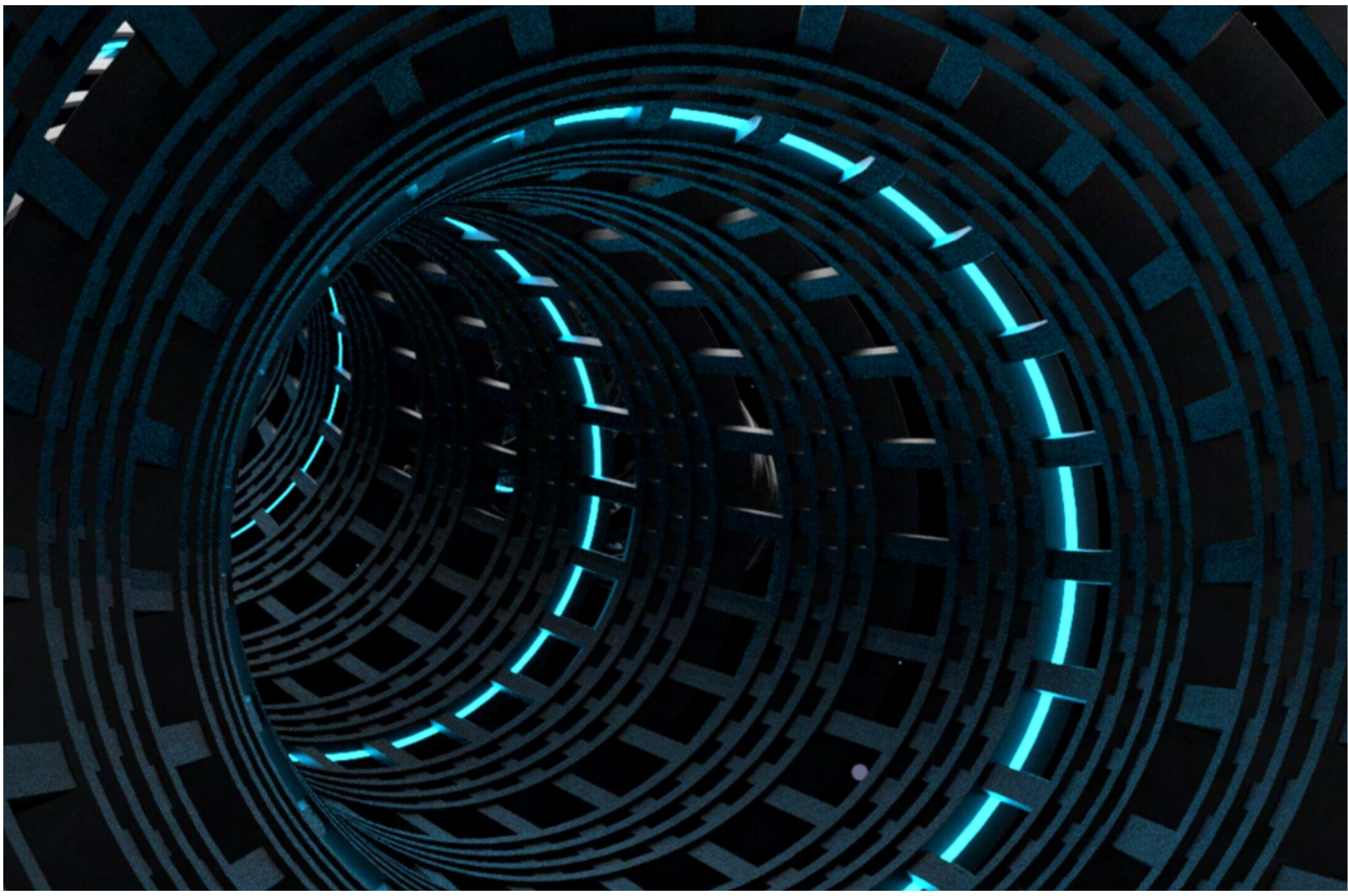
Superb AI

How to Automate Data Labeling using Transfer, Few-Shot, and Self-Supervised Learning



