Appendices

A Number of Classes in Different Hierarchies

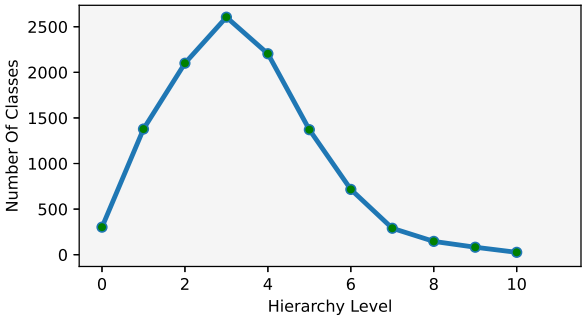

Figure 4: Number of classes in different hierarchies.

B Training Details

B.1 Single-label ImageNet-21K-P Training Details

To better handle the ground-truth inconsistencies of ImageNet-21K-P, we increase the label-smooth factor from the common value of 0.1 to 0.2. As explained in section 2.1, we also use squish-resizing instead of crop-resizing. We trained the models with input resolution 224, using an Adam optimizer with learning rate of 3e-4 and one-cycle policy [48]. When initializing our models from standard ImageNet-1K pretraining (pretraining weights taken from [57]), we found that 80 epochs are enough for achieving strong pretrain results on ImageNet-21K-P. For regularization, we used RandAugment [10], Cutout [12], Label-smoothing [50] and True-weight-decay [37]. We observed that the common ImageNet statistics normalization [33, 51] does not improve the training accuracy, and instead normalized all the RGB channels to be between 0 and 1. Unless stated otherwise, all runs and tests were done on TResNet-M architecture. On an 8xV100 NVIDIA GPU machine, training with mixed-precision takes 40 minutes per epoch on ResNet50 and TResNet-M architectures ($\sim 5000 \frac{\text{img}}{\text{sec}}$).

B.2 Multi-label ImageNet-21K-P Training Details

For multi-label training, we convert each image single label input to semantic multi labels, as described in section 2.2. Multi-label training details are similar to single-label training (number of epochs, optimizer, augmentations, learning rate, models initialization and so on), and training times are also similar. The main difference between single-label and multi-label training relies in the loss function: for multi-label training we tested 3 loss functions, following [3]: cross-entropy ($\gamma_- = \gamma_+ = 0$), focal loss ($\gamma_- = \gamma_+ = 2$) and ASL ($\gamma_- = 4, \gamma_+ = 0$). For ASL, we tried different values of γ_- to obtain the best mAP scores.

C Upstream Results

As we have a standardized dataset with a fixed train-validation split, the training metrics for each pretraining method can be used for future benchmark and comparisons.

C.1 Singe-label Upstream Results

For single-label training, regular top-1 accuracy metric becomes somewhat irrelevant - if pictures with similar content have different ground-truth labels, the network has no clear "correct" answer. Top-5 accuracy metric is more representative, but still limited. Upstream results of single-label training are given in Table 5. We can see that the top-1 accuracies obtained on ImageNet-21K-P, $37\% - 46\%$,

are significantly lower than the ones obtained on ImageNet-1K, $75\% - 85\%$. This accuracy drop is mainly due to the semantic structure and inconsistent tagging methodology of ImageNet-21K-P. However, as we take bigger and better architectures, we see from Table 5 that the accuracies continue to improve, so we are not completely hindered by the inconsistent tagging.

Model Name	Top-1 Acc. [%]	Top-5 Acc. [%]
MobileNetV3	37.8	66.3
ResNet50	42.2	72.0
TResNet-M	45.3	75.2
TResNet-L	45.5	75.6

Table 5: Accuracy of different models in single-label training.

