ProSub: Probabilistic Open-Set Semi-Supervised Learning with Subspace-Based Out-of-Distribution Detection

Supplementary Material

6 Qualitative Analysis of Feature Separation

To further analyze the effects of the self-supervision, ℓ_{self} from (10), and the subspace loss, ℓ_{sub} from (8), we plot t-SNE (Maaten and Hinton, 2008) reductions of features from ID and OOD test sets for a few different training setups. These experiments are done with CIFAR-100 as ID and CIFAR-10 as OOD. First, we train a fully supervised model using all 50,000 training data (with labels) from CIFAR-100. This model is never exposed to OOD data. Secondly, we train ProSub using 10,000 labels and first train until the end of the warm-up phase. At this stage, the model has only been trained with the labeled cross-entropy, ℓ_{sup} from (11), and self-supervision, ℓ_{self}. Finally, we carry out a full training run of ProSub, where ℓ_{semi} from (9) and ℓ_{sub} are applied after the warm-up stage.

The results are shown in Fig. 5. From the top panel, we see that the fully supervised model successfully clusters the ID data in feature space. However, most OOD data are not clustered or separated from ID, highlighting the challenge of OOD detection when we do not receive learning signals from these data.

The next evaluated model is ProSub at the end of the warm-up. This model is trained with fewer labeled data than the fully supervised model but is exposed to both (unlabeled) ID and OOD through self-supervision. OOD data now begin to form distinct clusters, visibly separated from ID data. This suggests that self-supervision facilitates the clustering of both ID and OOD data. Visually, it seems reasonable to believe that OOD detection in this feature space is easier than for the fully supervised case. However, there are still regions where ID and OOD are mixed.

Finally, we have the features from the fully trained ProSub. Now we see even more clear and separated clusters for both ID and OOD data, indicating that the subspace loss further contributes to forming separated clusters for ID and OOD. An interesting observation is that OOD forms multiple clusters instead of one, even though this is not explicitly encouraged by either the self-supervision or the subspace loss. This indicates that the model not only learns to separate ID from OOD but also learns to group data within OOD.

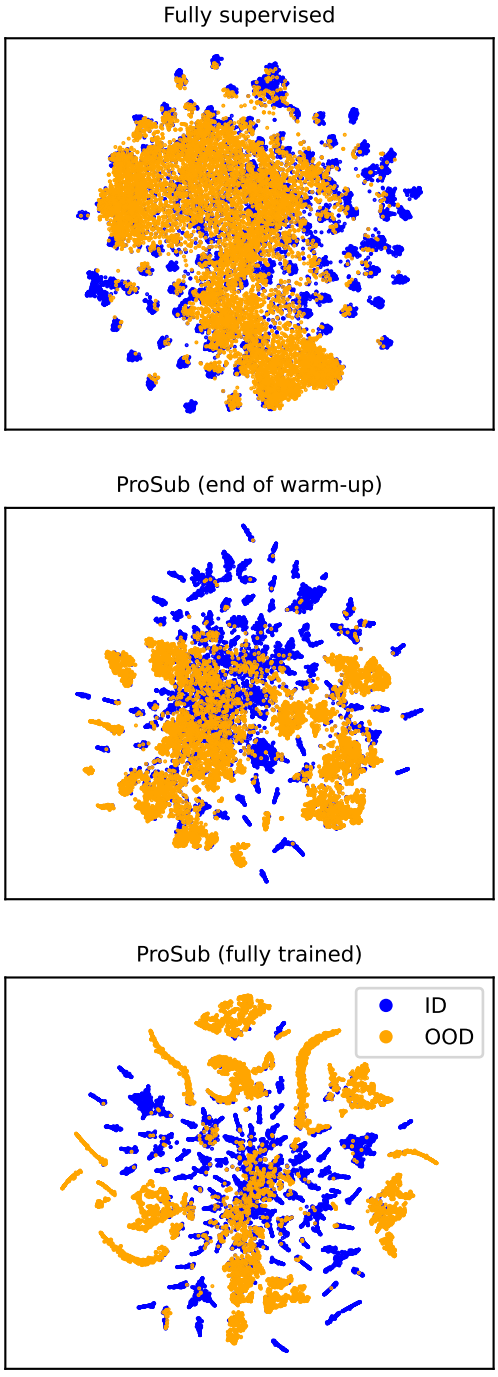

Fig. 5: t-SNE of features. ID: CIFAR-100, OOD: CIFAR-10.