SUPERB-AI

CONTENTS

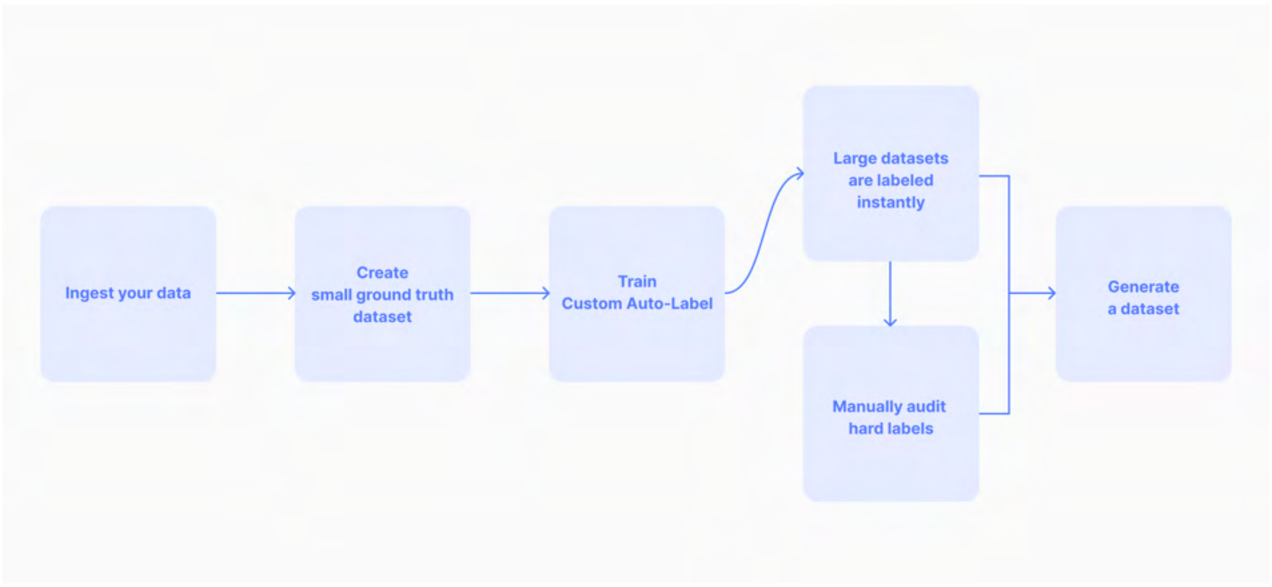
> In previous instances of this whitepaper tech series, we have introduced Superb AI's Custom Auto-Label (CAL) to speed up data labeling efforts and dived into the ML theory developed in-house to evaluate our CAL product. In this whitepaper, we will cover ML techniques underneath our CAL that powers its labeling automation capability.

1 - Labeling Automation 101

Modern machine learning (ML) techniques like deep neural networks have significantly impacted many computer vision tasks, achieving new state-of-the-art performances on benchmarks using little or no feature engineering. However, the conventional paradigm has been to train these systems with supervised learning, where performance has grown roughly logarithmically with labeled dataset size. The cost of such a labeling procedure has proven to be a scalability bottleneck for the continued advancement of state-of-the-art performance and a more fundamental barrier for the deployment of deep neural networks in application areas where labeled data are intrinsically rare, costly, dangerous, or time-consuming to collect.

Previously we have written a primer on data labeling approaches to building real-world ML applications. The nature of data means that there is generally a spectrum of difficulty involved with any labeling task. Some instances will undoubtedly be ambiguous and difficult to label, resulting in mislabeled data. We are bullish that **automatic labeling** is the right way to cut down on the time and labor costs of labeling. Model-assisted labeling (in which datasets are pre-labeled and an AI system is trained to predict annotations for unlabeled data) and AI-assisted labeling (in which AI-assisted software helps the labeler perform manual tasks more efficiently) can significantly improve the accuracy and speed of the labelers on a labeling task.

However, the productivity benefits of labeling assistance are still limited because the labeler still needs to consider and manually label each example.

