Learning Rare Category Classifiers on a Tight Labeling Budget (Supplementary)

Ravi Teja Mullapudi[2,4] rmullapu@cs.cmu.edu
Fait Poms[1] fpoms@cs.stanford.edu
William R. Mark[4] billmark@google.com
Deva Ramanan[2,3] deva@cs.cmu.edu
Kayvon Fatahalian[1] kayvonf@cs.stanford.edu

0.1. Feature update rate

Figure 1 shows variants our approach where we update features with different frequencies. The plot shows average F1 accuracy on 50 iNaturalist categories. The three variants in Figure 1 update the deep feature representation after every 25, 50 and 100 images are labeled by a human and correspond to the following configurations ($N = 5, B = 20, Q = 5$), ($N = 10, B = 10, Q = 5$) and ($N = 20, B = 5, Q = 5$) respectively. As once can see updating the features at low frequency (every 100 human labeled images) results in lower accuracy compared to more frequent updates, especially in the early iterations where only a small amount of data is labeled. However, as one would expect increasing the frequency of updates gives diminishing returns in accuracy since the representation change between iterations would be minimal with a very few additional labeled images.

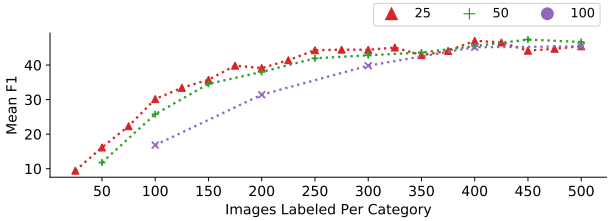

Figure 1. **Increasing feature update rate.** Plot shows average F1 accuracy for variants of our approach on 50 iNaturalist categories as a function of human labeling effort. The variants use different rates (number of labeled images) of updating the deep features. Updating at a low frequency (every 100 images) leads to lower accuracy models, especially in the early iterations. Very high frequency updates (every 25 images) gives diminishing improvements in model accuracy.

0.2. Fraction of auto negatives f_a

Figure 2 shows variants of our approach where we change the ratio (f_a) of pseudo negatives labeled per human labeled image. The plot shows average F1 accuracy on 50 iNaturalist categories and each series corresponds to a different value of f_a. Using a small value for f_a (25) results in slightly lower accuracy. However, our approach is robust to a range of values of f_a and provides significant improvement compared not using pseudo negatives $f_a = 0$.

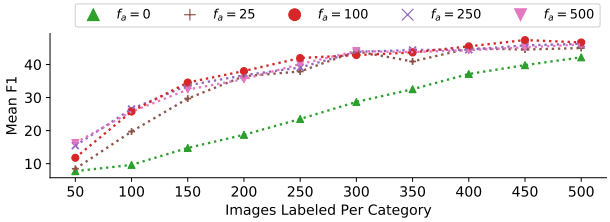

Figure 2. **Varying number of pseudo negatives f_a.** Plot shows average F1 accuracy for variants of our approach on 50 iNaturalist categories as a function of human labeling effort. The variants use different ratio (f_a) of pseudo negatives relative to the human labeled data. Low values of f_a leads to slightly lower accuracy but our overall approach is robust for a wide range of f_a.

0.3. Impact of adaptive sampling

Figure 3 shows variants of our approach where we change the sampling strategy for picking images to query humans. The plot shows average F1 accuracy on 20 Places categories. When there is a sufficient number of positives in the unlabeled data set always sampling the most likely positives is not effective since we would query humans on a lot of easy negatives. This scenario occurs with the Places dataset where there are 5000 positives in the unlabeled data

[1]Stanford University [2]Carnegie Mellon University [3]Argo AI [4]Google Research

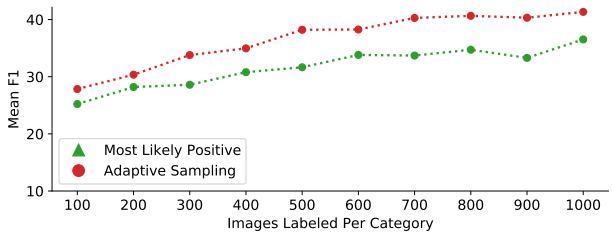

Figure 3. **Impact of adaptive sampling.** Plot shows average F1 accuracy for variants of our approach on 20 Places categories as a function of human labeling effort. Our adaptive sampling strategy for choosing the images to query humans is more effective compared to sampling most likely positives (strategy used by **Tropel**).