Table 5: Evaluating OOD detection on unseen OOD using TIN.

		AUROC	
	Accuracy	Seen OOD	Unseen OOD
OpenMatch	56.51	0.69	0.69
SeFOSS	64.09	0.68	<u>0.74</u>
ProSub	**66.06**	<u>0.80</u>	0.71

7 Experiments with Unseen Outliers

Tab. 1 evaluates AUROC of OOD detection on classes present in the unlabeled training set (seen OOD). While this is a core metric of OSSL performance, we also find value in exploring OOD detection for classes completely unseen during training (unseen OOD). To simulate this scenario, we divide Tiny ImageNet into three parts: 70 ID classes, 70 OOD classes present in the unlabeled training data, and 60 OOD classes entirely unseen during training. We use 3,500 labels. For this setting, we evaluate OpenMatch, SeFOSS, and ProSub. The results are shown in Tab. 5. We see that ProSub drops in AUROC when going from seen to unseen OOD, indicating that the losses applied to OOD data facilitate learning features to discriminate between ID and seen OOD specifically. In contrast, OpenMatch obtains consistent AUROC for both seen and unseen OOD and SeFOSS shows *better* AUROC for unseen OOD. However, despite this, ProSub demonstrates competitive results in OOD detection for unseen OOD.

8 Sensitivity Analysis of π

The probabilistic ID/OOD predictions of ProSub (see (1)) require specifying the proportion of ID data in unlabeled data, π. In the experiments conducted for this work, we use exact values of π, which is $\pi = 0.5$ for all scenarios except ImageNet20/10 where it is $\pi = 0.66$. While it may be hard to know the exact value of π in practice, we argue that it is easy to get an approximation by inspecting a subset of unlabeled data. If this approximation is unavailable, one can treat π as a hyperparameter. To study how the performance of ProSub varies with π, we conduct experiments with CIFAR-100 as ID (10,000 labels) with CIFAR-10 as OOD using different values of π. With this setup, $\pi = 0.5$ corresponds to the true proportion of ID data in unlabeled data.

Fig. 6 shows closed-set accuracy and AUROC as a function of π. We see that the obtained accuracy shows minimal dependency on π. The AUROC, interestingly, exhibits a stable high value as long as π does not exceed 0.5. This suggests avoiding misclassifying OOD as ID is more crucial than the reverse. One possible explanation is that the cross-entropy for labeled data (or from pseudo-labeling) acts as an "anchor" for ID data, counter-acting the subspace loss that pushes these data away from W_{id}. No such counterweight exists if OOD data are pushed towards W_{id}, making this type of error more detrimental.

To show that we do not make significant performance gains from knowing the exact value of π, we include results on some datasets where ProSub displays the best results in Tab. 1. In these experiments, we use $\pi = 0.4$, *i.e.*, lower than the true portion of ID data in the unlabeled data. These results are shown in Tab. 6, revealing that using an incorrect π does not significantly impact our results. For TIN, the accuracy is slightly lower when using $\pi = 0.4$, however, it is still higher than competing methods.

As a practical recommendation, we suggest using a π slightly lower than the approximation obtained from unlabeled data to avoid exceeding the true proportion.