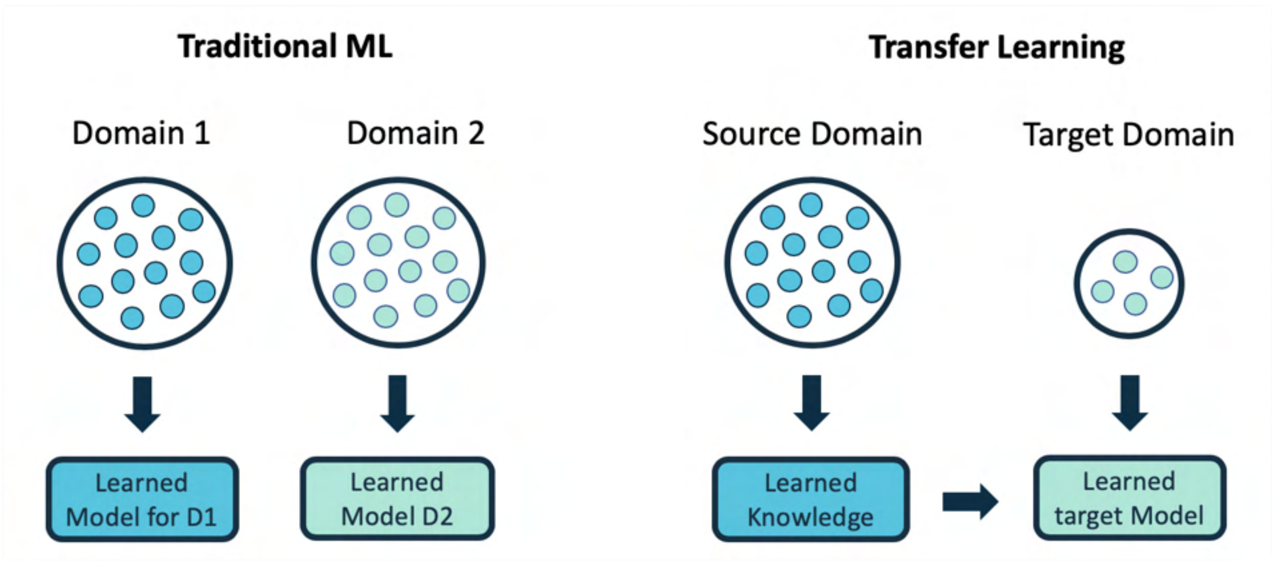

Ingest your data
Create small ground truth dataset
Train Custom Auto-Label
Large datasets are labeled instantly
Manually audit hard labels
Generate a dataset

The workflow of using Superb AI's CAL (Source)

With its accompanying demonstration of Superb AI's Custom Auto-Label (CAL) product, this whitepaper explores transfer learning, few-shot learning, and self-supervised learning techniques. These techniques are the backbone behind our Auto-Label AI underneath CAL, uniquely and scalably designed for labeling automation. More specifically, **our Auto-Label AI not only automatically labels the data but is also capable of assisting users in deciding which labels require manual review** (using difficulty/uncertainty metrics). This also means that CAL helps users select the hard examples, which tend to be more valuable for model training. The human labeler implicitly inspects and corrects the prediction of the Auto-Label AI, which in turn allows the Auto-Label AI to learn and improve. This process continues until the human is satisfied with the labeling performance of the Auto-Label AI and delegates labeling of the remaining data without further intervention.

2 - A Primer on Transfer Learning

In transfer learning, we utilize the available large volume of labeled data to train a model in one domain with easily collected data (source domain) and apply it to another related domain with scarce data (target domain). The learner is assumed to predict the label for a new unseen target data more accurately than a model trained only with limited available target data. This is motivated by the fact that humans can learn faster, easier, and more efficiently using the knowledge from previously learned tasks.