C.2 Multi-label Upstream Results

For multi-label training, we will use the common micro and macro mAP accuracy [3] as training metrics. However, due to the missing labels in the validation (and train) set, this metric also is not fully accurate. In Table 7 we compare the results for three possible loss functions for multi-label classification - cross-entropy, focal loss and ASL. We see that ASL loss [3], that was designed to

Loss Type	Micro-mAP [%]	Macro-mAP [%]
Cross-Entropy	47.3	73.9
Focal loss	47.4	74.1
ASL	48.5	74.7

Table 6: Comparing different loss functions for multi-label classification on ImageNet-21K-P.

cope with large positive-negative imbalancing, outperform cross-entropy and focal loss. This is in agreement with our analysis in section 3.2, where we identify extreme imbalancing as one of the optimization challenges that stems from multi-label training.

C.3 Semantic Softmax Upstream Results

With semantic softmax training, we can calculate for each hierarchy its top-1 accuracy metric. We can also calculate the total accuracy by weighting the different accuracies by the number of classes in each hierarchy (see Figure 4). Notice that we are not using classes above the maximal hierarchy for our metrics calculation. Hence, and unlike single-label and multi-label training, with semantic softmax our training metrics are fully accurate.

In Figure 5 we present the top-1 accuracies achieved by different models on different hierarchy levels, when trained with semantic softmax (with KD).

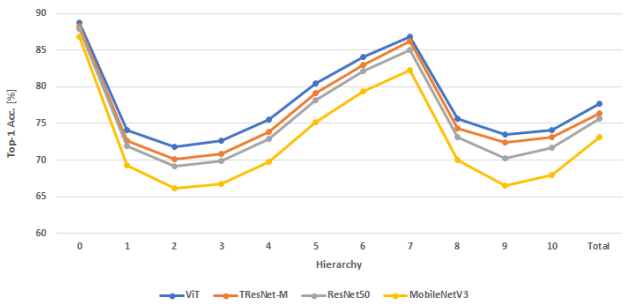

Figure 5: Top-1 accuracies on different hierarchies.

D Downstream Datasets Training Details

For single-label classification, our downstream datasets were ImageNet-1K [30], iNaturalist 2019 [55], CIFAR-100 [29] and Food-251 [25]. For multi-label classification, our downstream datasets were MS-COCO [34] and Pascal-VOC [16]. For video action recognition, our downstream dataset was Kinetics-200 [26].

General details:

- To minimize statistical uncertainty, for datasets with less than $150,000$ images (CIFAR-100, Food-251, MS-COCO, Pascal-VOC), we report result of averaging 3 runs with different seeds.
- All results are reported for input resolution 224.
- For all downstream datasets we used cutout of 0.5, rand-Augment and true-weight-decay of 1e-4.
- All single-label datasets are trained with label-smooth of 0.1
- Unless stated otherwise, dataset was trained for 40 epochs with Adam optimizer, learning rate of 3e-4, one-cycle policy and and squish-resizing.

Specific dataset details:

- ImageNet-1K - Since the dataset is bigger than the others, we finetuned our networks for 100 epochs using SGD optimizer, and learning rate of 4e-4. We used crop-resizing with the common minimal crop factor of 0.08.
- MS-COCO - We used ASL loss with $\gamma_- = 4$.
- Pascal-VOC - We used ASL loss with $\gamma_- = 4$, and learning rate of 5e-5.
- Kinetics-200 - we trained for 30 epochs with learning rate of 8e-5. We used the training method described in [47], with simple averaging of the embedding from each sample along the video.

E Downstream Results for Different Multi-label Losses

In Table 7 we compare downstream results when using multi-label pretraining with vanilla cross-entropy (CE) loss and ASL loss. We see that on all downstream datasets, pretraining with ASL leads to significantly better results

Dataset	Multi Label Pretrain (CE)	Multi Label Pretrain (ASL)
ImageNet1K[(1)]	79.6	**81.0**
iNaturalist[(1)]	69.4	**71.0**
Food 251[(1)]	74.3	**75.2**
CIFAR 100[(1)]	89.9	**90.6**
MS-COCO[(2)]	79.1	**80.6**
Pascal-VOC[(2)]	87.6	**87.9**
Kinetics 200[(3)]	81.1	**81.9**

Table 7: Comparing downstream results for different losses of multi-label pretraining. Dataset types and metrics: (1) - single-label, top-1 Acc.[%] ; (2) - multi-label, mAP [%]; (3) - action recognition, top-1 Acc. [%].