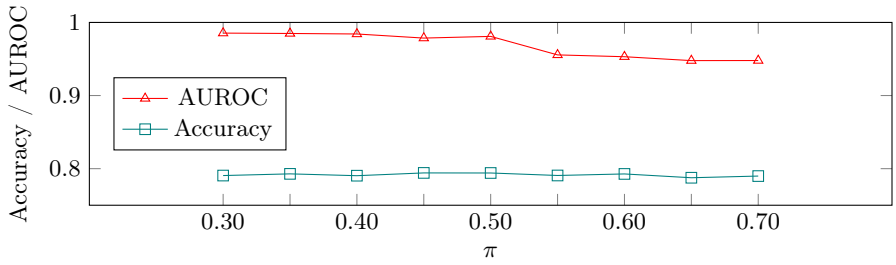

Fig. 6: Analyzing how ProSub performance depends on π ($\pi = 0.5$ corresponding to the true value).

Table 6: Results from using an offset π: $\pi = 0.4$.

	ID: CIFAR-100 (10,000 lab.) OOD: CIFAR-10	IN50/50	TIN100/100
ProSub (correct π)	79.59$\pm$0.37 0.98$\pm$0.00	71.15$\pm$0.80 0.96$\pm$0.00	60.92$\pm$0.32 0.72$\pm$0.00
ProSub ($\pi = 0.4$)	79.54 0.98	71.48 0.96	59.96 0.72

9 Hyperparameters

The values of most hyperparameters used in ProSub are gathered from existing works and used without further tuning. For example, we use $w_{\text{semi}} = 1.0$ and initial learning rate $\eta_0 = 0.03$, l^2-regularization w_{reg}, decay rate γ, EMA momentum, batch sizes, and SGD momentum following [51,56]. For the evaluations done on TIN100/100 (new for this work), we copy the values for w_{reg} and γ used for CIFAR-100 in [56] ($w_{\text{reg}} = 0.001$, $\gamma = 5/8$) because of the equal number of

ID classes. For ImageNet50/50 (also new for this work) we copy the values for w_{reg} and γ used for ImageNet20/10 in [56] ($w_{\text{reg}} = 0.0005$, $\gamma = 7/8$).

The main hyperparameter introduced for ProSub is w_{sub}, the weight for the subspace loss. We empirically find that $w_{\text{sub}} = 1.0$ works well across all evaluated datasets. Secondly, we use the cosine-based self-supervision from [55] that shows w_{self} can need dataset-specific tuning, which is why we use varying values of w_{self}.

9.1 Selecting Hyperparameters Using Validation Data

The hyperparameters we tune for ProSub are w_{self} and w_{sub}. Since labeled data are limited in OSSL, we suggest using a subset of labeled data as validation data to tune w_{self} and w_{sub}. Subsequently, these tuned values can be utilized in a training run using all available labeled data for training.

We illustrate this procedure using CIFAR-100 as ID (10,000 labels) with CIFAR-10 as OOD by using 5,000 labels for training and 5,000 for validation. Table 7 shows that $w_{\text{sub}} = 1.0$ and $w_{\text{self}} = 15.0$ yield the best validation accuracy among the evaluated values. Additionally, Tab. 7 shows that these values correspond to the best accuracy on the test set. Notably, the closed-set accuracies align reasonably well with the obtained AUROC, simplifying hyperparameter selection as AUROC cannot be evaluated directly from the validation set.

The gap in accuracy between the validation set and the test set arises from labeled data (and consequently validation data) being included in the unlabeled training set without labels. To obtain an absolute prediction of test accuracy (rather than a relative one), the validation data can be explicitly excluded from the unlabeled set.

Table 7: Tuning hyperparameters from validation data.

