

S2C2 - AN ORTHOGONAL METHOD FOR SEMI-SUPERVISED LEARNING ON AMBIGUOUS LABELS

Anonymous authors

Paper under double-blind review

1 SUPPLEMENTARY MATERIAL

1.1 FURTHER INSIGHTS INTO HYPERPARAMETERS

During method development, we looked at a larger variety of hyperparameters for Mean-Teacher (5) and the Plankton dataset (3). We found in general that the model was quite robust to changes of individual parameters. The most impact we noticed from design decision where and where not a gradient is propagated. For example, if we would propagate the error along the pseudo labels for the ambiguity loss calculation the system degenerates almost always. Aside from these design decisions, the weight for $\lambda_{CE^{-1}}$ had the most impact. While we use the same value for the labeled and the unlabeled data, we see slight evidence that a lower value could be beneficial on the labeled data. Moreover, slightly lower or higher weights for the other hyperparameters showed promising results under certain circumstances. As stated in the paper, we aimed at providing general parameters across different datasets and methods and thus did not fine-tune the parameters to a specific combination.

1.2 PSEUDOCODE

In 1, we give the main parts of our proposed method as pseudocode. The code is similar to python and Tensorflow code. In the following, we will describe the used parameters and methods. `tf` is an abbreviation for tensorflow and refers to that function. `prob_ambiguous` is the output $p_a(x)$ for a complete batch. `logits_x_over`, `logits_u_over` and `logits_u2_over` are the overclustering outputs $p_o(x)$ for a complete batch for the labeled data, the unlabeled data and the possible additional second unlabeled input respectively. The labels for the labeled data are given in `l`. `logits_u` is the output $p_n(x)$ for a complete batch of unlabeled data. The parameters `prior_ambiguity`, `wou`, `wa` and `ws` correspond to the hyperparameters p_A , $\lambda_{CE^{-1}}$, λ_a and λ_s in the paper respectively. The parameter `wol` is the weight $\lambda_{CE^{-1}}$ on the labeled data. The parameter `loss_tensor` is SSL loss as tensor (L_{SSL}). The function `threshold()` thresholds the elements of the given vector (first argument) based on the given threshold (second argument). The functions `inverse_ce()`, `ce()` and `be()` calculate the loss value for inverse cross-entropy (CE^{-1}), cross-entropy and binary cross-entropy respectively. The function `get_different_logits()` selects from the given batch logits (first argument) one logit for each image in the batch. A logit is randomly selected of all logits in the batch which to do not share the same label (`l`) or the same pseudo label based on `logits_u`. The function `get_pseudo_ambiguity_labels()` gets pseudo labels for every ambiguity prediction $p_a(x)$ based on the given prior ambiguity estimate p_A as described in the main paper.

The different outputs can easily be realized in a model by extending the dense output layer to the sum of the number of classes and the number of output clusters. Before calculating the loss or applying softmax activation the output can be separated in the desired input to our method.

1.3 EXTENDED RESULTS

In this subsection, we give the complete result tables used for the tables in the main paper. The definitions of the metrics are given in the main paper. The tables are Table 1, Table 2, Table 3, Table 4 and Table 5. Furthermore, we give in ?? an extended and larger version of the qualitative results in the main paper.

```

def calculate_loss(prob_ambiguous, l,
logits_x_over, logits_u, logits_u_over, logits_u2_over,
prior_ambiguity, wou, wol, wa, ws,
loss_tensor):

    # stop gradient on ambiguity scale
    ambiguous_scale = tf.stop_gradient(prob_ambiguous)
    certain_scale = 1 - ambiguous_scale

    pseudo_labels = tf.stop_gradient(tf.nn.softmax(logits_u))
    args_pseudo = tf.argmax(pseudo_labels, axis=1)
    pseudo_mask = threshold(pseudo_labels, 0.95)

    # get different image based on label or pseudo-label
    # for elements in the batch from the batch
    logits_x_over_inverse = get_different_logits(logits_x_over, l)
    logits_u_over_inverse = get_different_logits(logits_u_over, l)

    loss_xeil = inverse_ce(tf.nn.softmax(logits_x_over),
        tf.nn.softmax(logits_x_over_inverse))
    loss_xeil = tf.reduce_mean(loss_xeil)

    loss_xeiu = inverse_ce(tf.nn.softmax(logits_u_over),
        tf.nn.softmax(logits_u_over_inverse))
    loss_xeiu = tf.reduce_mean(loss_xeiu * pseudo_mask * certain_scale)

    # use pseudo labels based on the number of ambiguous elements
    # in each batch to calculate ambiguity loss
    pseudo_ambiguity_label =
        get_pseudo_ambiguity_labels(prob_ambiguous, prior_ambiguity)
    loss_ambiguity = be(tf.stop_gradient(pseudo_ambiguity_label),
        prob_ambiguous)
    loss_ambiguity = tf.reduce_mean(loss_ambiguity)

    # calculate similarity loss
    # use ce to add entropy based on the logits u over
    sim_loss = ce(tf.nn.softmax(logits_u_over),
        tf.nn.softmax(logits_u2_over))
    sim_loss = tf.reduce_mean(sim_loss * ambiguous_scale)

    loss = loss_tensor * certain_scale + wou * loss_xeiu + wol * loss_xeil
        + wa * loss_ambiguity + ws * loss_ambiguous_similarity

    return loss