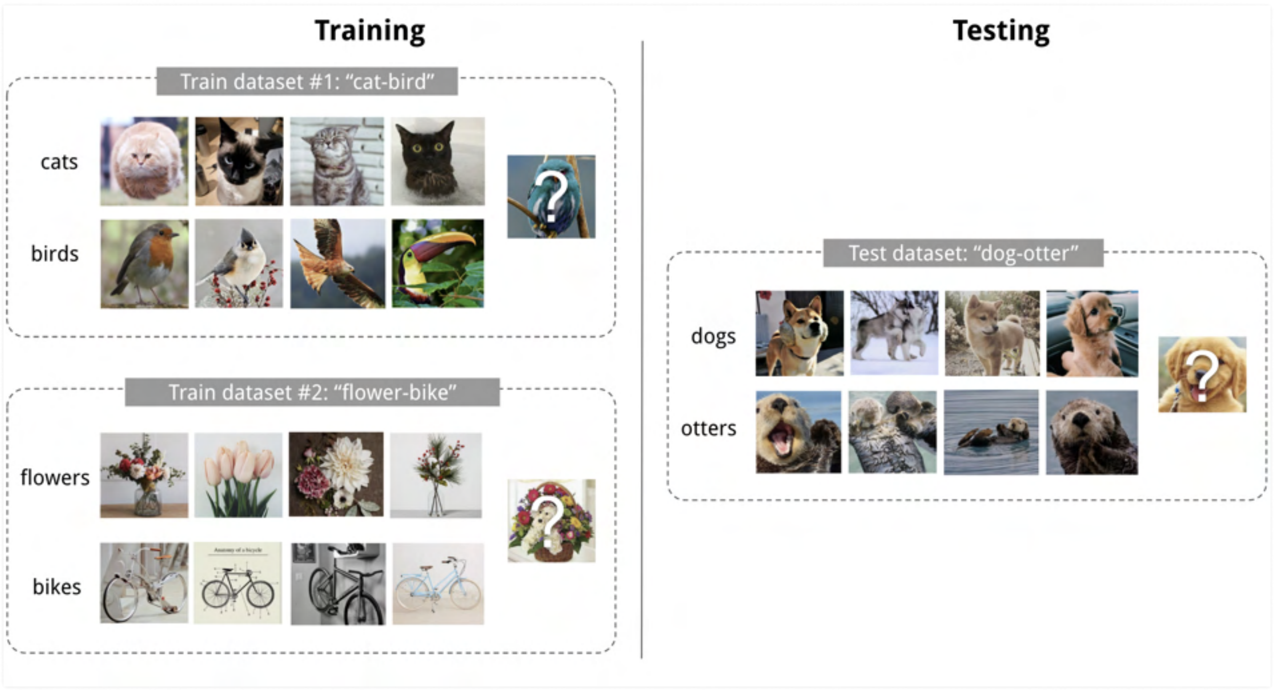
In traditional ML, each model is built based on a single domain in isolation. In transfer learning, the target model is built using the learned knowledge from the source domain (Source)

We can categorize transfer learning into homogeneous and heterogeneous settings regarding different scenarios of feature space and label space across domains.

1. **Homogeneous** transfer learning techniques handle situations where the feature and label spaces are the same across domains, while the marginal and conditional distributions may differ. These approaches aim to diminish the distribution difference between domains by utilizing metrics such as Mean Discrepancy, KL divergence, and correlation alignment to measure the domain divergence and minimize this discrepancy. The main categories of homogeneous transfer learning approaches are instance-based (Kernel Mean Matching, Conditional Probability-Based Multi-Source Domain Adaptation), feature-based (Transfer Component Analysis, Sampling Geodesic Flow, Feature Augmentation Method), parameter-based (Single-Model Knowledge Transfer, Multi-Model Knowledge Transfer), and relational-based (Cross-Domain Adaptation).
2. **Heterogeneous** transfer learning techniques deal with the situation where the feature and label spaces vary between domains. These approaches employ feature-based techniques to transfer knowledge across domains. Feature transformations can be accomplished symmetrically or asymmetrically. Symmetric feature transformation (Feature Augmentation Method, Heterogeneous Feature Augmentation, Cros-Domain Landmark Selection) maps the source and target features into a common latent feature space. In contrast, asymmetric feature transformation (Max-Margin Domain Transforms, Feature-Space Remapping) maps the features from one feature space into another.

3 - A Primer on Few-Shot Learning

In a few-shot learning scenario, you only have access to a very small number of data to train on (i.e., less than 50 samples per class in a classification task). This is an ill-posed problem and almost impossible to achieve high performance without utilizing another dataset based on our current understanding of machine learning. Thus, we can use another auxiliary dataset to learn from it and transfer the learned patterns to the initial small dataset.


Training
Testing
Train dataset #1: "cat-bird"
cats
birds
Train dataset #2: "flower-bike"
flowers
bikes
Test dataset: "dog-otter"
dogs
otters

Example of the meta-learning setup. The left half represents the meta-training set, where inside each gray box is a separate dataset that consists of the training set. The meta-test set in the right half is defined in the same way, but with a different set of datasets that cover classes not present in any of the datasets in the meta-training set (Source)

Most methods in literature address few-shot learning problems as meta-learning problems. The expectation here is that, compared to typical ML models that "learn to distinguish object classes," meta-learning models "learn how to learn to distinguish object classes" so that they can quickly learn to distinguish even more new object classes with a very small number of examples.