Validation results			
	w_{self}		
w_{sub}	5.0	15.0	25.0
0.1	86.82	87.58	88.12
1.0	86.66	**88.56**	88.36
10.0	52.00	79.44	85.84

Test results			
	w_{self}		
w_{sub}	5.0	15.0	25.0
0.1	72.67 0.96	75.72 0.97	75.08 0.96
1.0	72.76 0.86	**77.25** 0.98	77.15 0.98
10.0	12.25 0.58	58.78 0.67	71.43 0.86

9.2 The Number of Training Steps

We set the number of training steps, K, to obtain reasonable training times, which is why we use a lower number of training steps for the ImageNet experiments. We have not observed any issues with overfitting or training collapse.

The best performance is generally achieved at the end of training as shown in Fig. 7. This figure shows test accuracy and AUROC as a function of training steps for a run on ImageNet50/50. This likely means that increasing the number of training steps should obtain equal or better results.

We have set the number of warm-up steps, K_p, to be a small but non-trivial fraction of the total number of training steps. Table 8 shows results on ImageNet50/50 with varying K_p and a fixed $K = 10^5$, showing that the results are insensitive to the choice of K_p.

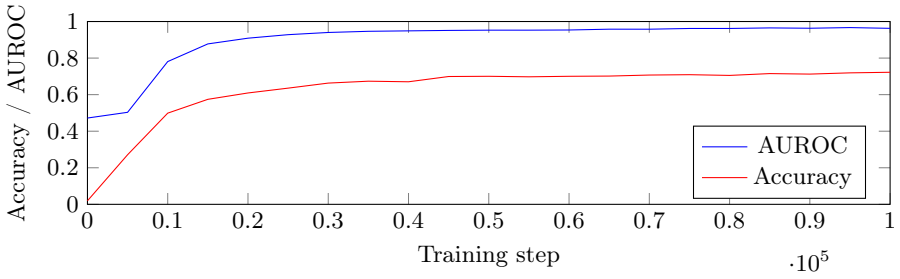

Fig. 7: ImageNet50/50 performance vs. training steps.

Table 8: Varying K_p on ImageNet50/50 (with $K = 10^5$).

$K_p/10^3$	15	20	25	30	35	40
Acc	71.92	72.52	71.44	71.15	71.40	71.60
AUROC	0.96	0.96	0.96	0.96	0.96	0.96

9.3 Fine-grained Hyperparameter Sensitivity

To further analyze the sensitivity of hyperparameters w_{sub} and w_{self} we run experiments on ImageNet50/50 with varying w_{sub} and w_{self}. Figure 8 shows that the results drop when we go far away from the values used to generate the main results in Tab. 1, but there are relatively large ranges for both w_{sub} and w_{self} where the results are stable.

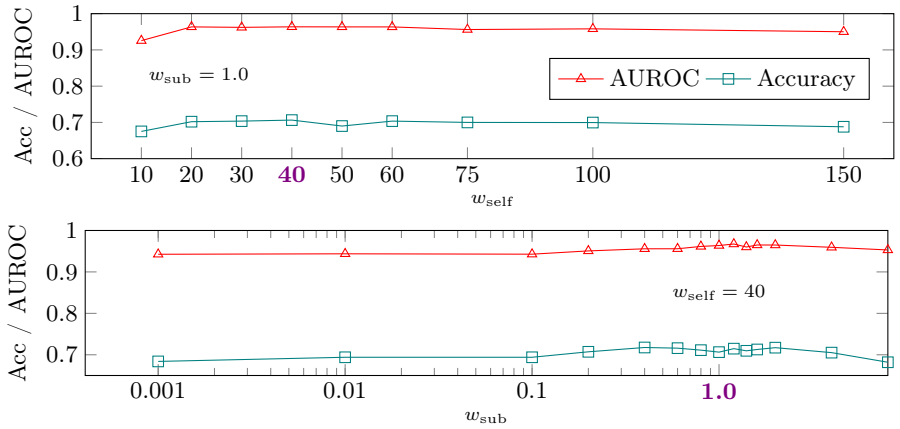

Fig. 8: Hyperparameter evaluations on ImageNet50/50 test sets. **Violet** marks values used for Tab. 1.