```

Pseudocode 1: Main pseudocode for the proposed method. The expected parameters and functions are explained in the corresponding subsection.

Table 1: Complete ablation results for Cross-Entropy – The vanilla algorithm is highlighted in light grey. The row below that extend the algorithm with CE^{-1} (3), Clustering & Classification (CC) or both (S2C2). Better results in comparison to the vanilla algorithm are marked bold.

	F1				Ambiguous				# Runs
	best	mean \pm std	best	mean \pm std	best	mean \pm std	best	mean \pm std	
CIFAR10-H									
Baseline	0.6771	0.6704 \pm 0.0062	0.5580	0.5627 \pm 0.0047	-0.1191	-0.1077 \pm 0.0099	-	-	3
+ CE ⁻¹	0.7383	0.7329 \pm 0.0078	0.4692	0.4712 \pm 0.0044	-0.2691	-0.2618 \pm 0.0120	-	-	3
+ CC ($p_A = 0.6$)	0.8570	0.8518 \pm 0.0049	0.8666	0.8662 \pm 0.0005	0.0096	0.0144 \pm 0.0049	0.6240	0.6197 \pm 0.0045	3
+ S2C2 ($p_A = 0.32$)	0.6656	0.6656 \pm 0.0055	0.2155	0.2498 \pm 0.0399	-0.4501	-0.4158 \pm 0.0364	0.2910	0.2960 \pm 0.0044	3
+ S2C2 ($p_A = 0.6$)	0.7827	0.6474 \pm 0.1178	0.5452	0.5096 \pm 0.0382	-0.2375	-0.1378 \pm 0.0870	0.6240	0.5775 \pm 0.0404	3
Plantkon									
Baseline	0.8671	0.8632 \pm 0.0034	0.3045	0.3057 \pm 0.0044	-0.5626	-0.5574 \pm 0.0062	-	-	3
+ CE ⁻¹	0.8896	0.8880 \pm 0.0023	0.2540	0.2602 \pm 0.0087	-0.6356	-0.6278 \pm 0.0110	-	-	2
+ CC ($p_A = 0.6$)	0.9596	0.9221 \pm 0.0337	0.8321	0.8419 \pm 0.0085	-0.1274	-0.0802 \pm 0.0422	0.5908	0.5949 \pm 0.0036	3
+ S2C2 ($p_A = 0.44$)	0.8625	0.9148 \pm 0.0461	0.2192	0.3090 \pm 0.0810	-0.6433	-0.6058 \pm 0.0354	0.4365	0.4511 \pm 0.0127	3
+ S2C2 ($p_A = 0.6$)	0.7824	0.8942 \pm 0.0974	0.2341	0.3580 \pm 0.1074	-0.5484	-0.5361 \pm 0.0155	0.5615	0.5873 \pm 0.0241	3
Turkey									
Baseline	0.8384	0.8307 \pm 0.0071	0.4298	0.4357 \pm 0.0058	-0.4086	-0.3951 \pm 0.0117	-	-	3
+ CE ⁻¹	0.7998	0.7998 \pm 0.0000	0.3338	0.3338 \pm 0.0000	-0.4660	-0.4660 \pm 0.0000	-	-	3
+ CC ($p_A = 0.6$)	0.8452	0.8033 \pm 0.0550	0.3565	0.3213 \pm 0.0542	-0.4887	-0.4820 \pm 0.0068	0.5156	0.5409 \pm 0.0295	3
+ S2C2 ($p_A = 0.22$)	0.7998	0.7998 \pm nan	0.2705	0.2705 \pm nan	-0.5293	-0.5293 \pm nan	0.1087	0.1087 \pm nan	1
+ S2C2 ($p_A = 0.6$)	0.8579	0.8195 \pm 0.0624	0.2764	0.2653 \pm 0.0627	-0.5814	-0.5543 \pm 0.0252	0.5694	0.5841 \pm 0.0776	3
Mice Bone									
Baseline	0.6955	0.6753 \pm 0.0198	0.5475	0.5667 \pm 0.0166	-0.1479	-0.1086 \pm 0.0353	-	-	3
+ S2C2 ($p_A = 0.6$)	0.9388	0.7373 \pm 0.3046	0.3658	0.3018 \pm 0.1005	-0.5730	-0.4355 \pm 0.2041	0.5680	0.5365 \pm 0.0448	3
STL-10									
Baseline	0.8048	0.7918 \pm 0.0125	-	-	-0.8048	-0.7918 \pm 0.0125	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8845	0.8671 \pm 0.0166	-	-	-0.8845	-0.8671 \pm 0.0166	0.5919	0.4727 \pm 0.2643	3