We can categorize the few-short learning literature into four methods:

1. **Optimization-based** methods treat inner-level tasks as optimization problems and extract meta-knowledge for inner tasks. This can be viewed as learning good global initial points for inner-level tasks so that in the inner loop, after a few steps of gradient descent, the model can quickly adapt to an unseen task. Examples of optimization-based methods include MAML, Reptile, LSTM Meta-Learner, and MetaOptNet.
2. **Metric learning-based** methods try to 'learn to compare' (compare samples from the query set to the samples from the support set) and obtain a metric function under which data samples in the query and support set stay closer if they are from the same class and stay further if they are from different classes. Examples of metric learning-based methods include Siamese Network, Matching Networks, Relation Network, and Protoptyical Networks.
3. **Model-based** methods don't have an explicit form of the base learner. The meta learner gets samples of each task as input and encodes the information in the internal states of the model. Based on these states, the model makes predictions for the query set. Examples of model-based methods include Memory-Augmented Networks, Neural-Attentive Meta-Learner, Meta Networks, and Imprinted Weights.
4. **Data augmentation-based** methods augment the data to mitigate the data scarcity problem. Sometimes, they learn a data generator using the base dataset and generate data for novel classes during meta-testing. Examples of data augmentation-based methods include Appearance Variance, Style Transfer, Pretrained Saliency Model, and Image Deformation Subnetwork.

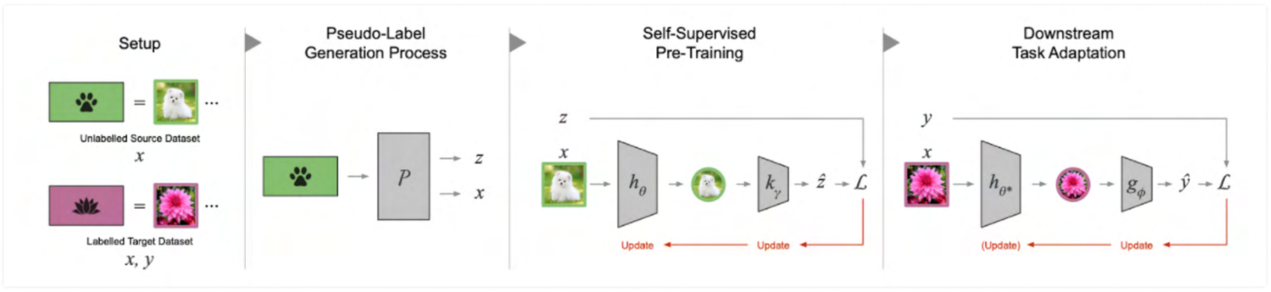
4 - A Primer on Self-Supervised Learning

In self-supervised learning, the model is trained on pseudo-labels that the model can generate by itself. By doing so, the model is learning to perform a proxy task.

For example, you can randomly rotate an image by "x" degrees and give it a pseudo-label of "x." The trained model will be able to predict by how many degrees a given image is rotated. This would be a proxy task for a downstream task such as learning to detect or classify objects within images.

Compared to supervised learning, where the model is trained to predict manually provided target output (i.e., object class and a bounding box around an object, etc.), self-supervised learning models are trained to perform a proxy task based on freely generated pseudo-labels and are then further trained on a downstream task.