F Calculating Teacher Confidence

Using the teacher prediction for hierarchy i and the semantic ground-truth, we want to evaluate the teacher confidence level, P_i, so we can weight properly the contribution of different hierarchies in the KD loss. Our proposed logic for calculating the teacher's (semantic) confidence is simple:
- If the ground-truth highest hierarchy is higher than i, set P_i to 1.

- Else, calculate the sum probabilities of the top 5% classes in the teacher prediction (we deliberately don't take only the probability of the highest class, to account for class similarities).

In Figure 6 we present the teacher confidence level for different hierarchies, averaged over an epoch.

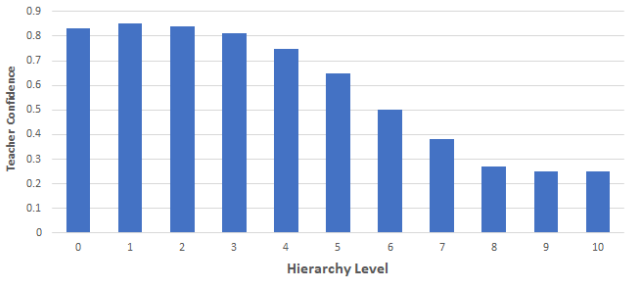

Figure 6: Teacher average confidence levels for different hierarchies.

We can see that lower hierarchies have, in average, higher confidence levels. This stems from the fact that not all hierarchies are relevant for each image. For the picture in Figure 3, for example, only hierarchies 0-5 are relevant, so we expect the teacher will have low confidence for hierarchies higher than 5.

G Semantic KD Vs Regular KD

Dataset	Single Label + KD Pretrain	Sematic Softmax + Semanic KD Pretrain
ImageNet1K[(1)]	81.5	**82.2**
iNaturalist[(1)]	72.4	**72.7**
Food 251[(1)]	76.0	**76.1**
CIFAR 100[(1)]	91.0	**91.7**
MS-COCO[(2)]	81.6	**82.2**
Pascal-VOC[(2)]	89.0	**89.8**
Kinetics 200[(3)]	83.6	**84.4**

Table 9: Comparing KD with different schemes. Dataset types and metrics: (1) - single-label, top-1 Acc.[%] ; (2) - multi-label, mAP [%]; (3) - action recognition, top-1 Acc. [%].

H ImageNet-1K Transfer Learning Results

Model Name	ImageNet-1K + KD Top-1 Acc. [%]
MobileNetV3	78.0
OFA-595	81.0
ResNet50	82.0
Mixer-B-16	82.2
TResNet-M	83.1
TResNet-L	83.9
ViT-B-16	84.4

Table 10: Transfer learning results On ImageNet-1K, when using ImageNet-21K-P pretraining.

I ImageNet-21K-P - Winter21 Split

For a fair comparison to previous works, the results in the article are based on the original ImageNet-21K images, i.e. we are using Fall11 release of ImageNet-21K (*fall11-whole.tar* file), which contains all the original images and classes of ImageNet-21K. After we processed this release to create ImageNet-21K-P, we are left with a dataset that contains 11221 classes, where the train set has 11797632 samples and the test set has 561052 samples. We shall name this variant *Fall11 ImageNet-21K-P*.

Recently, the official ImageNet site[1] used our pre-processing methodology to offer direct downloading of ImageNet-21K-P, based on a new release of ImageNet-21K - Winter21 (*winter21-whole.tar* file). Compared to the original dataset, the Winter21 release removed some classes and samples. The Winter21 variant of ImageNet-21K-P is a dataset that contains 10450 classes, where the train set has 11060223 samples and the test set has 522500 samples. We shall name this variant *Winter21 ImageNet-21K-P*.