Table 2: Complete ablation results for Mean-Teacher (5) – The vanilla algorithm is highlighted in light grey. The row below that extend the algorithm with CE^{-1} (3), Clustering & Classification (CC) or both (S2C2). Better results in comparison to the vanilla algorithm are marked bold.

	F1				Ambiguous				# Runs
	best	mean \pm std	best	mean \pm std	best	mean \pm std	best	mean \pm std	
CIFAR10-H									
Baseline	0.7353	0.7280 \pm 0.0065	0.4693	0.4807 \pm 0.0106	-0.2659	-0.2473 \pm 0.0171	-	-	3
+ CE-1	0.7360	0.7297 \pm 0.0054	0.4747	0.4753 \pm 0.0011	-0.2613	-0.2544 \pm 0.0061	-	-	3
+ CC ($p_A = 0.6$)	0.8565	0.7791 \pm 0.1243	0.8657	0.8747 \pm 0.0153	0.0092	0.0956 \pm 0.1396	0.6145	0.5962 \pm 0.0375	3
+ S2C2 ($p_A = 0.32$)	0.6614	0.7243 \pm 0.0554	0.3197	0.4615 \pm 0.1272	-0.3417	-0.2628 \pm 0.0805	0.2910	0.3070 \pm 0.0151	3
+ S2C2 ($p_A = 0.6$)	0.8513	0.7066 \pm 0.1260	0.5244	0.4328 \pm 0.0839	-0.3269	-0.2738 \pm 0.0610	0.6145	0.5757 \pm 0.0336	3
Plankton									
Baseline	0.8872	0.8828 \pm 0.0044	0.2584	0.2620 \pm 0.0037	-0.6287	-0.6208 \pm 0.0080	-	-	3
+ CE-1	0.8846	0.8821 \pm 0.0035	0.2568	0.2623 \pm 0.0050	-0.6278	-0.6198 \pm 0.0082	-	-	3
+ CC ($p_A = 0.6$)	0.9645	0.9345 \pm 0.0260	0.8303	0.8377 \pm 0.0065	-0.1342	-0.0968 \pm 0.0324	0.5928	0.5898 \pm 0.0029	3
+ S2C2 ($p_A = 0.44$)	0.8690	0.8634 \pm 0.0080	0.2753	0.2966 \pm 0.0301	-0.5937	-0.5667 \pm 0.0381	0.4064	0.4098 \pm 0.0049	2
+ S2C2 ($p_A = 0.6$)	0.9130	0.9056 \pm 0.0087	0.2484	0.2699 \pm 0.0267	-0.6646	-0.6357 \pm 0.0282	0.6164	0.6124 \pm 0.0143	3
Turkey									
Baseline	0.8182	0.8158 \pm 0.0049	0.4512	0.4579 \pm 0.0078	-0.3670	-0.3579 \pm 0.0079	-	-	3
+ CE-1	0.7998	0.7998 \pm 0.0000	0.3338	0.3338 \pm 0.0000	-0.4660	-0.4660 \pm 0.0000	-	-	3
+ CC ($p_A = 0.6$)	0.8527	0.8829 \pm 0.0295	0.3400	0.3816 \pm 0.0384	-0.5127	-0.5013 \pm 0.0098	0.5837	0.5428 \pm 0.0378	3
+ S2C2 ($p_A = 0.22$)	0.7998	0.7998 \pm nan	0.1719	0.1719 \pm nan	-0.6278	-0.6278 \pm nan	0.5252	0.4748 \pm nan	1
+ S2C2 ($p_A = 0.6$)	0.8645	0.8639 \pm 0.0008	0.3392	0.3439 \pm 0.0067	-0.5253	-0.5200 \pm 0.0075	0.7691	0.7970 \pm 0.0394	2
Mice Bone									
Baseline	0.6641	0.6688 \pm 0.0217	0.4883	0.5209 \pm 0.0284	-0.1758	-0.1479 \pm 0.0301	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8984	0.8940 \pm 0.0124	0.3511	0.4300 \pm 0.0887	-0.5473	-0.4641 \pm 0.0848	0.5266	0.5444 \pm 0.0178	3
STL-10									
Baseline	0.8067	0.7863 \pm 0.0173	-	-	-0.8067	-0.7863 \pm 0.0173	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8928	0.8751 \pm 0.0188	-	-	-0.8928	-0.8751 \pm 0.0188	0.5897	0.4732 \pm 0.2646	3