9.4 Initiation of Beta Parameters

For the IMM estimation, we use the initial guess $\alpha_{\text{id}} = \beta_{\text{ood}} = 10$, $\alpha_{\text{ood}} = \beta_{\text{id}} = 2$. We make this choice to ensure that the estimate for the ID distribution lies closer to 1.0 than the OOD distribution. However, because of the warm-up phase, the estimates have time to improve and settle before they are used to generate training signal through ℓ_{semi} (9) and ℓ_{sub} (8). We have not found the initiation of these parameters to be significant for our performance.

10 Varying ID/OOD Ratios in Unlabeled Data

In the experiments of Tab. 1, most of our benchmark problems have equal amounts of ID and OOD in the unlabeled set. Here, we study how ProSub performs with varying ratios of ID to OOD data in the unlabeled set. Figure 9 shows closed-set accuracy and AUROC for ProSub with varying OOD frequencies. We let π follow the true ID/OOD ratio. For these experiments, we use CIFAR-100 (2,500 labels) as ID with CIFAR-10 as OOD, and ImageNet50/50. As expected, AUROC increases with more OOD data because the exposure to OOD data through self-supervision enables better OOD detection (see Sec. 4.3). Conversely, closed-set accuracy drops as ID data decreases due to fewer pseudo-labels that help us learn the ID classes. The results indicate optimal OOD frequencies around 0.4 - 0.5 that yield the best results for both OOD detection and closed-set accuracy. However, the OOD frequency is difficult to control in real-world scenarios.

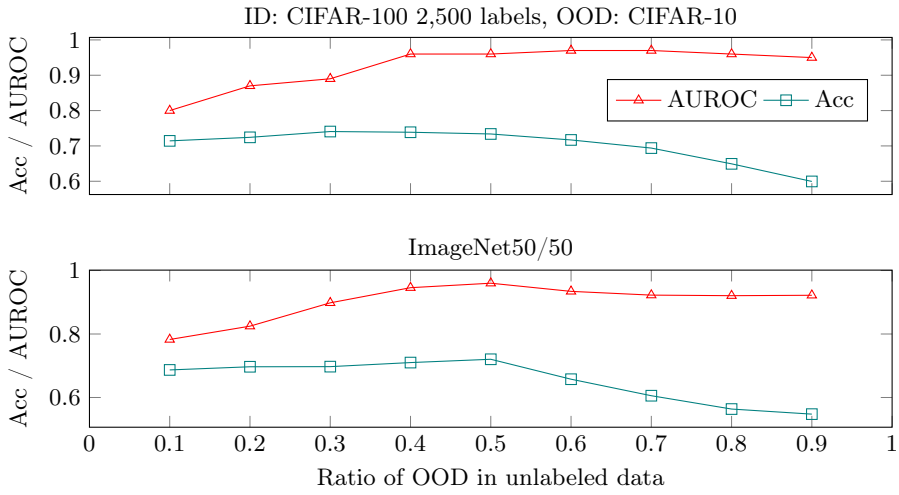

Fig. 9: ProSub performance with varying ratios of ID and OOD in the unlabeled set.

11 Regularization of ID Probabilities

Based on the observation in Sec. 8 that avoiding misclassifying ID as OOD is more important than the reverse, we find it beneficial to regularize the ID probabilities when computing the random mask in (7). This is achieved by adding a constant, ϵ, to the denominator of (1) as

$$p(\mathbf{x} \in \mathcal{ID} | s(\mathbf{z})) = \frac{\pi p_{\text{id}}(s(\mathbf{z}))}{\pi p_{\text{id}}(s(\mathbf{z})) + (1-\pi) p_{\text{ood}}(s(\mathbf{z})) + \epsilon}. \tag{14}$$

We have found $\epsilon = 0.1$ to be a suitable value. Note that this regularization is used only for computing the random mask and not in the IMM estimation.