For enabling future comparison and benchmarking, we report the upstream accuracies also on this new variant of ImageNet-21K-P:

	Single-Label Training Acc. [%]	Multi-Label Training Macro-mAP [%]	Semantic Softmax Training Acc. [%]
Fall11 ImageNet-21K-P	45.3	74.7	75.6
Winter21 ImageNet-21K-P	47.3	78.7	77.7

Table 11: Upstream results, with different pretraining methods, for different variants of ImageNet-21K-P. Tested model - TResNet-M.

Note that the Winter21 variant of ImageNet-21K-P contains 10% fewer classes and 6% fewer images. In Table 12 we compare downstream results when using Winter21 and Fall11 variants of ImageNet-21K-P

Dataset	Fall11 ImageNet-21K-P	Winter21 ImageNet-21K-P
ImageNet1K(1)	**81.4**	81.2
iNaturalist(1)	**72.0**	71.8
Food 251(1)	**75.8**	75.5
CIFAR 100(1)	90.4	**90.5**
MS-COCO(2)	**81.3**	81.1
Pascal-VOC(2)	89.7	**90.1**
Kinetics 200(3)	**83.0**	82.8

Table 12: Comparing downstream results when using different variant of ImageNet-21K-P. All results are for MTResNet model, with semantic softmax pretraining. Dataset types and metrics: (1) - single-label, top-1 Acc.[%] ; (2) - multi-label, mAP [%]; (3) - action recognition, top-1 Acc. [%].

We can see that compared to Fall11 variant, using Winter21 variant leads to a minor reduction in performances on downstream tasks.

J Additional Ablation Tests

In this section we will bring additional ablation tests and comparisons.

[1] www.image-net.org

J.1 Comparison to Pretraining on Open Images Dataset

Open Images (v6) [31] is a large scale multi-label dataset, which consists of 9 million training images and 9600 labels. In Table 13 we compare downstream results when using two different datasets for pretraining: ImageNet-21K (semantic softmax training) and Open Images (multi-label training).

Dataset	ImageNet-21K Pretrain	Open Images Pretrain
ImageNet1K[(1)]	**81.4**	81.0
iNaturalist[(1)]	**72.0**	70.7
Food 251[(1)]	**75.8**	74.8
CIFAR 100[(1)]	**90.4**	89.4
MS-COCO[(2)]	**81.3**	80.5
Pascal-VOC[(2)]	**89.7**	89.6
Kinetics 200[(3)]	**83.0**	81.6

Table 13: Comparing ImageNet-21K pretraining to Open Images pretraining. Downstream dataset types and metrics: (1) - single-label, top-1 Acc.[%] ; (2) - multi-label, mAP [%]; (3) - action recognition, top-1 Acc. [%].

As we can see, ImageNet-21K pretraining consistently provides better downstream results than Open Images. A possible reason is that Open Images, as a multi-label dataset with large number of classes, suffers from the same multi-label optimization pitfalls we described in section 3.2.

J.2 Comparison on Additional Non-Classification Computer-Vision Tasks

In Table 14 and Table 15 we compare 1K and 21K pretraining on two additional computer-vision tasks: object detection (MS-COCO dataset) and image retrieval (INRIA holidays dataset).

	1K Pretraining	21K Pretraining
mAP [%]	42.9	44.3

Table 14: Comparing downstream results on MS-COCO object detection dataset.

	1K Pretraining	21K Pretraining
mAP [%]	81.1	82.1

Table 15: Comparing downstream results on on INRIA Holidays image retrieval dataset.

We can see that also on non-classification tasks such as object detection and image retrieval, pretraining on ImageNet-21K translates to better downstream results than ImageNet-1K pretraining.

J.3 Impact of Different Number of Training Samples

In Figure 7 we test the impact of the number of training samples on on the upstream accuracies. As we can see, there is no saturation - more training images lead to better semantic accuracies.

K Pseudo-code

In the following sections we will bring pseudo-code (PyTorch-style) to some components in our semantic softmax training scheme: logits sampling, KD calculation and estimating teacher confidence.