Table 3: Complete ablation results for Pi-Model (1) – The vanilla algorithm is highlighted in light grey. The row below that extend the algorithm with CE^{-1} (3), Clustering & Classification (CC) or both (S2C2). Better results in comparison to the vanilla algorithm are marked bold.

	F1				Ambiguous				# Runs
	best	mean \pm std	best	mean \pm std	best	mean \pm std	best	mean \pm std	
CIFAR10-H									
Baseline	0.7153	0.7153 \pm 0.0005	0.4913	0.5008 \pm 0.0085	-0.2240	-0.2145 \pm 0.0087	-	-	3
+ CE ⁻¹	0.7255	0.7163 \pm 0.0109	0.4917	0.4988 \pm 0.0138	-0.2337	-0.2175 \pm 0.0244	-	-	3
+ CC ($p_A = 0.6$)	0.8420	0.6991 \pm 0.1238	0.8670	0.8823 \pm 0.0133	0.0250	0.1832 \pm 0.1371	0.6120	0.5728 \pm 0.0349	3
+ S2C2 ($p_A = 0.32$)	0.7291	0.7038 \pm 0.0506	0.3528	0.3564 \pm 0.0849	-0.3764	-0.3474 \pm 0.0466	0.3230	0.3193 \pm 0.0095	3
+ S2C2 ($p_A = 0.6$)	0.8305	0.8352 \pm 0.0143	0.4340	0.4680 \pm 0.0311	-0.3965	-0.3672 \pm 0.0258	0.6125	0.6122 \pm 0.0035	3
Plankton									
Baseline	0.8757	0.8747 \pm 0.0024	0.2843	0.2840 \pm 0.0019	-0.5914	-0.5907 \pm 0.0007	-	-	3
+ CE ⁻¹	0.8826	0.8783 \pm 0.0043	0.2700	0.2745 \pm 0.0084	-0.6126	-0.6038 \pm 0.0122	-	-	3
+ CC ($p_A = 0.6$)	0.8027	0.8899 \pm 0.0773	0.4346	0.7044 \pm 0.2336	-0.3681	-0.1855 \pm 0.1594	0.5708	0.5869 \pm 0.0159	3
+ CC ($p_A = 0.6$)	0.9260	0.9260 \pm 0.0028	0.3411	0.3633 \pm 0.0216	-0.5850	-0.5627 \pm 0.0231	0.4597	0.4585 \pm 0.0064	3
+ S2C2 ($p_A = 0.6$)	0.7979	0.8561 \pm 0.0910	0.1908	0.2613 \pm 0.0960	-0.6071	-0.5948 \pm 0.0108	0.5782	0.5829 \pm 0.0042	3
Turkey									
Baseline	0.8211	0.8172 \pm 0.0038	0.3946	0.4253 \pm 0.0398	-0.4265	-0.3919 \pm 0.0410	-	-	3
+ CE ⁻¹	0.7998	0.7998 \pm 0.0000	0.3338	0.3338 \pm 0.0000	-0.4660	-0.4660 \pm 0.0000	-	-	3
+ CC ($p_A = 0.6$)	0.7828	0.7987 \pm 0.0166	0.3071	0.3266 \pm 0.0192	-0.4758	-0.4722 \pm 0.0031	0.6025	0.5800 \pm 0.0289	3
+ S2C2 ($p_A = 0.22$)	-	-	-	-	-	-	-	-	0
+ S2C2 ($p_A = 0.6$)	0.8743	0.8768 \pm 0.0035	0.2333	0.3071 \pm 0.1044	-0.6410	-0.5697 \pm 0.1009	0.8093	0.9047 \pm 0.1348	2
Mice Bone									
Baseline	0.6815	0.6673 \pm 0.0123	0.5411	0.5510 \pm 0.0150	-0.1403	-0.1164 \pm 0.0237	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8801	0.8575 \pm 0.0228	0.3099	0.4348 \pm 0.1424	-0.5702	-0.4227 \pm 0.1649	0.5621	0.5266 \pm 0.0307	3
STL-10									
Baseline	0.8256	0.8013 \pm 0.0169	-	-	-0.8256	-0.8013 \pm 0.0169	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8954	0.7951 \pm 0.0856	-	-	-0.8954	-0.7951 \pm 0.0856	0.5823	0.3425 \pm 0.3128	3