12 Limitations

Section 7 shows ProSub's superior performance on OOD detection for *seen* OOD specifically. While ProSub remains competitive for unseen OOD detection, other methods may perform better if unseen OOD detection is your most important metric. Furthermore, this work only considers datasets that are balanced in terms of classes. We do not know how big shifts in class balances impact our performance. Finally, a limitation of ProSub lies in its dependence on dataset-specific tuning of w_{self} and the necessity to tune π or approximate the proportion of ID data within the unlabeled data.

13 Score Distributions and Estimates

In Sec. 4.2 and Fig. 3 we look at the distributions of scores and the corresponding estimates at two different time steps during training. Here, in Fig. 10, we show

the equivalent evaluations at more time steps during training to display how the distributions and their corresponding estimates progress. These results are from a run using CIFAR-100 (2,500 labels) as ID with CIFAR-10 as OOD. The current training step is denoted by k and the warm-up phase runs for 50,000 steps.

Figure 10 shows that during the warm-up phase, most data stay fairly close to W_{id}, but as training progresses, we start to distinguish between ID and OOD when the distribution of OOD moves slowly away from W_{id}. Interestingly, despite the overlapping distributions, the estimated Beta distributions accurately capture the individual mixture components throughout the warm-up phase.

After the warm-up phase (indicated by the horizontal black dashed line in Fig. 10), when we apply ℓ_{sub} from (8), we see that the distribution of scores for OOD data quickly moves away from W_{id} (lower scores). The distribution of scores for ID data similarly moves closer to W_{id} (higher scores). The estimated Beta distributions adapt well to this sudden change.

However, we also see that a few OOD data incorrectly get scores close to 1.0, highlighting that our obtained ID/OOD classifier does not have perfect accuracy. Notably, the set of OOD data that obtain high scores after the warm-up phase seems to grow and shrink in size at different time steps, indicating that the model can recover from misclassifying these data.