Algorithm 1: Integrating baseline pseudo labeling approaches.

Input: U, I_p, M^d_{pre}, B, N, Q, f_a, $method$
Output: $M^d_{1...N}$

1 $F \leftarrow$ **cacheFeatures**(M^d_{pre}, U)
2 $A_l \leftarrow$ **auxiliaryLabels**(M^d_{pre}, U)
3 $c \leftarrow$ **computeFeatureCentroid**(M^d_{pre}, I_p)
4 $R_{pos} \leftarrow$ **rankByCosineSimilarity**(c, F)
5 $L_p \leftarrow I_p$, $L_n \leftarrow \{\}$, $W_n \leftarrow \{\}$
6 $M^d_0 \leftarrow M^d_{pre}$
7 **for** $i \leftarrow 1$ **to** N **do**
8 **for** $j \leftarrow 1$ **to** Q **do**
9 **if** $|L_p| < |L_n|$ **then**
10 $H_p, H_n \leftarrow$ **queryHumanTopK**(B, R_{pos})
11 **else**
12 $H_p, H_n \leftarrow$ **queryHumanTopK**(B, R_{ent})
13 $L_p \leftarrow H_p \cup L_p$, $L_n \leftarrow H_n \cup L_n$
14 $W_n \leftarrow$ **pseudoNegativeLabels**(R_{pos}, f_a)
15 $M^l \leftarrow$ **trainLinearModel**(F,L_p,L_n,W_n)
16 $U \leftarrow U - (H_p \cup H_n)$
17 $R_{pos} \leftarrow$ **sortPositiveScore**(M^l, U)
18 $R_{ent} \leftarrow$ **sortMarginDistance**(M^l, U)
19 **if** $method$ == *Distillation* ***and*** $i > 1$ **then**
20 $W_p, W_n \leftarrow$ **KD-Semi**(M^d_{i-1}, U)
21 **else if** $method$ == *Label Propagation* **then**
22 $W_p, W_n \leftarrow$ **LabelProp**(F, L_p, L_n, U)
23 $M^d_i \leftarrow$ **trainBGSplit**(M^d_{i-1},L_p, L_n, W_p, W_n, A_l)
24 $F \leftarrow$ **cacheFeatures**(M^d_i, U)

for most of the categories. In such scenarios, as Figure 3 shows our adaptive strategy that switches between samples on the margin and most likely positives performs better than a fixed strategy which queries humans on images which are likely to be positives (strategy used by **Tropel**).

0.4. Baseline details

Algorithm 1 shows modifications to our approach to incorporate distillation (**KD-Semi**) and label propagation (**DeepProp**) to pseudo label positives in addition negatives. Lines 19-23 show how we incorporate these pseudo labeling techniques, instead of only using pseudo negatives (W_n) to update the deep representation we use distillation or label propagation to pseudo label both positives and negatives (W_p and W_n) and use them when updating the deep representation. **KD-Semi** uses the deep model from the previous iteration to pseudo label high confidence predictions on unlabeled data (U). **DeepProp** uses the cached features F and current human labeled data L_p, L_n to propagate labels on the unlabeled data (U).

0.5. Training details

When training deep models in the active active loop we train for 15 epochs on the currently labeled data. We start with a learning rate of 0.025 and follow a step schedule for decaying the learning rate every 5 epochs by a factor of 10. When training the fully supervised models we train for 50 epochs and follow a step schedule for decaying the learning rate every 20 epochs by a factor of 10. We use SGD with momentum and a batch size 256 for training all the models. The hyper parameters for the background splitting loss are the same as the ones recommended in the paper, i.e., the weight of the auxiliary loss is set to 0.1.