LANCET. Instead of providing a collection of independent techniques each targeting only one of the three research problems in isolation, LANCET solves all three of the above core automated labeling problems using a principled approach.

LANCET is built on a theoretical foundation that introduces the Covariate-shift condition and the Continuity condition in Sec. 3, guiding us to develop the label propagation and label candidate selection strategies.

Label Propagation. The Covariate-shift condition means that the unlabeled objects share the label with their near-by labeled neighbors. Intuitively if the distributions of the labeled objects and the unlabeled objects satisfy the Covariate-shift condition, then we can accurately produce labels by automatically propagating labels from the labeled objects to their near-by unlabeled neighbors. However, this Covariate-shift condition rarely holds true in the raw feature space of complex data. To solve this problem, we propose a feature embedding strategy, called *conditional feature matching* (CFM), to learn a new feature space satisfying this Covariate-shift condition. The key idea is to express the Covariate-shift condition as a *conditional feature matching loss* that serves as a regularization term to the loss function of any semi-supervised feature embedding method we plug into LANCET.

We then show in Sec. 5, that in this feature embedding space, a simple linear model would be sufficient to automatically propagate labels – without having to explicitly specify a distance function nor a distance threshold. This is beneficial as it is often impossible for experts familiar with their domain but not with machine learning to provide such machine learning specific parameters and functions.

Label Candidate Selection. Next, we observe that to reliably assign a label to each unlabeled object in a to-be-labeled dataset, each unlabeled object should always have some sufficiently near-by labeled neighbors in this embedding space. This observation is formalized as the Continuity condition in Sec. 3.

Guided by the Continuity condition, we design a *label candidate selection* strategy. It selects the candidate objects to be labeled by the domain experts as those whose coordinates x in the data space has a large value on the probability density function (PDF) of the unlabeled objects, but a small value on the PDF of the labeled objects. We call these unlabeled objects as the objects with large *probability density ratios*. These objects are not well represented by existing labeled objects.

Because estimating the PDF is notoriously hard in a high dimensional space [6, 9, 33, 43], we propose a strategy that estimates the *density ratios* without having to explicitly know the PDF. It leverages our insight that the density ratio estimation problem can be mapped to a distribution matching problem. That is, we show that given a set of weights w.r.t. each unlabeled object, if these weights together minimize the difference (distance) between the *weighted distribution* of the unlabeled objects and the distribution of labeled objects, they effectively estimate the density ratios.

We then solve this distribution matching problem by designing a *distribution matching network* (DMN), which given an object, determines if it is sampled from the weighted unlabeled distribution or the labeled distribution. We then learn these weights by *maximizing* its classification error rate and thus effectively minimizing the difference between those two distributions.

Labeling Termination. Using the above DMN, we naturally establish an *effective termination condition* for the labeling process. Namely, given objects in our data set, if the classification error rate of the DMN is close to 0.5 – indicating that it fails to distinguish labeled objects from unlabeled objects – then the labeling process should stop. The intuition is that because this condition assures that labeled objects effectively represent the distribution of the unlabeled objects, it would no longer improve the accuracy of the prediction model if we were to label more objects.

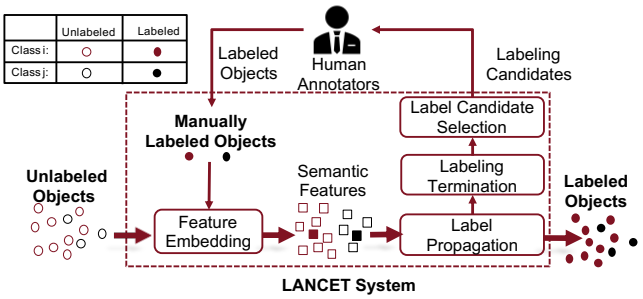


Figure 1: Overview of LANCET

The *conditional feature matching* strategy, the *label candidate selection strategy*, and the labeling termination condition jointly ensure that LANCET produces a sufficient number of quality labels with minimal human efforts.

Contributions. Our key technical contributions include:

- We propose LANCET, an end-to-end solution that effectively labels complex data at scale by solving the problems of label propagation, label candidate selection, and the termination of labeling in a *principled, integrated fashion.*
- We propose a conditional feature matching strategy that produces a feature space in which the objects belonging to different classes are linearly separable and thus greatly simplify the *automatic label generation problem.*
- We develop a density ratio estimation method that uses a distribution matching network to discover the labeling candidates that, if selected, would be most effective at improving the accuracy of the prediction model – solving the *label candidate selection problem.*
- We establish an effective criteria to promptly *terminate the labeling process* when additional labels will no longer bring a significant performance gain to the prediction model.
- Our experiments on diverse data sets ranging from images to time series confirm that LANCET significantly outperforms all alternative methods in the accuracy of both the generated labels and the machine learning models trained on these labels – up to 30 percentage points increase in accuracy.

2 LANCET OVERVIEW

LANCET targets the multi-class classification problem defined over an input space $\mathcal{X} \in \mathrm{R}$ and a label space $\mathcal{Y} = \{1, ..., C\}$. The objective of LANCET is to use minimal human labeling efforts to produce labels that are sufficient to solve the classification problem.

As shown in Fig. 1, LANCET consists of four components, namely feature embedding, label propagation, label candidate selection, and labeling termination. Next, we briefly introduce how to use LANCET to solve the labeling problem.

Labeling in LANCET is an iterative process. Using the existing manually labeled objects, LANCET first employs the feature embedding component to project the raw input data into a semantic feature space that takes the classification task into consideration. On top of the semantic feature space, LANCET uses the label propagation component to propagate labels from the manually labeled objects to the unlabeled objects. The labeling termination component then determines if existing labels are already sufficient to train an accurate classification model. If not, LANCET uses the label candidate selection component to select at most b objects as candidates for manual labeling, where b is a user-controlled parameter.

In the next iteration, LANCET uses the enriched pool of manually labeled objects to update the semantic feature space and continues the labeling process. It stops when the labeling termination component determines that continuing to label will not lead to clear performance gains.

3 LANCET THEORETICAL FOUNDATION

In this section, we establish the theoretical foundation for LANCET. The key insight is that as long as the distributions of both the labeled objects and unlabeled objects satisfy certain statistical properties, we can accurately infer the labels of the unlabeled objects. This finding guides the design of our strategies for feature embedding, label propagation, and label candidate selection.

We denote the labeled objects as $\mathcal{D}_l = \{(x_l^i, y_l^i)\}_{i=1}^{N_l}$ and unlabeled objects as $\mathcal{D}_u = \{x_u^i\}_{i=1}^{N_u}$, where y_l^i is the label of object x_l^i, and N_l and N_u denote the number of labeled and unlabeled objects, respectively. We assume the labeled objects are sampled from the joint probability distribution $P_l(x, y)$ in domain $\mathcal{X} \times \mathcal{Y}$. Accordingly, the unlabeled data objects are sampled from the joint probability distribution $P_u(x, y)$. Note that in practice, we do not have access to their ground truth labels for the unlabeled data objects $\{y_u^i\}_{i=1}^{N_u}$.

We show that, if the *labeled* and *unlabeled* objects jointly satisfy certain conditions detailed below, we can accurately predict the label $\hat{y}_u$ of each unlabeled object x_u. More formally, as long as these conditions hold, there exists a classification model $\phi(x)$ that makes Eq. 1 hold.

$$\mathbf{E}_{(x,y)\in P_u(x,y)}[loss(\hat{y}, y)] = \int_x p_u(x) loss(\hat{y}, y) dx < \epsilon \quad (1)$$

In Eq. 1, ϵ is a positive value that can be arbitrarily small. $\hat{y}$ represents the label w.r.t. each object x predicted by $\phi(x)$. $loss(\hat{y}, y)$ corresponds to the cross entropy loss between the prediction $\hat{y} = \phi(x)$ and the ground truth label y, as defined in Eq. 2.

$$loss(\hat{y}, y) = -\sum_{i=1}^{C} p_u(y_i|x) log p(\hat{y}_i|x) \quad (2)$$

In other words, the two conditions introduced below ensure that the label propagated to the unlabeled objects has an *expected* cross entropy loss *close to 0* and hence is near perfect. Next, we introduce these two conditions and then formally prove the above *sufficiency* claim in Lemma 1.

Definition 3.1. The Covariate-shift condition [33, 43], expressed by Eq. 3, indicates that the joint distribution of unlabeled objects $p_u(x, y)$ and that of labeled objects $p_l(x, y)$ should have the same conditional probability mass function given a value x in the feature space $\mathcal{X}$.

$$p_u(y|x) = p_l(y|x) \quad (3)$$

Intuitively, if the Covariate-shift condition holds, highly likely an unlabeled object will share the label with its close labeled neighbors. Otherwise, it would be impossible to accurately infer the labels of unlabeled objects no matter how many labeled objects were to exist. This is because machine learning models tend to infer the class of an unlabeled object based on its similarity to labeled objects.