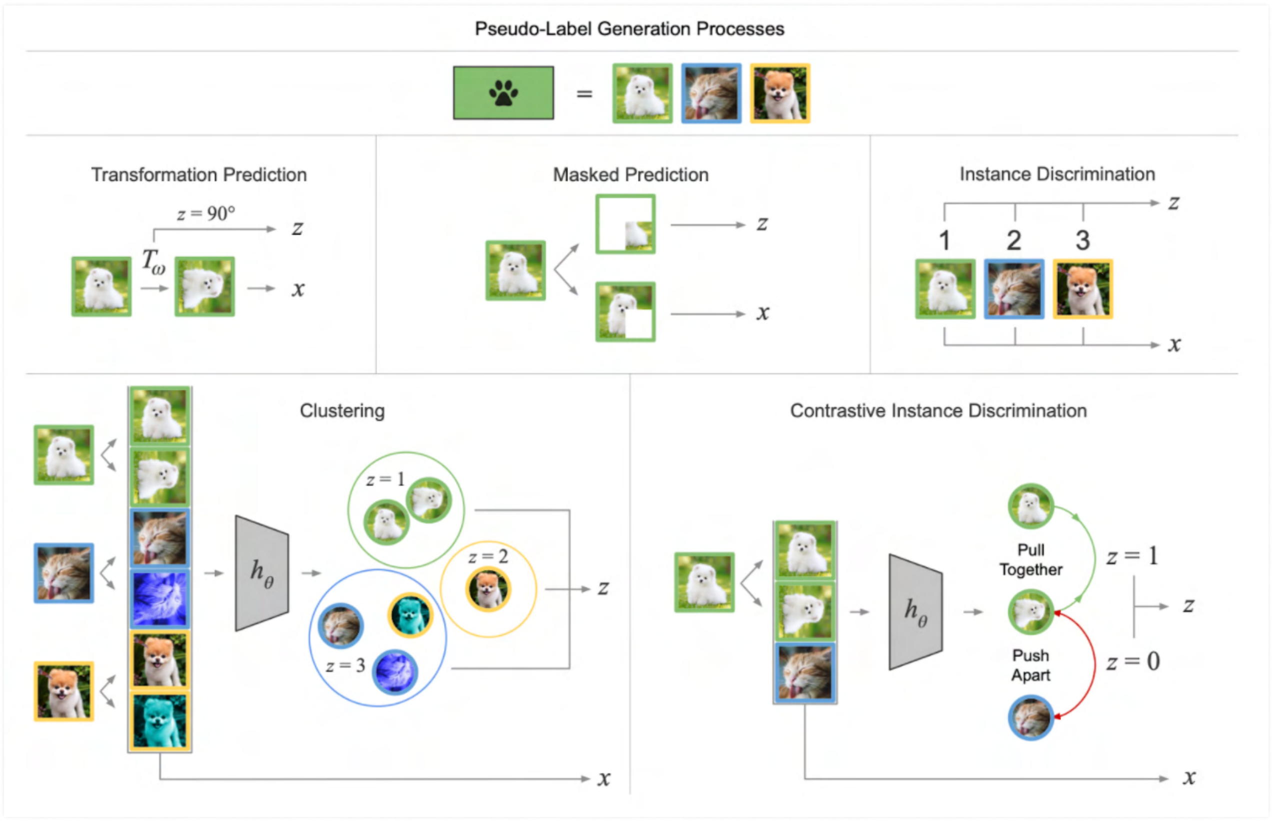
The core idea is that models trained to perform a proxy task using freely or programmatically generated pseudo-labels should be able to learn "representations" (or simply develop a high-level understanding of a given data domain), which can be useful for learning more specific tasks.


Setup
Unlabelled Source Dataset
x
Labelled Target Dataset
x, y
Pseudo-Label Generation Process
P
z
x
Self-Supervised Pre-Training
Update
Update
Downstream Task Adaptation
(Update)
Update

The workflow of self-supervision (Ericsson et al., 2021)

As seen above, the self-supervised workflow starts with an unlabeled source dataset (x) and a labeled target dataset (x, y). As defined by the proxy task (p), pseudo-labels (z) are programmatically generated from the unlabeled set. The resulting inputs (x) and pseudo-labels (z) are used to pre-train the model (composed of feature extractor h and outputs k modules) to solve the pretext task. After pre-training is complete, the learned weights (θ) of the feature extractor (h) are transferred and used together with a new output module (g) to solve the downstream target task.

As the pseudo-labels are created from some intrinsic structure in the data, a model learning to predict those labels must recognize and exploit this structure to solve the task successfully. Thus, self-supervised algorithm design requires and exploits prior human knowledge about the structure in the data to help define meaningful pretext tasks. Furthermore, different pretext tasks will induce different invariance properties in the learned representations, so the choice of method can also be informed by what properties of the representation are required by the downstream task.



Pseudo-labels are generated in the four families of pretext tasks: transformation prediction, masked prediction, instance discrimination, and clustering (Ericsson et al., 2021)

We can categorize the self-supervised learning literature into four broad families as seen above:

1. **Transformation prediction** applies a transformation that maps from a canonical view to alternative views and trains the model to predict what transformations have been applied. It assumes that inputs have a canonical view and that certain transformations can be applied to that view to change it. The canonical view can, for example, depend on the effects of gravity in vision or temporal ordering in video/speech. Common transformation methods include rotating the raw images (in images) or shuffling the temporal order of signal samples (in videos and speeches).
2. **Masked prediction** is characterized by training the model to fill in missing data removed by the pretext task. It assumes that context can be used to infer some types of missing information in the data if the domain is well modeled. Common masking methods involve hiding words in sentences for language modeling, hiding time-slices in speech, hiding regions of images for in-painting, or hiding edges in graphs. Defining an ideal masking strategy is important to use masked prediction effectively.