Table 4: Complete ablation results for Pseudo-Label (2) – The vanilla algorithm is highlighted in light grey. The row below that extend the algorithm with CE^{-1} (3), Clustering & Classification (CC) or both (S2C2). Better results in comparison to the vanilla algorithm are marked bold.

	F1		best	mean \pm std	best	mean \pm std	Ambiguous		# Runs
	best	mean \pm std					best	mean \pm std	
CIFAR10-H									
Baseline	0.6970	0.6914 \pm 0.0057	0.5330	0.5359 \pm 0.0050	-0.1640	-0.1554 \pm 0.0103	-	-	3
+ CE-1	0.7054	0.6977 \pm 0.0108	0.5194	0.5264 \pm 0.0113	-0.1860	-0.1713 \pm 0.0221	-	-	3
+ CC ($p_A = 0.6$)	0.8265	0.6583 \pm 0.1457	0.8670	0.8838 \pm 0.0147	0.0404	0.2255 \pm 0.1603	0.6190	0.5803 \pm 0.0337	3
+ S2C2 ($p_A = 0.32$)	0.6236	0.6941 \pm 0.0611	0.2382	0.4058 \pm 0.1464	-0.3854	-0.2883 \pm 0.0870	0.3245	0.3237 \pm 0.0063	3
+ S2C2 ($p_A = 0.6$)	0.8374	0.7448 \pm 0.1440	0.5132	0.4854 \pm 0.0967	-0.3242	-0.2594 \pm 0.0618	0.6100	0.5957 \pm 0.0311	3
Plankton									
Baseline	0.8762	0.8730 \pm 0.0045	0.2742	0.2821 \pm 0.0098	-0.6020	-0.5908 \pm 0.0143	-	-	3
+ CE-1	0.8737	0.8727 \pm 0.0015	0.2788	0.2794 \pm 0.0009	-0.5949	-0.5932 \pm 0.0024	-	-	2
+ CC ($p_A = 0.6$)	0.8919	0.9046 \pm 0.0220	0.4085	0.6967 \pm 0.2497	-0.4833	-0.2078 \pm 0.2400	0.6242	0.5991 \pm 0.0221	3
+ S2C2 ($p_A = 0.44$)	0.8640	0.9016 \pm 0.0327	0.2661	0.3285 \pm 0.0545	-0.5979	-0.5731 \pm 0.0217	0.4304	0.4491 \pm 0.0167	3
+ S2C2 ($p_A = 0.6$)	0.8931	0.8539 \pm 0.0555	0.3176	0.2844 \pm 0.0469	-0.5755	-0.5695 \pm 0.0085	0.5945	0.5843 \pm 0.0144	2
Turkey									
Baseline	0.8237	0.8245 \pm 0.0012	0.4488	0.4527 \pm 0.0056	-0.3749	-0.3718 \pm 0.0044	-	-	2
+ CE-1	0.7998	0.7998 \pm 0.0000	0.3338	0.3338 \pm 0.0000	-0.4660	-0.4660 \pm 0.0000	-	-	3
+ CC ($p_A = 0.6$)	0.8486	0.8207 \pm 0.0334	0.3708	0.3445 \pm 0.0319	-0.4778	-0.4762 \pm 0.0016	0.6310	0.5947 \pm 0.0601	3
+ S2C2 ($p_A = 0.22$)	0.7998	0.7998 \pm 0.0000	0.1675	0.2291 \pm 0.0871	-0.6322	-0.5706 \pm 0.0871	0.5000	0.4783 \pm 0.0307	2
+ S2C2 ($p_A = 0.6$)	0.8344	0.8292 \pm 0.0074	0.3504	0.3883 \pm 0.0536	-0.4841	-0.4409 \pm 0.0610	0.8560	0.5305 \pm 0.4604	2
Mice Bone									
Baseline	0.6660	0.6524 \pm 0.0124	0.5703	0.5768 \pm 0.0057	-0.0957	-0.0756 \pm 0.0176	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8658	0.7275 \pm 0.2267	0.3752	0.3465 \pm 0.0978	-0.4906	-0.3810 \pm 0.1363	0.5444	0.5444 \pm 0.0118	3
STL-10									
Baseline	0.8248	0.8016 \pm 0.0184	-	-	-0.8248	-0.8016 \pm 0.0184	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8887	0.7903 \pm 0.0854	-	-	-0.8887	-0.7903 \pm 0.0854	0.5921	0.4606 \pm 0.2581	3