This Covariate-shift condition guides us to develop the *feature embedding* approach to simplify the label generation problem, as described in Sec. 4.

Continuity Condition. The Continuity condition is based on the concept of Radon-Nikodym derivative (RND) [43].

Definition 3.2. The Radon-Nikodym derivative (RND) [43] is denoted as $\beta(x) = \frac{p_u(x)}{p_l(x)}$, where $p_u(x)$ and $p_l(x)$ represent the probability density functions (PDF) of unlabeled objects and labeled objects respectively. The **Continuity condition** holds if $\beta(x) < B$ and $B \ll \infty$.

The Radon-Nikodym derivative (RND) is also called the *importance weight* or *density ratio* in the literature [23, 43]. RND is not well defined if there exists an unlabeled object with coordinate x such that x has a large value on the PDF of the unlabeled objects ($p_u(x) > 0$), but a very small value on the PDF of the labeled objects ($p_l(x) \approx 0$). Intuitive, in this case this object does not have close labeled neighbors to represent it. Driven by this Continuity condition, we design a *label candidate selection* method (Sec. 6).

Based on the definitions of Covariate-shift condition and Continuity condition, we are ready to prove the key sufficiency claim.

Lemma 1. *Assume in a feature space $\mathcal{X}$, the unlabeled objects $x_u \in \mathcal{D}_u$ and labeled objects $x_l \in \mathcal{D}_l$ satisfy the Covariate-shift and Continuity conditions. Then given a classification model $\phi(x)$ which infers the label of each object $\hat{y}$, if $\int_{x\in\mathcal{D}_l} p_l(x)loss(\hat{y_l}, y_l))dx \leq \frac{\epsilon}{B}$, then $\int_{x\in\mathcal{D}_u} p_u(x)loss(\hat{y_u}, y_u))dx \leq \epsilon$, where ϵ corresponds to a positive value that can be arbitrarily small and B is the threshold used to define Continuity condition.*

Proof. When the Covariate-shift and Continuity conditions hold, the expected cross-entropy loss on the unlabeled objects can be easily estimated based on the training loss of the labeled objects and the RND value $\beta(x)$ used in Continuity condition.

$$\begin{aligned}\int_x p_u(x)l(\hat{y_u}, y_u)dx &= -\int_x p_u(x)[\sum_{i=1}^{C} p_u(y_i|x)logp(\hat{y}_i|x)]dx \\ &= -\int_x \frac{p_u(x)}{p_l(x)} p_l(x) \sum_{i=1}^{C} p_l(y_i|x)logp(\hat{y}_i|x)dx \\ &= \int_x \beta(x)p_l(x)loss(\hat{y_l}, y_l)dx \\ &\leq B\int_x p_l(x)loss(\hat{y_l}, y_l)dx \leq \epsilon\end{aligned} \tag{4}$$

Lemma 1 is proven. □

Lemma 1 shows that a model $\phi(x)$ will be effective at propagating labels in a feature space that satisfies the Covariate-shift and Continuity conditions, if $\phi(x)$ is able to minimize the training loss on the labeled objects close to 0. Such a model $\phi(x)$ tends to exist in practice. This is because it is widely observed [44] that machine learning models, especially deep learning models, can perfectly fit any training dataset even if its labels are randomly produced. In other words, they are able to minimize the training loss to 0.

However, usually the Covariate-shift and Continuity conditions do not naturally hold in the real world, complex data sets. The goal of LANCET is thus to make the data satisfy these conditions and ensure the effectiveness of the downstream classification tasks by transforming the data space (Sec. 4), effectively propagating labels (Sec. 5), and smartly selecting objects for human to labels (Sec. 6).

4 FEATURE EMBEDDING

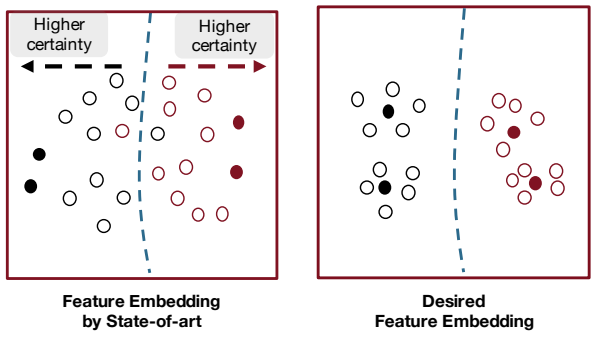


Figure 2: (a) The semantic features extracted by existing semi-supervised models [6, 31]. Because theses models tend to overfit the labeled objects by pushing them far from decision boundaries, the semantic features of labeled and unlabeled data diverge from each other. (b) The desired semantic features to perform label propagation. The labeled and unlabeled objects should be close to each other if they fall into the same class.

According to the Covariate-shift condition, to accurately infer the labels of the unlabeled objects, any unlabeled object should share the label of its close labeled neighbors as shown in Fig. 2 (b). However, this often does not hold in the raw feature space of the input data, especially for the highly complex, high dimensional data such as images or time series. This is because the raw features of the complex data typically are not informative at distinguishing between the objects belonging to different classes [16]. Thus, the objects are not naturally separable by their classes.

To solve this problem, we develop a feature embedding method customized for our labeling task, called conditional feature matching or CFM for short. Guided by the small number of labels at hand, CFM projects the input data into a semantic feature space $\mathcal{Z}$ in which Eq. 3 holds, as required by the Covariate-shift condition.

4.1 Conditional Feature Matching

Our goal is to learn a feature embedding model $\mathcal{M}(x)$ that maps the input data $\mathcal{X}$ into a semantic feature space $\mathcal{Z}$ satisfying the Covariate-shift condition. LANCET learns $\mathcal{M}(x)$ based on both the labeled objects $\mathcal{D}_l$ and unlabeled objects $\mathcal{D}_u$. We use both classes of objects for two reasons. First, the Covariate-shift condition reflects the statistical properties of both labeled and unlabeled objects. Therefore, the feature embedding model has to take the labeled objects into consideration. Purely unsupervised feature embedding method such as AutoEncoders [36, 39] do not satisfy this need. Second, in the labeling task, not many labeled objects are available apriori. Therefore, it is not practical to train a purely supervised feature embedding model.

Intuitively, semi-supervised classification techniques such as SemiGAN [6, 31] appear to be a natural solution to this problem. Semi-supervised classification learns a classification model by jointly minimizing the loss incurred by both the labeled objects and unlabeled objects, namely the labeled loss and the unlabeled loss. Similar to the classical deep learning models, the learned classification model $\mathcal{F}(x, \theta)$ automates the feature extraction process. Therefore, logically it can be decomposed into a feature extractor $\mathcal{M}$ and a classifier f. That is, $\mathcal{F}(x, \theta) = f(\mathcal{M}(x))$. $\mathcal{M}(x)$ produces a semantic feature space $\mathcal{Z}$.

The Insufficiency of Existing Semi-supervised Feature Embedding. Although the semantic features learned in this way is shown to be effective at separating objects belonging to different classes, they do not necessarily respect the Covariate-shift condition. In particular, we observe that these techniques tend to overly minimize the labeled loss by pushing the labeled objects far from the classification decision boundary in the learned semantic feature space. As a consequence, the labeled objects often are isolated far from the unlabeled objects, thus violating the Covariate-shift condition, as depicted in Fig. 2 (a).

Solution: Condition Feature Matching. To solve this problem, we propose the conditional feature matching strategy. The key idea is to express the Covariate-shift condition defined in Eq. 3 as a conditional feature matching (CFM) loss (Eq. 5). This way, it can be seamlessly plugged into the loss function of the semi-classification model $\mathcal{F}(x, \theta)$ which originally had only included the labeled loss and the unlabeled loss. As a regularization of the learning process, this conditional feature matching loss enforces that the unlabeled objects are close to the labeled objects in the semantic feature space $\mathcal{Z}$ if they belong to the same class; and it satisfies the Covariate-shift condition with a theoretical guarantee.