We can categorize the self-supervised learning literature into four broad families as seen above:

3. **Instance discrimination** treats each instance in the raw source dataset as its own class and trains a model to discriminate between different instances. The discriminative model can be trained with categorical cross-entropy loss, contrastive learning, or regularization techniques. Common instance discrimination methods in computer vision involve MoCo and SimCLR.
4. **Clustering** divides the training data into several groups with high intra-group similarity and low inter-group similarity. It assumes that there exist meaningful similarities by which the data can be grouped. There are multiple ways of determining cluster assignments, such as connectivity, centroids-fitting, likelihood maximization, and more. Common clustering methods in computer vision involve DeepCluster, Online Deep Cluster, and SwAV.

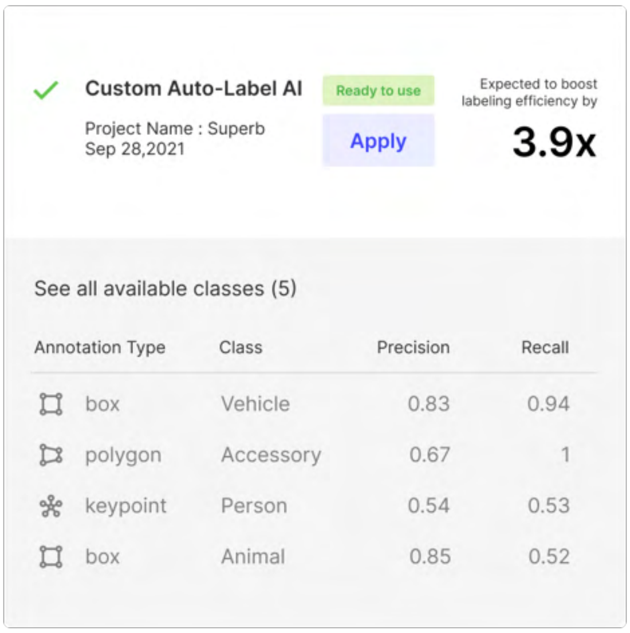
5 - Superb AI's CAL For Labeling Automation

Superb AI's CAL initial Auto-Label feature gave users a pre-trained model that our users can use to label their data. In addition to this, we have expanded to a Custom Auto-Label feature, where a user can easily train and fine-tune a given model using their data. Let's say a user has a very niche dataset, maybe from a specific manufacturing pipeline. And this user wants to have our model to be fine-tuned on their data. So what such users will do is they will manually label maybe the first 100 images and use these labeled images to fine-tune our model using just a few clicks. They hit a button and wait around 30 minutes to an hour, and CAL will produce a model that's fine-tuned to the client's data. From that point on, they can use the new updated model to label the next batch of their dataset. This cycle can repeat, and the CAL model's accuracy should improve with each iteration and each additional batch of data.

CAL uses a combination of transfer learning and few-shot learning to quickly adapt and tailor our proprietary models to your data in your specific application domain. Additionally, CAL utilizes self-supervised learning to pre-train our models on popular application scenarios for computer vision. Suppose you work with niche datasets in highly specialized domains (microscopic imagery, computer graphics, etc.). In that case, you will soon be able to select from our list of pre-trained models that have been self-supervised on each of these application scenarios, which might work well for your domain.

Our customers use CAL with a wide range of data volumes, from a very small amount (e.g., 100 images) to a large enough amount to train a model from scratch (e.g., 100,000 images or more). Customers using CAL with a small amount of data want their models to show good initial performance without memorizing that given small data or overfitting to the training data. Customers using CAL with a large amount of data want their models to achieve high accuracy.

Regardless of which needs the customers have, it's paramount that we help our customers quickly label datasets, audit them to be 100% accurate, and continue iterating on this process and expanding dataset coverage.



Custom Auto-Label AI

Project Name : Superb
Sep 28,2021

See all available classes (5)

Annotation Type	Class	Precision	Recall
box	Vehicle	0.83	0.94
polygon	Accessory	0.67	1
keypoint	Person	0.54	0.53
box	Animal	0.85	0.52

Source: Automate Data Preparation

In order to best achieve the needs of all of our customers, our R&D team at Superb AI has implemented and filed numerous patents for proprietary model architectures and learning algorithms that help users quickly train high-performance models with a small number of training data in a short amount of time.