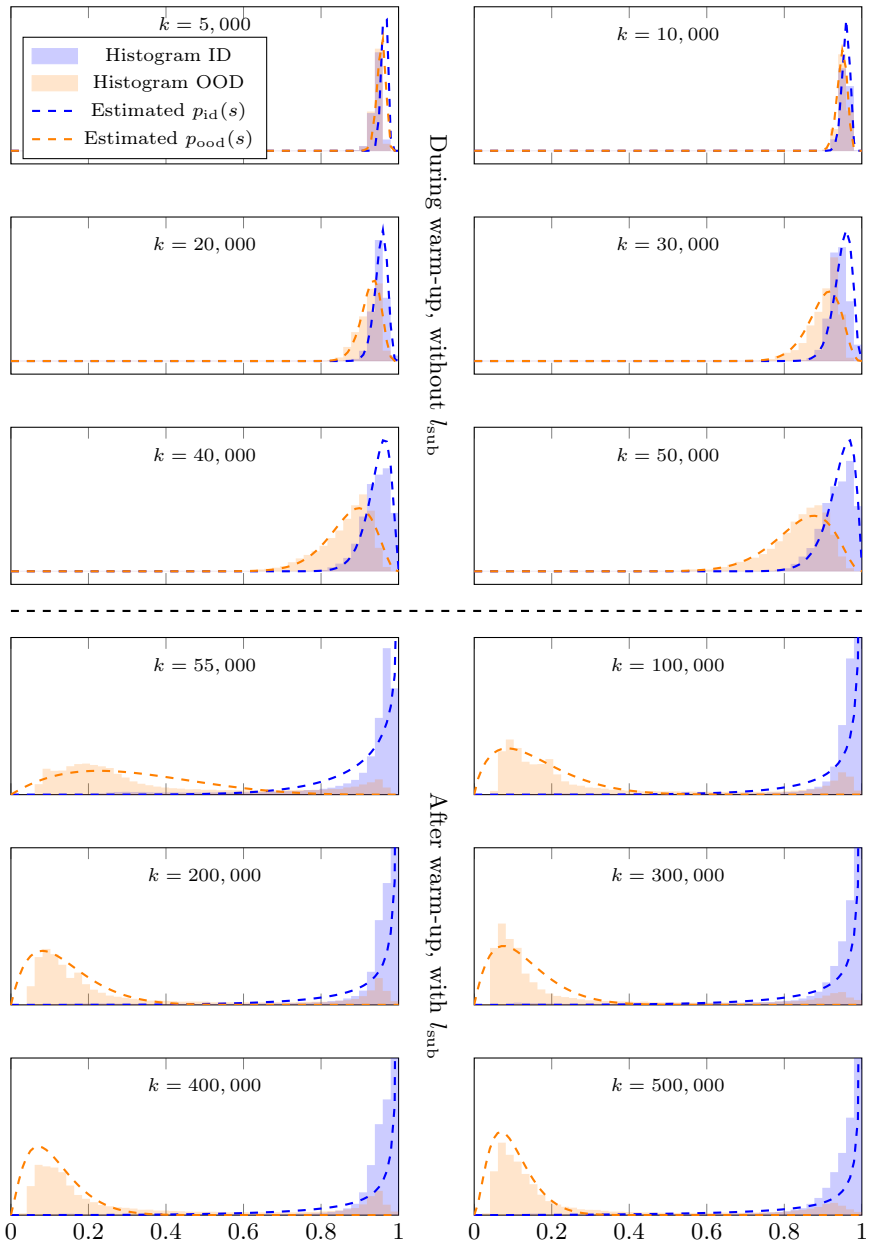

Fig. 10: Distributions of scores and their corresponding estimates at different time steps during training.

14 Indexing of Classes in TIN and IN100

For completeness, we specify how we divide the classes of Tiny ImageNet and ImageNet100 into ID and OOD. How classes are indexed in ImageNet100 are shown in Tab. 9. Here, we use indices 0-49 as ID and 50-99 classes as OOD.

The indexing of classes in Tiny ImageNet is shown in Tab. 10. For experiments on TIN100/100, we use indices 0-99 as ID and 100-199 as OOD. For the experiments conducted using unseen OOD in Sec. 7, we use 0-69 ID, 70-139 as seen OOD, and 140-199 as unseen OOD.

Table 9: Class indexing for ImageNet100.

Class	Index	Class	Index
n01440764	0	n01773797	50
n01443537	1	n01774384	51
n01484850	2	n01774750	52
n01491361	3	n01775062	53
n01494475	4	n01776313	54
n01496331	5	n01795545	55
n01498041	6	n01796340	56
n01514668	7	n01798484	57
n01514859	8	n01806143	58
n01531178	9	n01818515	59
n01537544	10	n01819313	60
n01560419	11	n01820546	61
n01582220	12	n01824575	62
n01592084	13	n01828970	63
n01601694	14	n01829413	64
n01608432	15	n01833805	65
n01614925	16	n01843383	66
n01622779	17	n01847000	67
n01630670	18	n01855672	68
n01632458	19	n01860187	69
n01632777	20	n01877812	70
n01644900	21	n01883070	71
n01664065	22	n01910747	72
n01665541	23	n01914609	73
n01667114	24	n01924916	74
n01667778	25	n01930112	75
n01675722	26	n01943899	76
n01677366	27	n01944390	77
n01685808	28	n01950731	78
n01687978	29	n01955084	79
n01693334	30	n01968897	80
n01695060	31	n01978287	81
n01698640	32	n01978455	82
n01728572	33	n01984695	83
n01729322	34	n01985128	84
n01729977	35	n01986214	85
n01734418	36	n02002556	86
n01735189	37	n02006656	87
n01739381	38	n02007558	88
n01740131	39	n02011460	89
n01742172	40	n02012849	90
n01749939	41	n02013706	91
n01751748	42	n02018207	92
n01753488	43	n02018795	93
n01755581	44	n02027492	94
n01756291	45	n02028035	95
n01770081	46	n02037110	96
n01770393	47	n02051845	97
n01773157	48	n02058221	98
n01773549	49	n02077923	99

Table 10: Class indexing for Tiny ImageNet.