In the semantic feature space $\mathcal{Z}$, we denote a labeled object as z^i_{l,c_m} if it belongs to class c_m. We use N_{l,c_m} to represent the number of labeled objects that belong to class c_m. Accordingly, we denote one unlabeled object as z^i_{u,c_m} if it is classified to class c_m. N_{u,c_m} denotes the number of unlabeled objects classified to class c_m. Next, we formally define the CFM loss.

$$loss_{cfm} = \sum_{c_m=1}^{C} [\|\mu_{l,c_m} - \mu_{u,c_m}\| + \sum_{n \neq m}^{C} \max\{0, \lambda - \|\mu_{l,c_m} - \mu_{u,c_n}\|\}] \tag{5}$$

In Eq. 5, $\mu_{l,c_m} = \frac{1}{N_{l,c_m}} \sum_{i=1}^{N_{l,c_m}} z^i_{l,c_m}$ represents the center of all class c_m *labeled* objects, while $\mu_{u,c_m} = \frac{1}{N_{u,c_m}} \sum_{i=1}^{N_{u,c_m}} z^i_{u,c_m}$ represents the center of all *unlabeled* objects that are predicted as class c_m. λ controls the distance between μ_{u,c_m} and μ_{u,c_n}, where c_m and c_n denote two different classes. Minimizing Eq. 5 thus will encourage the unlabeled and the labeled objects to centralize into the same region in the semantic feature space if they potentially share the same label, and push them far away from each other if they belong to different classes.

Theoretical Guarantee. We first show that an unlabeled object is guaranteed to share the label with its close labeled neighbor if we can minimize the CFM loss to 0. Then we establish the connection between the CFM loss and the Covariate-shift condition.

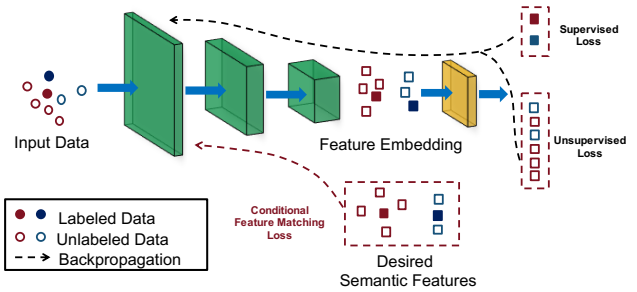


Figure 3: Training Process of Conditional Feature Matching

Assume the unlabeled objects in class c_m follow the Gaussian distribution. So do the labeled objects. The distributions are independent to each other. Their covariance matrices are the unit matrix, i.e. $\{z^i_{l,c_m}\}_{i=0}^{N_{l,c_m}} \sim \mathcal{N}(\mu_{l,c_m}, I)$, $\{z^j_{u,c_m}\}_{j=0}^{N_{u,c_m}} \sim \mathcal{N}(\mu_{u,c_m}, I)$. Given a pair of labeled and unlabeled objects (z^i_l, z^j_u), we use $c(i = j)$ to represent that they belong to the same class and use P^+_{ij} to denote its probability. Accordingly we use $c(i \neq j)$ to represent the pair of objects belonging to different class and denote its probability as P^-_{ij}. We use d_{ij} to denote $z^i_l - z^j_u$.

Lemma 2. *Assume the labeled objects $\{z^i_{l,c_m}\}_{i=0}^{N_{l,c_m}}$ and unlabeled objects $\{z^j_{u,c_m}\}_{j=0}^{N_{u,c_m}}$ of class c_m independently show Gaussian distribution. Then given a pair of labeled and unlabeled objects (z^i_l, z^j_u) where $\|d_{ij}\| < \epsilon$ with ϵ denoting a small value close to 0, if the CFM loss is minimized to 0, then $P(c(i = j) \big| \|d_{ij}\| < \epsilon) \geq \frac{P^+_{ij}}{P^+_{ij} + \sum_{i \neq j} P^-_{ij} e^{-\frac{\lambda^2}{4}}}$.*

Due to space limit, please refer to the extended version [1] of this paper for the proof.

Note $0 \leq P^-_{ij} \leq 1$. Because the parameter λ is large, $e^{-\frac{\lambda^2}{4}}$ is close to 0. Therefore, typically $P(c(i = j) \big| \|d_{ij}\| < \epsilon)$ is close to 1. This shows that, if a pair of labeled and unlabeled objects are close, then the probability that they belong to the same class is close to 1.

The Connection to Covariate-shift Condition. By Lemma 2 if $\|z^i_l - z^j_u\| < \epsilon$, then the probability that they belong to the same class is close to 1. That is, $P(c(i = j) \big| \|z^i_l - z^j_u\| < \epsilon) = 1$. This is equivalent to $p_l(y|z) = p_u(y|z + \epsilon \cdot r)$, where r represents any unit vector. When ϵ approaches to 0, then $(z + \epsilon \cdot r) \to z$. Therefor, the Covariate-shift condition $p_u(y|z) = p_l(y|z)$ holds.

4.2 The Feature Embedding Method

Because Eq. 5 is differentiable, the conditional feature matching loss can be seamlessly plugged into the loss function of the semi-supervised classification model $\mathcal{F}(x, \theta)$, requiring no change to its learning strategy. Therefore, we can leverage any existing semi-supervised model to learn a semantic feature space satisfying the Covariate-shift condition. In this work, we adopt the state-of-the-art of semi-supervised classification, namely, Generative Adversarial Network (GAN)-based SemiGAN model [6, 31].

An Overview of SemiGAN. SemiGAN is an extension of the Generative Adversarial Networks(GAN) [24, 29]. GANs were originally developed to generate synthetic data objects. Toward this goal, a GAN model trains a Generator network G to transform a random

vector $z \sim \mathcal{N}(0, I)$ to a synthetic data objects. To evaluate the quality of generated objects, a GAN contains a discriminator network D that is trained to distinguish the true data objects and synthetic data objects. The generator network is then trained to fool the discriminator network to accept its output as being real. In turn the discriminator network needs to learn a precise boundary of true data objects to reject the fake objects. In the learning process, a GAN produces a good feature embedding of the true data.

As with GANs, a SemiGAN also has a generator G and a discriminator D. Unlike the traditional GAN, in the SemiGAN model, the discriminator and generator use different objective functions in order to leverage a small number of available labels. The generator aims to generate fake objects that have a similar feature embedding to that of real objects in the semantic feature space. The discriminator is jointly trained to satisfy two objectives, namely correctly classifying the labeled objects and distinguishing the fake objects from real objects.

Formally, the objective function of the generator in SemiGAN is:

$$loss_G = ||\mathbf{E}_x(\mathcal{M}(x)) - \mathbf{E}_s(\mathcal{M}(G(s)))||_2^2 \quad (6)$$

where $\mathcal{M}$ is the feature embedding model of the discriminator. s is the input of the generator model and sampled from a Gaussian distribution. $G(s)$ is the fake data object produced by the generator.

In a K class classification task, the objective function of the discriminator in SemiGAN is composed of two types of losses, namely, the labeled loss to classify the real objects:

$$loss_l = \mathbf{E}_{x,y\in\mathcal{D}_l}(logP_D(y|x, y \leq K)) \quad (7)$$

and the unlabeled loss that distinguishes the real and fake objects:

$$loss_u = \mathbf{E}_{x,y\in\mathcal{D}_u}(logP_D(y \leq K|x)) + E_s(logP_D(K+1|G(s))) \quad (8)$$

The discriminator classifies the unlabeled real objects into one of the K classes, while assigns the fake objects to a $K+1$th class.

Enhancing SemiGAN with CFM. LANCET learns a semantic feature space that satisfies the Covariate-shift condition by adding the conditional feature matching loss $loss_{cfm}$ (Eq. 5) into the objective of the discriminator, as depicted in Fig. 3.

$$loss_D = loss_l + loss_u + loss_{cfm} \quad (9)$$

Because the conditional feature matching $loss_{cfm}$ is differentiable, the optimization strategy of SemiGAN is still effective at minimizing the new form of loss function l_D without further change.

Note that LANCET does not rely on any specific semi-supervised models to extract features. Rather, users can choose a semi-supervised model that best fits their targeted classification task and their dataset. Our conditional feature matching loss can seamlessly be added into its loss function.

5 LABEL PROPAGATION

Next, we introduce our label propagation strategy that automatically propagates labels from manually labeled objects to the unlabeled data objects. As described in Sec. 4, after the semantic feature embedding, objects belonging to the same class are close to each other in the semantic feature space. In this scenario, label propagation seems straightforward. For example, we could first measure the similarity between the labeled object z_l and the unlabeled object z_u, and then if they were close enough we could propagate the label of z_l to z_u. However, this intuitive strategy raises some critical research questions:

(1) How to decide if two objects are similar? The semantic feature space typically corresponds to a high dimensional space, especially when the original input data is complex. Yet effectively measuring the similarity in high dimensional spaces remains an open problem due to the curse of dimensionality [25].