Table 5: Complete ablation results for FixMatch (4) – The vanilla algorithm is highlighted in light grey. The row below that extend the algorithm with CE^{-1} (3), Clustering & Classification (CC) or both (S2C2). Better results in comparison to the vanilla algorithm are marked bold.

	F1				Ambiguous				# Runs
	best	mean \pm std	best	mean \pm std	best	mean \pm std	best	mean \pm std	
CIFAR10-H									
Baseline	0.7809	0.7713 \pm 0.0097	0.4199	0.4332 \pm 0.0122	-0.3611	-0.3381 \pm 0.0218	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8309	0.7947 \pm 0.0335	0.4949	0.4746 \pm 0.0190	-0.3360	-0.3200 \pm 0.0145	0.5805	0.5688 \pm 0.0169	3
Plankton									
Baseline	0.8581	0.8324 \pm 0.0278	0.3029	0.3237 \pm 0.0231	-0.5552	-0.5088 \pm 0.0509	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8720	0.8666 \pm 0.0649	0.3128	0.3228 \pm 0.0659	-0.5592	-0.5438 \pm 0.0134	0.5770	0.5782 \pm 0.0259	3
Turkey									
Baseline	0.8214	0.8196 \pm 0.0019	0.4333	0.4455 \pm 0.0121	-0.3881	-0.3741 \pm 0.0130	-	-	3
+ S2C2 ($p_A = 0.6$)	0.8356	0.8401 \pm 0.0146	0.2817	0.3499 \pm 0.0673	-0.5539	-0.4902 \pm 0.0581	0.2691	0.4827 \pm 0.1856	3
STL-10									
Baseline	0.8957	0.8948 \pm 0.0011	-	-	-0.8957	-0.8948 \pm 0.0011	-	-	3
+ S2C2 ($p_A = 0.6$)	0.9145	0.8690 \pm 0.0440	-	-	-0.9145	-0.8690 \pm 0.0440	0.5746	0.5586 \pm 0.0266	3