Class	Index	Class	Index	Class	Index	Class	Index
n02814533	0	n03100240	50	n07615774	100	n01768244	150
n02113799	1	n04149813	51	n03355925	101	n03617480	151
n02883205	2	n01917289	52	n04371430	102	n04487081	152
n04597913	3	n04507155	53	n01945685	103	n07768694	153
n03733131	4	n02892201	54	n03649909	104	n02002724	154
n04179913	5	n03089624	55	n03404251	105	n06596364	155
n02802426	6	n02132136	56	n03891332	106	n03042490	156
n04070727	7	n04254777	57	n07695742	107	n04285008	157
n03706229	8	n02927161	58	n04311004	108	n03544143	158
n02321529	9	n03983396	59	n02823428	109	n03980874	159
n02085620	10	n02123045	60	n07749582	110	n02279972	160
n03970156	11	n02791270	61	n04399382	111	n03770439	161
n02730930	12	n09246464	62	n07875152	112	n04560804	162
n02268443	13	n03447447	63	n09193705	113	n07711569	163
n02099712	14	n04417672	64	n02074367	114	n04356056	164
n04133789	15	n07579787	65	n03937543	115	n02977058	165
n04251144	16	n07583066	66	n02206856	116	n03854065	166
n03026506	17	n02795169	67	n01698640	117	n03179701	167
n04532106	18	n03393912	68	n02788148	118	n02486410	168
n07614500	19	n04023962	69	n02917067	119	n02058221	169
n07747607	20	n04486054	70	n01983481	120	n09428293	170
n01742172	21	n02233338	71	n02504458	121	n04265275	171
n03160309	22	n01855672	72	n02281406	122	n01443537	172
n03992509	23	n02814860	73	n04376876	123	n03814639	173
n01784675	24	n04067472	74	n02056570	124	n02165456	174
n01644900	25	n02410509	75	n03388043	125	n02129165	175
n02808440	26	n02480495	76	n02423022	126	n02509815	176
n01774750	27	n03126707	77	n07720875	127	n02190166	177
n02669723	28	n07753592	78	n02125311	128	n02124075	178
n03838899	29	n03085013	79	n03400231	129	n07920052	179
n01910747	30	n02988304	80	n02226429	130	n03804744	180
n03444034	31	n02099601	81	n04465501	131	n01770393	181
n04118538	32	n04501370	82	n02841315	132	n04562935	182
n03662601	33	n02909870	83	n02843684	133	n03976657	183
n02948072	34	n03014705	84	n09332890	134	n04328186	184
n02231487	35	n04146614	85	n02415577	135	n03599486	185
n02106662	36	n02666196	86	n04596742	136	n02999410	186
n02094433	37	n04074963	87	n04275548	137	n03637318	187
n07873807	38	n01882714	88	n01774384	138	n03584254	188
n01641577	39	n03930313	89	n02793495	139	n02769748	189
n03977966	40	n07734744	90	n02395406	140	n02123394	190
n04259630	41	n04366367	91	n07715103	141	n04540053	191
n07871810	42	n03837869	92	n03255030	142	n03763968	192
n02906734	43	n03250847	93	n02403003	143	n03902125	193
n02364673	44	n02236044	94	n04456115	144	n03670208	194
n04008634	45	n03201208	95	n04398044	145	n03796401	195
n09256479	46	n02437312	96	n12267677	146	n01629819	196
n02815834	47	n02837789	97	n03424325	147	n02950826	197
n02481823	48	n02699494	98	n01950731	148	n04532670	198
n02963159	49	n04099969	99	n01984695	149	n01944390	199