(2) How close is close enough? We need an appropriate similarity threshold to determine if two objects are close enough to be said to belong to the same class. This threshold is hard to set.

A Linear Model-based Solution. LANCET uses a lightweight machine learning model to solve the above problems. It fully explores the advantage of the semantic feature space $\mathcal{Z}$ produced by our conditional feature matching (CFM) strategy in that it satisfies the Covariate-shift condition (Def. 3.1). Instead of explicitly measuring the similarity among the objects and carefully selecting a similarity threshold to propagate labels, LANCET learns a machine learning model $\mathcal{F}(z)$ to propagate labels in the semantic feature space $\mathcal{Z}$. This model $\mathcal{F}(z)$ produces a prediction for each unlabeled object given its features in $\mathcal{Z}$. Because $\mathcal{Z}$ satisfies the Covariate-shift condition, $\mathcal{F}(z)$ can be very lightweight. We will prove that in fact a linear neural net [22] is sufficient for this label propagation task.

More precisely, we parameterize the linear neural network $\mathcal{F}(z)$ by a weight matrix W and a bias vector b, i.e., $\mathcal{F}(z|W, b) = W \cdot z + b$. We use cross-entropy as the loss function. Given an unlabeled object z_u, $\mathcal{F}(z|W, b)$ produces a label vector $\hat{y}_u = [y_1, ..., y_C]$, where C represents the number of classes. Each dimension of $\hat{y}_u$ represents the probability that z_u belongs to one particular class i $\in \{1, ..., C\}$. In deep learning, this vector is typically produced by a softmax function $f(\cdot)$. Learning this linear neural network is straightforward, and we use the common approach of SGD optimization [16].

Theoretical Guarantee. We formally prove that this lightweight model is effective at propagating labels, leveraging that the semantic feature space satisfies the Covariate-shift condition (Eq. 3). Due to space limit, please refer to the extended version [1] of this paper for the proof.

Jointly Learning with Conditional Feature Matching. Due to the linearity of this model, it is jointly learnable with the semantic feature embedding model. For example, if LANCET uses SemiGAN to realize our CFM strategy to produce the semantic feature space, then our linear neural network can thus be simply implemented as the final layer of the discriminator of SemiGAN. Therefore, LANCET can learn the semantic feature extraction model and the label propagation model jointly using Stochastic Gradient Decent (SGD) and back propagation, following the typical training process of SemiGAN. This produces a propagation model best aligned to the semantic feature embedding.

6 LABEL CANDIDATE SELECTION & LABELING TERMINATION

In this section, we first introduce our label candidate selection method. Then in Sec 6.4 we show that this method naturally establishes an effective criteria for termination of the labeling process.

Continuity Condition Driven Label Candidate Selection. Guided by the Continuity condition in Sec. 3, LANCET selects (at most) b objects that, if labeled by the human annotators, would

be most effective at improving the quality of the produced labels, where b is a user configurable parameter.

Our label candidate selection method is inspired by the Continuity condition. To recap, the Continuity condition holds as long as the Radon-Nikodym derivative (RND) $\beta(z) = \frac{p_u(z)}{p_l(z)}$ is bounded by a small value (Sec. 3). Here $p_u(z)$ and $p_l(z)$ represent the probability density functions (PDF) of unlabeled objects and labeled objects, respectively. Intuitively, given an unlabeled object with its value as z, if the RND $\beta(z)$ is large, then it does not have a close labeled neighbor to represent it.

We thus conclude that to satisfy the Continuity condition, we should give high priority to the unlabeled objects with large RND when selecting objects for labeling, so that unlabeled points have sufficient numbers of labeled points near-by to perform propagation. Therefore, once we have computed the RND for each object, naturally we can select as labeling candidates the b objects that have the *largest* RNDs.

However, calculating the RND requires the estimation of the PDF $p_u(z)$ and $p_l(z)$. This is hard in our scenario where the dimensionality of the semantic feature space often is high, because accurately estimating the PDF of high dimension data is an open problem [6, 9, 33, 43].

6.1 Learning RND By Distribution Matching

To solve this problem, we propose a method that directly estimates the RND $\beta(z)$ without having to first separately estimate $p_l(z)$ and $p_u(z)$. For the ease of presentation, we define the concept of weight $\mathbf{w(z)} = \frac{p_l(z)}{p_u(z)}$ as the reverse of $\beta(z)$. Essentially, $w(z)$ represents the probability that a unlabeled object has a close labeled neighbor.

Solution: Mapping to the Weighted Distribution Matching Problem. Our insight is that the problem of estimating the $w(z)$ can be transformed into the *weighted distribution matching* problem.

$$w(z) = \frac{1}{\beta(z)} = \frac{p_l(z)}{p_u(z)} \Rightarrow w(z)p_u(z) = p_l(z) \tag{10}$$

As shown in Eq. 10, after being weighted by $w(z)$, the probability density $p_u(z)$ of the unlabeled objects is equivalent to the probability density $p_l(z)$ of the labeled object. This implies that we can directly estimate $w(z)$ by identifying an appropriate weight for the unlabeled objects such that the weighted distribution of unlabeled objects matches the distribution of labeled objects.

However, even learning $w(z)$ by distribution matching is challenging. First, given two distributions in a high dimensional space, we need a method to evaluate if these two distributions match with each other. Second, even if we can correctly evaluate if the weighted distribution of the unlabeled objects matches the distribution of the labeled objects, we still need an effective yet efficient method to search for the weight $w(z)$ to match the distribution.

We propose to use a *distribution matching network (DMN)* to solve this problem. We first introduce the concept of a DMN and then show how to efficiently learn these weights through DMN.

6.2 Distribution Matching Network (DMN)

Let $\phi(z, \theta_d)$ denote a neural network model, where θ_d represents the parameters. We use y_d to denote the label of z, indicating which distribution z belongs to. The objective is to assign each object to one distribution from which it is most likely sampled. Therefore, we call this neural network the *Distribution Matching Network* (DMN). If two distributions exactly match each other – meaning $p_l(z) = p_u(z)$, DMN will fail to distinguish between the labeled and unlabeled objects in the semantic feature space $\mathcal{Z}$. As a result, its classification errors would be around 0.5 on average. Formally, a DMN $\phi(z, \theta_d)$ minimizes the weighted expected loss between the predicted label $\hat{y_d} = \frac{e^{\phi(z,\theta_d)}}{\sum_{i=0}^{2} e^{\phi(z,\theta_d)_i}}$ and y_d, as defined below.

$$\begin{aligned}\mathbf{E}(loss(\hat{y_d}, y_d)) = -\int_z [&w(z)p_u(z)p(y_d)logp(\hat{y_d}|z_u) + \\ &p_l(z)p(y_d)logp(\hat{y_d}|z_l)]dz \\ = \frac{1}{(N_u + N_L)}[&\sum_{i=0}^{N_u} w_i loss(\hat{y_d}_i, y_d) + \sum_{i=0}^{N_l} loss(\hat{y_d}_i, y_d)]\end{aligned} \tag{11}$$

$$y_d|_{z_i} = \begin{cases} 0, \text{if } z_i \text{ is unlabeled} \\ 1, \text{if } z_i \text{ is labeled} \end{cases} \tag{12}$$

In Eq. 11, w_i denotes the weight corresponding to z_i. w_i is given beforehand and remains fixed throughout the training process. The expected loss $E(loss(z, y_d))$ corresponds to the $\mathcal{H}$ divergence, a widely used metric to measure the difference between two distributions [3, 13, 14]. DMN learns the parameter θ_d to minimize the $\mathcal{H}$ divergence.

If $E(loss(z, y_d))$ is still large after the DMN converges, then the weighted distribution of the unlabeled objects matches the distribution of the labeled objects. In particular, when the two distributions are identical, the expected loss $loss(z, y_d, w(z))$ should be around 1.38 [33], which corresponds to the maximal value of cross-entropy loss in binary classification. This offers us a criteria to verify if two given distributions match each other.

6.3 Learning Weights Through DMN: Online Weight Approximation

Key Idea. DMN offers us a tool to effectively verify if the weighed distribution of the unlabeled objects matches the distribution of the labeled objects, assuming the weights w_i were to be available beforehand. Next, we show how to use the DMN to learn these weights. The key idea is to treat the weights $\{w_i\}_{i=0}^{N_u}$ as the hyper-parameters of the DMN and learn them by *maximizing* the expected loss of $\theta_d(z, y_d)$. The intuition is that the appropriate weights should best match the distribution of unlabeled objects to that of labeled objects, while two matching distributions will lead to a *large* loss in DMN, as discussed above. This objective is formalized in Eq. 13.

$$\begin{aligned}\theta^*(w) &= \arg\min \mathbf{E}(loss(\hat{y}, y_d)|_w) \\ w^* &= \arg\max \sum_{i=0}^{N_u} w_i loss(\hat{y}, y_d)|_{\theta(w)}\end{aligned} \tag{13}$$

Here $\theta^*(w)$ indicates the learned parameter of DMN given the current weights.