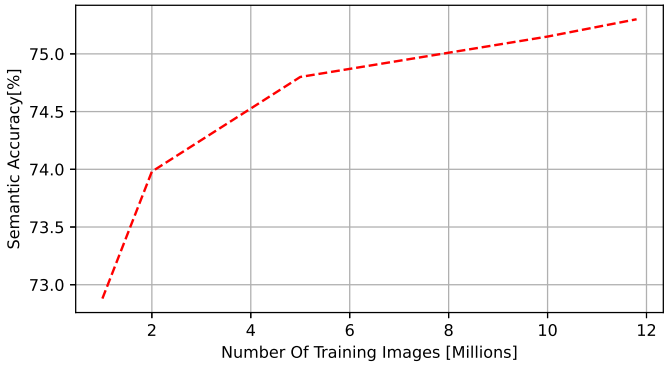

Figure 7: Upstream results for different number of training images.

K.1 Logits Sampling

```
def split_logits_to_semantic_logits(logits, hierarchy_indices_list):
    semantic_logit_list = []
    for i, ind in enumerate(hierarchy_indices_list):
        logits_i = logits[:, ind]
        semantic_logit_list.append(logits_i)
    return semantic_logit_list
```

K.2 KD Logic

```
def calculate_KD_loss(input_student, input_teacher, hierarchy_indices_list):
    semantic_input_student = split_logits_to_semantic_logits(
    input_student, hierarchy_indices_list)
    semantic_input_teacher = split_logits_to_semantic_logits(
    input_teacher, hierarchy_indices_list)
    number_of_hierarchies = len(semantic_input_student)

    losses_list = []
    # scanning hirarchy_level_list
    for i in range(number_of_hierarchies):
        # converting to semantic logits
        inputs_student_i = semantic_input_student[i]
        inputs_teacher_i = semantic_input_teacher[i]

        # generating probs
        preds_student_i = stable_softmax(inputs_student_i)
        preds_teacher_i = stable_softmax(inputs_teacher_i)

        # weight MSE-KD distances according to teacher confidence
        loss_non_reduced = torch.nn.MSELoss(reduction='none')(preds_student_i,
        preds_teacher_i)
        weights_batch = estimate_teacher_confidence(preds_teacher_i)
        loss_weighted = loss_non_reduced * weights_batch.unsqueeze(1)
        losses_list.append(torch.sum(loss_weighted))

    return sum(losses_list)
```

K.3 Teacher Confidence

```
def estimate_teacher_confidence(preds_teacher)
    with torch.no_grad():
        num_elements = preds_teacher.shape[1]
        num_elements_topk = int(np.ceil(num_elements / 20))  # top 5%
        weights_batch = torch.sum(torch.topk(preds_teacher,
        num_elements_topk).values, dim=1)
    return weights_batch
```

L Limitations

In this section we will discuss some of the limitations of our proposed pipeline for pretraining on ImageNet-21K:

1) While our work did put a large emphasis on the efficiency of the proposed pretraining pipeline, for reasonable training times we still need an 8-GPUs machine (1 GPU training will be quite long, 2-3 weeks).

2) For creating an efficient pretraining scheme, and also to stay within our inner computing budget, we did not incorporate training tricks that significantly increase training times, although some of these tricks might give additional benefits and improve pretraining quality.

An example - techniques for dealing with extreme multi-tasking, such as GradNorm [7] and PCGrad [60], that would probably improve the pretrain quality of multi-label training, but would significantly increase training times.

Another example of methods from the literature we have not tested - general "semantic" techniques that can be used for training neural networks ([4, 54] for example). We found that most of these techniques are not feasible for large-scale efficient training. In addition, we believe that since our novel method, semantic softmax, is designed and tailored to the specific needs and characterizations of ImageNet-21K, it will significantly outperform general semantic methods.

3) When using private datasets which are larger than ImageNet-21K, such as JFT-300M [49], the pretrain quality that can be achieved is probably still higher than the one we offer.