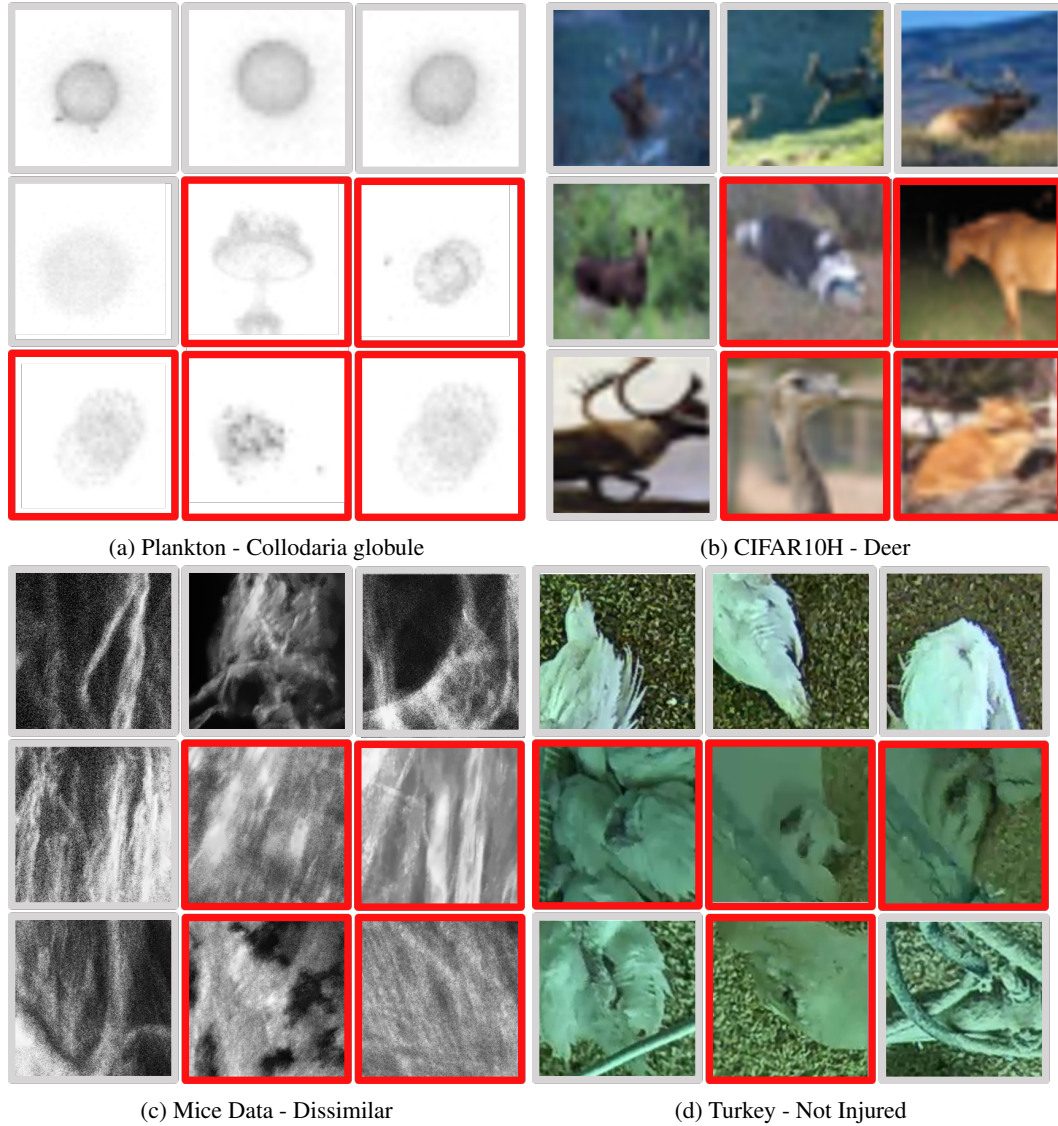


Figure 1: Qualitative Results for selected classes across different confidences and ambiguity predictions – In each block, the top row is predicted as certain and confident about the class. The middle row is predicted as ambiguous and confident about the class. The last row shows non-confident predictions and uncertain ambiguity predictions. Wrong classifications based on the normal head are highlighted in red.

REFERENCES

- [1] Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. In: International Conference on Learning Representations (2017)
- [2] Lee, D.H.: Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on challenges in representation learning, ICML. vol. 3, p. 2 (2013)
- [3] Schmarje, L., Brünger, J., Santarossa, M., Schröder, S.M., Kiko, R., Koch, R.: Fuzzy Overclustering: Semi-supervised classification of fuzzy labels with overclustering and inverse cross-entropy. *Sensors* (2021)
- [4] Sohn, K., Berthelot, D., Li, C.L., Zhang, Z., Carlini, N., Cubuk, E.D., Kurakin, A., Zhang, H., Raffel, C.: FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence. *Advances in Neural Information Processing Systems 33 pre-proceedings (NeurIPS 2020)* (2020)
- [5] Tarvainen, A., Valpola, H.: Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In: *ICLR* (2017)