Learning the optimal value of w_i can be very expensive, because the re-weighting process requires two loops of learning, namely, (1) learning the parameter θ^* of DMN to minimize the loss in Equation 13, and (2) learning the weights to maximize the loss of DMN based on the newly learned parameter θ^*. The two loops iterate back and forth until convergence – until the loss of DMN is around

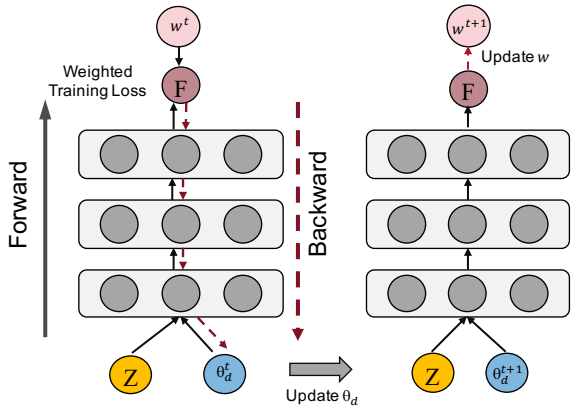


Figure 4: Training Process of Online Weight Approximation

1.38, because by [33] it indicates the two distributions now match with each other.

Online Weight Approximation: an Efficient Method. We propose an online weight approximation (OWA) strategy to efficiently learn w_i. OWA alternates updates of θ^* with updates of w^*, as shown in Fig. 4. Because the number of unlabeled object N_u can be large, training a DMN by iteratively learning the weights with respect to all objects is very expensive. OWA employs a mini-batch based optimization strategy to address the efficiency concern. During each training iteration, OWA randomly divides the data into many mini-batches and concurrently learns the weights with respect to each mini-batch, so called *online weight approximation*. Each mini-batch contains only $n \ll N_u$ objects.

More specifically, given a DMN $\phi(z, \theta_d)$, at the t-th iteration of the training, $\phi()$ is trained to minimize the weighted loss of the current batch with size $2n$, which includes n labeled data objects and n unlabeled data objects, assuming each w_i^t is fixed at the current iteration t.

$$loss(\hat{y_d}, y_d)_t = \frac{1}{2n}[\sum_{i=0}^{n} w_i^t loss(y_{\hat{d}_u}, y_{d_u}) + \sum_{i=0}^{n} loss(y_{\hat{d}_l}, y_{d_l})]$$
$$\theta_d^{t+1} = \theta_d^t - \eta \nabla \sum_{i}^{2n} loss(\hat{y_d}, y_d)|\theta_d^t \tag{14}$$

where η is the descent step size of the optimizer.

OWA searches for the optimal w^* of the unlabeled objects in the mini-batch that maximizes the weighted loss $loss(\hat{y_d}, y_d)$ by Eq. 14.

$$w^* = \arg\max_{w} \frac{1}{n} \sum_{i=1}^{n} loss(\hat{y_d}, y_d)|_{\theta_{t+1}} \tag{15}$$

OWA adopts gradient descent to update the weight w_i^t at each iteration t. OWA recalculates the loss based on the updated parameters, and then updates the weight of the unlabeled objects according to the updated loss.

$$loss(y_{\hat{d}_u}, y_d)_{t+1} = \frac{1}{n} \sum_{i=0}^{n} w_i^t loss(y_{\hat{d}_u}, y_d)|_{\theta_{t+1}}$$
$$w_i^{t+1} = \frac{\partial}{\partial w_i^t} loss(y_{\hat{d}_u}, y_d)_{t+1} \tag{16}$$

At the end of each iteration, the weights of all objects in each mini-batch have to be normalized to make sure the total sample weights add up to 1. It is also necessary to ensure $w_i \geq 0$ for all i, since minimizing the negative training loss can result in unstable behavior. Therefore, we enforce these constraints:

$$(1) w_i^t = \max(0, w_i^t); (2) w_i^t = \frac{w_i^t}{\sum_{i=0}^{N} w_i^t} \tag{17}$$

Eventually OWA trains the DMN based on Eq. 14 and Eq. 16~ 17. It assigns a small weight to an unlabeled object if it has a small loss as computed in Eq. 14, because a small loss indicates this object is very different from any labeled objects and hence can be easily recognized to be unlabeled. An unlabeled object with a small weight is less likely to have a labeled neighbor and thus violates the Continuity condition. Accordingly, LANCET selects the b objects with the smallest weights for annotators to label.

6.4 Termination Condition

To save the human annotator's efforts, the label candidate selection process should terminate when labeling more objects would not significantly reduce the errors of the automatically produced labels.

In LANCET, the distribution matching network (DMN) used in our label candidate selection method naturally offers an effective yet simplistic way to terminate the labeling process. Specifically, LANCET will suggest the annotators to terminate the labeling process when the following two conditions hold: (1) when the weighted distribution of the unlabeled objects is similar to the distribution of the labeled objects; (2) when the Radon-Nikodym derivative (RND) $\beta(z) = \frac{p_u(z)}{p_l(z)}$ is bounded by a small value β_t and consequently the Continuity condition holds.

Verifying Condition (1) is straightforward. As discussed in Sec. 6.1, when the two distributions are identical, the expected loss $l(z, y_d, w(z))$ (Eq. 11) should be around 1.38 by [33]. Therefore, we can determine if Condition (1) holds simply by checking the value of Eq. 11.

LANCET verifies Condition (2) based on the weight w_i estimated by our online weight approximation (OWA) method for each unlabeled object. By Eq. 10, the weight w_i w.r.t. each unlabeled data z_i corresponds to an approximation of the reverse RND value at z_i. Therefore, if $\forall$ unlabeled object z_i, $w_i > \alpha_t$ where $\alpha_t = \frac{1}{\beta_t}$, then Condition (2) holds.

Intuitively, when α_t is large (or β_t is small), Condition (2) is hard to satisfy. In this case, LANCET is guaranteed to produce a sufficient number of labels to train a robust machine learning model, but potentially wasting some labeling efforts. Based on our experiments, setting α_t around 0.1 or β_t around 10 balances the label sufficiency requirement and the labeling costs in all cases. Therefore, we can apply this same threshold to different datasets without careful tuning.

7 EXPERIMENTS

Our experimental study focuses on the following three questions:

1. **Quality of Generated Labels**: How does LANCET compare with existing labeling approaches in term of the quality of the generated labels?

2. **Accuracy of Trained Machine Learning Models**: How does the performance of the machine learning models trained with labels generated by LANCET compare with the models trained with the labels generated by existing labeling approach?

3. **Ablation Study**: How do the key techniques of LANCET work?

7.1 Experiment Setup

Datasets. We evaluate our methods on classification tasks using in total four benchmark datasets including two classic image datasets and two popular time series datasets described below.

- ***SVHN*** *Street View House Numbers* dataset [27] is a widely used real-world image dataset obtained from house numbers in Google Street View images. Each image contains a digit ranging from 0 to 9; thus it has 10 classes. SVHN consists of 73257 training images and 26032 images for testing. The resolution of each image is 3×32×32.
- ***CIFAR-10*** *benchmark* dataset [20] is a popular image dataset composed of 10 classes of natural scenes with 50000 training images and 10000 testing images. Each is an RGB image of size 32×32.
- ***HAR*** *Human Activity Recognition* dataset [2] is a popular multivariate time series dataset built from the recordings of 30 subjects performing activities throughout their day while carrying a waist-mounted smartphone with embedded inertial sensors. The activities correspond to 6 classes including Sitting, Standing, Walking, etc. The continuous recordings are broken into 10,299 none-overlapping time segments. Each segment is composed of 128 readings over time with each reading a 9-tuple produced by 9 sensors – thus a 9×128 time series.
- ***SpeechCommands*** *[41]* is a public time series dataset used for speech recognition. It has 65,000 utterances of 30 short words by 2,618 speakers. Each utterance is stored as a one-second (or less) WAVE format file encoding the sample data as linear 16-bit single-channel PCM values at a 16 KHz rate. To preprocess the utterances, we first extract normalized spectrograms from the original waveforms and then resize the spectrograms to equalize their sizes at 160×101, as in [41].