Below you can find the techniques and algorithms incorporated into CAL that are related to transfer, few-shot, and self-supervised learning. We provide them as a part of our platform so that our users can leverage these advanced ML techniques with only a few clicks.

1. The initial weights of our models are pre-trained by self-supervised learning (inspired by algorithms like BEiT and iBOT). Our users, who handle niche applications like medical imaging, microscopy, or satellite imagery (for which large-scale open-source datasets are not readily available), can fine-tune our pre-trained models instead of starting from scratch.

2. The backbone model for our auto-label AI quickly learns target classes and annotation policies from the customer's data. This model (based on PoseFix) guarantees a good detection performance even with a small amount of data (as long as the customer's data is in the real-world domain and the target classes are covered by the model). For your information, this model covers hundreds of general, common object classes.
3. This backbone model can be fine-tuned by an adaptive training schedule for the amount of the customer's data. We apply data augmentation algorithms (inspired by algorithms like CutMix and Simply Copy-Paste), which are particularly effective for small amounts of data. We also pre-determine the model's hyperparameters through analysis with our benchmark datasets.

Since CAL is used for data labeling purposes, it is designed to maximize algorithmic performance (auto-labeling accuracy) at the expense of inference speed and model size. While being evaluated for the object detection task on a subset of the Microsoft Common Objects in Context dataset, CAL achieved a box average precision performance of 57.1, which is within the top 10 of the state-of-the-art (SOTA) leaderboard for this task. In other words, CAL users can utilize SOTA-performing models tailored to their specific data, which automatically label images at ~600ms each, thereby saving them a ton of time.

Source: COCO Dataset

Many companies have teamed up with Superb AI to label images faster using custom label automation fine-tuned for their exact use case - accelerating time to market for their trailblazing domain-specific solutions.

1. Fox Robotics is a startup that builds agricultural robots to handle routine jobs for soft fruit farms, thereby increasing farmer productivity and crop yields. With the support of our CAL product, they observed a **72% reduction in per-annotation cost** with an average of **5.5x faster labeling speed per image.**
2. Intuitivo is a startup that uses computer vision to deliver the best one-on-one shopping experience. With the support of our CAL product, they were able to improve their in-house **model accuracy by 10+%.**
3. Autonet is a startup that provides image recognition and AI-based decision-making service for damaged vehicle claims processing. With the support of our CAL product, they were able to achieve **98% accuracy** on their car view use case.

6 - Conclusion

In this paper, we have described our Custom Auto-Label workflow and product. We discussed the algorithmic components of the product and the corresponding user experience. Our CAL builds upon cutting-edge transfer, few-shot, and self-supervised learning techniques and presents a novel structure for semi-automated data labeling. These techniques allow us to equip all our users with labeling automation features and customize/fine-tune models for each client in an automated fashion.

Going forward, we plan to further evaluate our CAL as a working paradigm by measuring its utility across a wider range of dataset sizes (from tiny to gigantic ones). We will support a more diverse set of labeling tasks, data types, and application domains. We will also expand our automation across the ML value chain: we started with labeling automation and will expand to data collection/curation/auditing automation.

Superb AI Tech Series: Auto-Labeling

Part 1. Introduction to Superb AI's Auto-Labeling Tech
Part 2. Estimating Auto-Label Uncertainty for Active Learning
Part 3. Automating Data Labeling with Transfer, Few-Shot, and Self-Supervised Learning (You are here)

About Superb AI

Superb AI is an end-to-end training data platform that automates data preparation at scale and makes building and iterating on datasets quick, systematic, and repeatable. Launched in 2018 by data scientists, academics, and ML engineers, Superb AI is reinventing how teams of all sizes label, manage, curate, and deliver training data.

Fueled by decades of experience and academic research in computer vision and deep learning, including 25+ publications, 7,300+ citations, and 100+ patents, Superb AI empowers companies at all stages to build and deploy computer vision applications faster than ever before.

For More Information

To find out how Superb AI can help you build better datasets faster, or get started for free, visit https://www.superb-ai.com/.

Youtube: youtube.com/channel/UCssu44tuxrMb9lePN61xWKA
LinkedIn: linkedin.com/company/superb-ai/
Facebook: facebook.com/superbaihq/
Twitter: twitter.com/superb_hq

Superb AI