Alternative Methods. We compare LANCET against the core state-of-the-art labeling approaches including Snuba [38], GOGGLES [7], and Core-Set [32]. We also compare LANCET against SemiGAN [6] and AutoEncodoer [39] as additional baselines.

- **Snuba** [38] is the state-of-art of weak supervised label generation system. As the successor of Snorkel [30], Snuba uses a small set of manually labeled examples as seeds to produce labels.
- **GOGGLES** [7] is the most recent labeling approach targeting on image data. Because GOGGLES does not support time series data, we compare against GOGGLES using the two image datasets. We use the code made available by the authors.
- **Core-Set** [32] corresponds to the state-of-art ~~of~~ active learning designed for deep neural nets which are used in our experiments.
- **SemiGAN** [6] is the state-of-the-art of semi-supervised feature embedding. It is also able to produce a prediction with respect to each unlabeled object in the given dataset.
- **AutoEncoder** [39]. AutoEncoder is a popular feature embedding method. we develop this baseline by replacing the feature embedding component of LANCET with a pre-trained AutoEncoder model. This new baseline then uses LANCET's label propagation method to produce labels.
- **Ground Truth.** The machine learning models trained with the *ground truth* labels using the deep neural networks are also evaluated. They represent the *best case upper bound* on the prediction accuracy w.r.t each dataset, as they are given apriori all the correct labels instead of first having to infer these labels.

Methodology. We measure the **quality of the produced labels** and the **testing accuracy** of the machine learning models trained on these labels. Following the state-of-the-art [7], in most cases we evaluate LANCET on CNN-based classification problems, because CNNs are known to perform better than other models on complex data such as images or time series. We also run additional experiments to evaluate the performance of LANCET on other model architectures. In particular, we use Support Vector Machines (SVM) as the downstream machine learning model. We run the experiments on the HAR dataset, because SVMs are known to work well on it. We measure the quality of the produced labels following the experimental methodology of Snuba and GOGGLES. That is, given a small set of human supplied labels, we use one label generation approach to propagate labels to all unlabeled objects and then evaluate the *accuracy* of the automatically produced labels. Here accuracy is measured as the ratio of the correctly generated labels over all generated labels.

When evaluating the accuracy of the trained machine learning models, in addition to comparing to Snuba, GOGGLES, SemiGAN, and AutoEncoder, we also compare LANCET against the models trained on the labels supplied by the active learning method Core-Set, as well as the models trained using the ground truth labels.

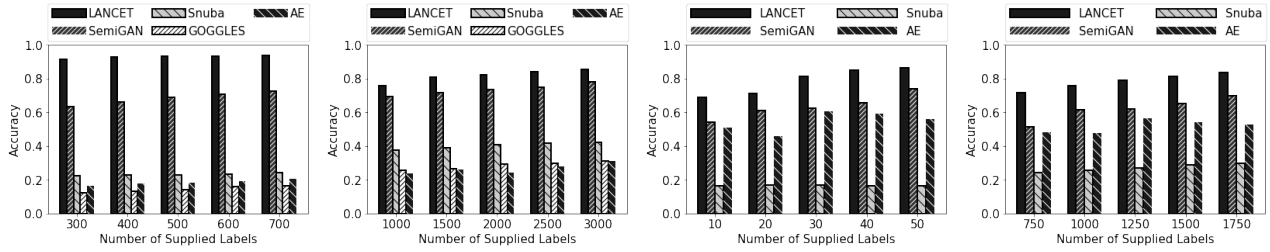
In our ablation study, we evaluate the accuracy of the models trained on the labels produced by different LANCET-based variants to verify the effectiveness of our condition feature mapping (CFM) strategy and the label candidate selection method, which correspond to the key techniques of LANCET. In addition, we also evaluate how active learning works when used together with weak supervision (Snuba + AL).

Finally, we evaluate the termination condition by varying the termination threshold α_t and measuring how the label propagation efficiency of the full-fledged LANCET changes, where propagation efficiency represents the number of *correct* labels automatically inferred per human label.

7.2 The Accuracy of Generated Labels

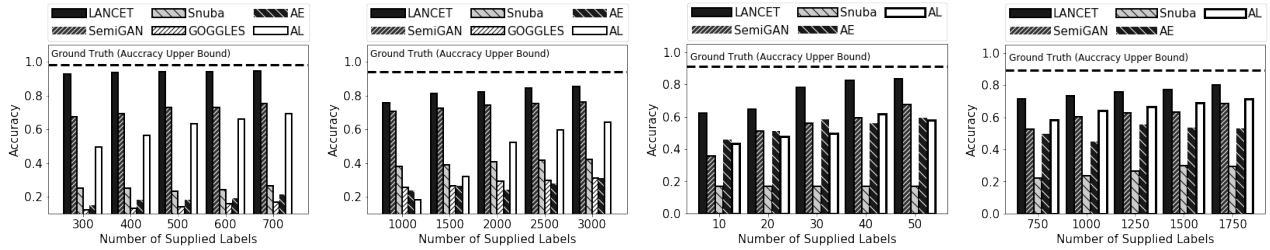
In this set of experiments, we evaluate LANCET, Snuba, SemiGAN, and AutoEncoder on all four image and time series datasets, while GOGGLES was only run on the image datasets, because GOGGLES is specific to image data. When running Snuba on the image datasets, we first use the pre-trained VGG16 model on ImageNet to extract features from the raw image data. We then use principal component analysis (PCA) to project the extracted features into a lower dimensional space with densely rich features, as in the GOGGLES [7] paper. For the time series datasets, Snuba directly uses the raw data as input features to train the models.

All experiments start with a small initial pool of labeled examples randomly sampled from the dataset, corresponding to about 0.5% of the total dataset. The pool of unlabeled data corresponds to the remaining data, from which candidates are selected for the human labelers to annotate. Because all datasets have ground truth available, in our experiments an oracle who already knows the ground truth beforehand simulates human labeler (that is, the oracle will always provide a perfect label when asked). We gradually increase



(a) **SVHN (Image)** (b) **CIFAR-10 (Image)** (c) **HAR (Time Series)** (d) **Speechcommands (Time Series)**

Figure 5: Accuracy of Generated Labels: Varying the Number of Manually Supplied Labels



(a) **SVHN (Image)** (b) **CIFAR-10 (Image)** (c) **HAR (Time Series)** (d) **Speechcommands (Time Series)**

Figure 6: The Accuracy of the Trained Machine Learning Models: Varying the Number of Manually supplied Labels

the number of human supplied labels and evaluate the accuracy of the generated labels. We use the same number of human supplied labels for all methods in all experiments.

As shown in Fig. 5, LANCET consistently outperforms other methods on all datasets with a large margin. Next, we explain where the superiority of LANCET comes.

First, our conditional feature matching (CFM) strategy presented in Sec. 4.1 produces a feature embedding satisfying the Covariate-shift condition (Def. 3.1). That is, in the embedding space, the objects belonging to different classes are well separated; while the unlabeled objects tend to share their label with that of their near-by labeled neighbors. We confirm this by visually examining the plot of objects from each dataset by plotting the feature embedding produced by LANCET using t-SNE, a widely used visualization technique that perceives proximity when mapping to 2-dim space to support visual inspection. Because the feature embedding satisfies the Covariate-shift condition, LANCET is able to effectively propagate labels from labeled objects to their unlabeled neighbors using a lightweight linear model, as analyzed in Sec. 5.

Although SemiGAN also produces a feature embedding that separates objects belonging to different classes, it tends to push the labeled objects far from any unlabeled objects. Thus, it violates the Covariate-shift condition, as depicted in Fig. 2 (a). It is thus ineffective in helping the system to predict the labels of the unlabeled objects, because many of them do not have near-by labeled neighbors to utilize. Therefore, although SemiGAN has been shown to perform better than other baseline methods, especially when handling image datasets, LANCET still consistently outperforms SemiGAN by up to 20 percentage points.

Second, driven by the Continuity condition (Def. 3.2), our distribution matching network-based (DMN) method presented in Sec. 6.3 successfully discovers the areas in the high dimensional embedding space that do not contain enough labeled objects. By selecting objects in these areas for the human domain expert to manually label, LANCET discovers the labeling seeds that are most effective at automatically producing new labels. Therefore, LANCET quickly reaches a high accuracy using very few labels and is able to consistently improve the accuracy of the generated labels as the number of human labels increases, as shown in Fig. 5.

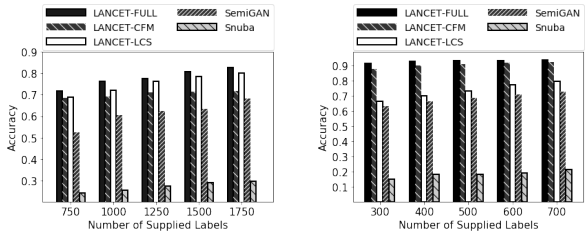
7.3 The Accuracy of Trained Models

As was done in Snuba and GOGGLES, we evaluate the accuracy of the machine learning models trained on the automatically generated labels. For all datasets, we use the standard train/test split from the original source.

For the image classification task, all methods use the popular Preact-ResNet [17] as the downstream machine learning architecture, as was done in GOGGLES [7]. For the timeseries dataset *Speechcommands*, we designed CNN-based deep neural net to classify them by treating them as image data. Our experimental results in 6(d) show it works well. For the timeseries dataset *HAR*, we use Support Vector Machine (SVM) as the downstream machine learning model, because SVM is known to work well on it. We gradually increase the number of human supplied labels until we meet the termination condition of LANCET, where the termination threshold α_t is set to 0.1.

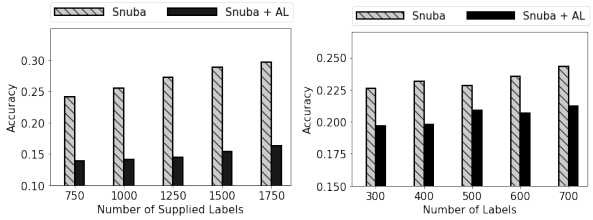
Fig. 6 shows the accuracy of LANCET is at least 40 percentage points higher than that of Snuba and GOGGLES. LANCET also significantly outperforms SemiGAN and AutoEncoder by up to 25 percentage points. This is because the labels produced by LANCET are much more accurate, for the reasons described above in Sec. 7.2.

Although the active learning method Core-Set is able to improve the model accuracy as the number of manual labels increases, LANCET still outperforms active learning in all scenarios. This is because LANCET automatically produces a large number of labels



(a) **Speechcommands (Time Series)** (b) **SVHN (Image)**

Figure 7: Ablation Study



(a) **Speechcommands (Time Series)** (b) **SVHN (Image)**

Figure 8: The Accuracy of Snuba with/without Active Learning

with high accuracy, while a much larger training set tends to produce a more accurate machine learning model when the error rate on the labels is low.

In comparison to the fully supervised model trained on the full set of ground truth data, the accuracy of LANCET is only slightly less than the ideal model by 0.05 to 0.1, while using at most 11% of the ground truth labels as human supplied labels.

In particular, in the case of the SVHN dataset (Fig. 6(a)), LANCET performs almost identically to a model trained on the fully labeled dataset, while it only uses 1% of the manually supplied labels. We believe the superior performance of LANCET on SVHN comes from the high quality feature embedding LANCET produces. To confirm this, we use t-SNE to visualize the feature embedding produced by LANCET and by SemiGAN. We find that LANCET's feature embedding component, that effectively enhances SemiGAN with our conditional feature matching (CFM) strategy, is able to produce a high quality feature embedding where objects from different classes are well isolated from one another and the labeled objects share the same label with their near-by labeled neighbors. Thus it satisfies the Covariate-shift condition (Def. 3.1).

The feature embedding produced by SemiGAN also separates the objects belonging to different classes. Therefore, similar to LANCET, using a small number of manual labels, SemiGAN achieves high accuracy and outperforms other baselines. This indicates that the SemiGAN structure effectively captures the distinct properties of different classes in SVHN and in turn explains the extraordinary performance of LANCET in this case.

7.4 Ablation Study

We conduct an ablation study on SpeechCommand and SVHN datasets. We evaluate the effectiveness of our key techniques including the conditional feature matching (CFM) for feature embedding and label candidate selection (LCS) strategies, subject to different types of datasets (image data and time series data). LANCET-CFM uses our CFM strategy for feature embedding, but randomly samples objects for the humans to label (no label candidate selection). LANCET-LCS uses our label candidate selection method to select labeling candidates, but employs SemiGAN [6] to learn feature embedding model (no CFM). We compare the LANCET-based methods against SemiGAN and Snuba.

As shown in Fig. 7, LANCET-LCS outperforms SemiGAN by up to 17 percentage points. Because LANCET-LCS uses the same feature embedding model to SemiGAN, this confirms that the label candidate selection strategy of LANCET is clearly more effective than randomly picking objects for humans to label.

LANCET-CFM significantly outperforms SemiGAN by up to 26 percentage points. Because LANCET-CFM and SemiGAN both randomly picking labeling candidates, this confirms that our CFM strategy effectively improves the quality of feature embedding.

LANCET-FULL outperforms LANCET-CFM and LANCET-LCS. This shows that our CFM strategy and label candidate selection strategy are compatible with each other, because they are developed based on one unified theoretical foundation in an integrated fashion.

Weak Supervision + Active Learning. This set of experiments also compares Snuba + AL against the original Snuba. The original Snuba uses randomly sampled manual labels as the labeling seeds, while Snuba + AL uses active learning (Core-Set method) to select the manual labels. The results in Fig. 8 show that Snuba + AL performs even worse than the original Snuba, confirming that the active learning methods do not necessarily work well when used together with weak supervision, as noted in the introduction.

7.5 Evaluation of Termination Condition

We evaluate the effectiveness of the termination condition in LANCET. We report the results on CIFAR-10 and Speechcommands datasets. The results on SVHN and HAR datasets show the similar trend.

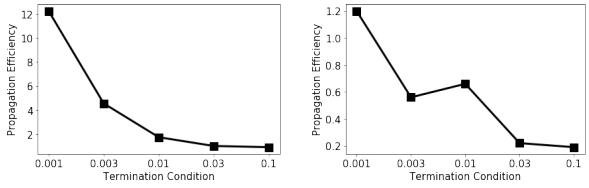
For this, we vary the termination threshold α_t from 0.001 to 0.1. As analyzed in Sec. 6.4, the larger the α_t is, the more human labels will be collected.

To quantify the effectiveness of the acquired human labels in improving the quality of the automatically produced labels, we define a metric, Propagation Efficiency (PE), which measures the number of *correct* labels automatically inferred per human label. For example, if 500 human labels correctly result in the production of 5,000 labels, then the propagation efficiency (PE) is $\frac{5000}{500} = 10$.

When evaluating the termination condition, after reaching each α_t and where the algorithm would normally terminate, we instead conduct one additional round of labeling and measure the PE based on the new acquired human labels. As shown in Fig. 9, the PE decreases as α_t gets higher. In particular, in both cases when α_t is higher than 0.03, the PE gets stable and drops to below 0.5, indicating two human labels only produce one additional label. This confirms the effectiveness of using α_t as the termination condition. Based on the experiments, we recommend the users to set α_t around 0.03 - 0.1.

8 RELATED WORK

Weak Supervision. In recent years, the database community is interested in developing systems and techniques [7, 30, 38] to solve the labeling problem. They aim to use only a small amount of



(a) **Speechcommands (Time Series)** (b) **CIFAR-10 (Image)**

Figure 9: Termination Condition: Varying Threshold α_t

manual labeling to learn good machine learning models. In particular, Snorkel [30] and Snuba [38] use the concept of weak supervision. The goal is to produce high quality labels from a set of labeling sources which produce a large amount of noisy labels. They then produce the final labels by combining the noisy labels using ensemble-based techniques.

Snorkel [30] provides interfaces for users to write labeling functions, which as the labeling sources, produce labels for subsets of data. However, in some scenarios, especially in complex scenarios such as our medical application, designing these labeling functions is hard even for domain experts. Our LANCET instead only requires users to provide some examples for each class, and thus tends to be more user-friendly than Snorkel.

Snuba [38] uses some lightweight machine learning models as labeling sources, such as decision tree or logistic regression, because they are less label thirsty than deep learning and potentially can be trained using a small number of manually supplied labels. However, when applied on complex data such as images and time series, these simplistic models tend to produce poor predictions, as confirmed in our experiments (Sec. 7.2 and 7.3). Therefore, ensemble cannot extract much useful information from them to produce accurate labels. In contrast, because of its feature embedding strategy and the ability of automatically modeling the data distribution in high dimensional space, our LANCET effectively processes complex data.

GOGGLES [7] uses a *transfer learning* inspired method to label image data. Given an input dataset, it first leverages the pre-trained deep learning model for the big ImageNet [8] data to extract features for all instances in the given dataset and calculates the affinity score between each pair of instances. Then, based on this affinity matrix, it infers the labels of the unlabeled images given some manually labeled images. The intuition is that if the instances in the input data resemble some instances of ImageNet, then a fine tune using a small number of labels potentially is sufficient to adapt the pre-trained model to the input data. Therefore, GOGGLES only targets image data. Our LANCET is instead more general and can handle other types of complex data such as time series.

Active Learning. Similar to the label selection of our LANCET, active learning aims to construct a small training set that can train a specific machine learning model with satisfactory accuracy. In general, the active learning methods can be divided into two categories. The first category [10–12, 34] uses metrics to evaluate the importance of the samples to the performance of the current model, such as uncertainty [11], entropy [12] and expected loss [10, 42]. However, it has been shown that such metrics are often hard to estimate in the cutting edge machine learning techniques for example Deep Neural Networks [32]. The second category [32, 34, 40] seeks to form a compact set of unlabeled instances that represents the overall distribution of the entire unlabeled dataset. However, for high dimensional data such as images, it requires a large number of samples to represent such a distribution. Driven by the Continuity condition, LANCET instead cleverly picks specific samples to label that ensure the unlabeled objects always have near-by labeled neighbors. This maximizes the efficacy of automatic label propagation – a critical feature not considered in active learning.

Semi-supervised Classification. Semi-supervised classification leverages the properties of unlabeled data objects to improve the accuracy of machine learning models. Its goal is to classify data as accurately as possible, assuming a small set of labels is available beforehand. Although similar to our work in that it also solves problems caused by a shortage of labeled objects, it tackles classification as a problem rather than the labeling generation problem. Therefore, unlike LANCET, it does not cover the label selection problem, that is, how to select the most informative objects to label.

Some semi-supervised classification techniques [6, 18, 19, 21, 26, 31, 35, 36, 39] use unlabeled data to learn the low dimensional manifold of the dataset. They then train a machine learning model on this low dimensional feature embedding and thus are unlikely to overfit the small labeled dataset. As discussed in Sec. 4.1, our conditional feature matching strategy naturally leverages these techniques to produce feature embeddings that best fit label propagation.

Deep Learning-based Feature Extraction. Deep learning models have been proposed to extract features from complex datasets such as images. In particular, Generative Adversarial Net (GAN) [6, 29, 31] extracts features by forcing a neural network to learn a high density manifold distribution to resist the adversarial attacks from another co-trained neural network. Auto-Encoder [18, 19, 39] methods extract features that are most informative for reconstructing the data objects. Some other works instead design customized heuristics for a certain type of datasets, such as data-augmentation [5, 35], rotation prediction [15], relative patch prediction [28] and colorization [45]. Although all these deep learning-based methods can be used by LANCET to extract features, these methods do not guarantee that the objects belonging to the same class fall into a small region in the learned feature space – essential to LANCET.

9 CONCLUSION

In this work, we tackle the challenging problem of how to produce a sufficient number of labels for data sets void of labels so to train label thirsty machine learning models with minimal manual labeling effort. Our proposed solution, LANCET, solves three critical interdependent subproblems essential for an effective labeling solution, namely, what objects to label, how to automatically produce labels, and when to terminate labeling. LANCET does so in a principle way based on a solid theoretical foundation. Our experiments using multiple public data sets demonstrate that LANCET significantly outperforms all alternative solutions in both accuracy of labels generated and quality of the machine learning models, including weak supervision and active learning based methods.

ACKNOWLEDGEMENTS

This research was supported in part by the National Science Foundation under grants IIS-1910880, CSSI-2103832, and CNS-1852498.

REFERENCES

[1] 2020. LANCET: Labeling Complex Data at Scale (Extended Version). https://drive.google.com/file/d/1cX8l-CyPyoqW4DGrIAgvfaTvRfV4Twx5/view?usp=sharing.

[2] Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge Luis Reyes-Ortiz. 2013. A public domain dataset for human activity recognition using smartphones.. In *Esann*, Vol. 3. 3.

[3] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. 2010. A theory of learning from different domains. *Machine learning* 79, 1-2 (2010), 151–175.

[4] Lei Cao, Wenbo Tao, Sungtae An, Jing Jin, Yizhou Yan, Xiaoyu Liu, Wendong Ge, Adam Sah, Leilani Battle, Jimeng Sun, Remco Chang, M. Brandon Westover, Samuel Madden, and Michael Stonebraker. 2019. Smile: A System to Support Machine Learning on EEG Data at Scale. *Proc. VLDB Endow.* 12, 12 (2019), 2230–2241.

[5] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709* (2020).

[6] Zihang Dai, Zhilin Yang, Fan Yang, William W Cohen, and Russ R Salakhutdinov. 2017. Good semi-supervised learning that requires a bad gan. In *Advances in neural information processing systems*. 6510–6520.

[7] Nilaksh Das, Sanya Chaba, Renzhi Wu, Sakshi Gandhi, Duen Horng Chau, and Xu Chu. 2020. GOGGLES: Automatic Image Labeling with Affinity Coding. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 1717–1732.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*.

[9] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. 2016. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803* (2016).

[10] Melanie Ducoffe and Frederic Precioso. 2018. Adversarial active learning for deep networks: a margin based approach. *arXiv preprint arXiv:1802.09841* (2018).

[11] Yarin Gal and Zoubin Ghahramani. 2016. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*. 1050–1059.

[12] Yarin Gal, Riashat Islam, and Zoubin Ghahramani. 2017. Deep bayesian active learning with image data. *arXiv preprint arXiv:1703.02910* (2017).

[13] Yaroslav Ganin and Victor Lempitsky. 2015. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*. 1180–1189.

[14] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research* 17, 1 (2016), 2096–2030.

[15] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. 2018. Unsupervised representation learning by predicting image rotations. *arXiv preprint arXiv:1803.07728* (2018).

[16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. The MIT Press.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Identity mappings in deep residual networks. In *European conference on computer vision*. Springer, 630–645.

[18] Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. 2016. beta-vae: Learning basic visual concepts with a constrained variational framework. (2016).

[19] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. 2014. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*. 3581–3589.

[20] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).

[21] Samuli Laine and Timo Aila. 2016. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242* (2016).

[22] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436–444.

[23] Anqi Liu, Lev Reyzin, and Brian D Ziebart. 2015. Shift-pessimistic active learning using robust bias-aware prediction. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*.

[24] Xudong Mao, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. 2017. Least squares generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 2794–2802.

[25] R. B. MARIMONT and M. B. SHAPIRO. 1979. Nearest Neighbour Searches and the Curse of Dimensionality. *IMA Journal of Applied Mathematics* 24, 1 (08 1979), 59–70.

[26] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence* 41, 8 (2018), 1979–1993.

[27] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning. (2011).

[28] Mehdi Noroozi and Paolo Favaro. 2016. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*. Springer, 69–84.

[29] Alec Radford, Luke Metz, and Soumith Chintala. 2015. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434* (2015).

[30] Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, Vol. 11. NIH Public Access, 269.

[31] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *Advances in neural information processing systems*. 2234–2242.

[32] Ozan Sener and Silvio Savarese. 2017. Active learning for convolutional neural networks: a core-set approach. *arXiv preprint arXiv:1708.00489* (2017).

[33] Hidetoshi Shimodaira. 2000. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference* 90, 2 (2000), 227–244.

[34] Samarth Sinha, Sayna Ebrahimi, and Trevor Darrell. 2019. Variational adversarial active learning. In *Proceedings of the IEEE International Conference on Computer Vision*. 5972–5981.

[35] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. 2020. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *arXiv preprint arXiv:2001.07685* (2020).

[36] Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. 2016. Ladder variational autoencoders. In *Advances in neural information processing systems*. 3738–3746.

[37] Simon Tong and Daphne Koller. 2002. Support Vector Machine Active Learning with Applications to Text Classification. *J. Mach. Learn. Res.* 2 (March 2002), 45–66.

[38] Paroma Varma and Christopher Ré. 2018. Snuba: Automating weak supervision to label training data. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, Vol. 12. NIH Public Access, 223.

[39] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. 1096–1103.

[40] Keze Wang, Dongyu Zhang, Ya Li, Ruimao Zhang, and Liang Lin. 2016. Cost-effective active learning for deep image classification. *IEEE Transactions on Circuits and Systems for Video Technology* 27, 12 (2016), 2591–2600.

[41] Pete Warden. 2017. Speech commands: A public dataset for single-word speech recognition. (2017).

[42] Donggeun Yoo and In So Kweon. 2019. Learning loss for active learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 93–102.

[43] Yaoliang Yu and Csaba Szepesvári. 2012. Analysis of kernel mean matching under covariate shift. *arXiv preprint arXiv:1206.4650* (2012).

[44] Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. 2016. Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530* (2016).

[45] Richard Zhang, Phillip Isola, and Alexei A Efros. 2016. Colorful image colorization. In *European conference on computer vision*. Springer, 649–666.