In [1]:
```
%matplotlib notebook
import os
import pandas as pd
import numpy as np
from numpy.random import randint
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from matplotlib import pyplot as plt
import matplotlib.dates as mdates
import time
from PIL import Image

import torch
from torch import nn
from torch.nn import functional as F
import torch.utils.data as td
import torchvision as tv
from sklearn.metrics import confusion_matrix
```

In [2]:
```
import cv2
import random
```

In [3]:
```
device = 'cuda' if torch.cuda.is_available() else 'cpu'
print(device)
print(torch.__version__)
```

```
cuda
1.0.1.post2
```

In [5]:
```
from glob import glob
from PIL import Image
```

In [6]:
```
data_dir = 'train.csv'
```

In [7]:
```
train_data_label = pd.read_csv(data_dir)
```

In [8]:
```
train_data_label
```

Out[8]:

	Image	Id
0	00022e1a.jpg	w_e15442c
1	000466c4.jpg	w_1287fbc
2	00087b01.jpg	w_da2efe0
3	001296d5.jpg	w_19e5482
4	0014cfdf.jpg	w_f22f3e3
5	0025e8c2.jpg	w_8b1ca89
6	0026a8ab.jpg	w_eaad6a8
7	0031c258.jpg	new_whale
8	0035632e.jpg	w_3d0bc7a
9	0037e7d3.jpg	w_50db782
10	00389cd7.jpg	w_2863d51
11	0042dcc4.jpg	w_6dc7db6
12	0042ea34.jpg	w_968f2ca
13	00467ae9.jpg	w_fd1cb9d
14	004a97f3.jpg	w_60759c2
15	004c5fb9.jpg	w_ab6bb0a
16	005c57e7.jpg	w_79b42cd
17	006d0aaf.jpg	w_c9ba30c
18	0078af23.jpg	w_e6ec8ee
19	007c3603.jpg	new_whale
20	00863b8c.jpg	new_whale
21	008809b5.jpg	w_7e5cc5e
22	008baccf.jpg	w_ab6db0f
23	0091c92b.jpg	w_bc8d634
24	009dca38.jpg	w_b59c523
25	00a29f63.jpg	w_2850471
26	00aa021c.jpg	new_whale
27	00ab018e.jpg	w_51969d2
28	00ac0e86.jpg	w_4be8a3e
29	00acb5a9.jpg	w_c0cfd5b
...	...	...
9820	ff2d0d82.jpg	new_whale

	Image	Id
9821	ff3509c0.jpg	w_8ba2066
9822	ff38054f.jpg	w_6734e40
9823	ff3c2c25.jpg	w_f5eb6c6
9824	ff3f151f.jpg	w_9d5f5cc
9825	ff421460.jpg	w_a16b600
9826	ff4bb3a4.jpg	w_2d99a0c
9827	ff5a5ed4.jpg	w_f3bd33a
9828	ff6946b4.jpg	w_17ee910
9829	ff6c1a92.jpg	w_4f0676a
9830	ff6d6894.jpg	w_b426ff3
9831	ff7247f6.jpg	w_41afa58
9832	ff8b2ad0.jpg	w_0f20cbc
9833	ff92447f.jpg	w_372ae75
9834	ff9d60a3.jpg	w_eb0a6ed
9835	ffa6b9ac.jpg	w_edf5f77
9836	ffa78ccc.jpg	w_89d9c03
9837	ffb71ac2.jpg	new_whale
9838	ffbaa734.jpg	w_a190b3b
9839	ffbca206.jpg	w_73b26b7
9840	ffc0b437.jpg	w_ace8c54
9841	ffc6986f.jpg	w_fce6ab2
9842	ffcfd124.jpg	w_41a260a
9843	ffd01d82.jpg	w_6249155
9844	ffd1e7aa.jpg	new_whale
9845	ffe5c306.jpg	w_2ceab05
9846	ffeaa7a4.jpg	w_b067417
9847	ffecec63.jpg	w_8b56cb1
9848	fff04277.jpg	w_2dcbf82
9849	fffd4260.jpg	w_b9bfd4e

9850 rows × 2 columns

In [9]:
```
data_train, data_val = train_test_split(train_data_label,test_size=0.2,
random_state=42)
```

In [10]:
```
data_train
```

Out[10]:

	Image	Id
4275	70238365.jpg	w_ebf3f26
4533	75e189d4.jpg	w_715c557
1745	2e08f2ba.jpg	w_b48535f
5427	8cea266f.jpg	w_8e92baa
1452	26482485.jpg	w_b3655a6
101	0267139c.jpg	w_e156c87
319	07a58418.jpg	w_4e7fc3e
9218	f0411154.jpg	w_326e389
2742	47575ccc.jpg	w_afe953f
625	0fc63e94.jpg	w_9771603
1922	32754498.jpg	new_whale
9770	fdd5b843.jpg	new_whale
7290	bcf56bfe.jpg	new_whale
3133	515c76d0.jpg	new_whale
1254	2091af92.jpg	w_37dd956
1195	1f1122d6.jpg	w_6132293
2436	3f514e6e.jpg	w_6a16373
4217	6eb7c487.jpg	w_ff7630a
2609	442005d4.jpg	w_1e7bb93
8895	e7b13d2a.jpg	new_whale
259	06252a55.jpg	w_b688397
4429	734430c9.jpg	w_e7f8e67
334	07e96de0.jpg	w_7307089
3071	502c2aeb.jpg	w_90f5d3b
3946	677c2196.jpg	w_8114b1b
8809	e5457edf.jpg	w_cd65880
5654	928ef6f7.jpg	w_b0362e2
7514	c29996d3.jpg	w_ea2385d
8889	e79c6fa7.jpg	w_4a17405
461	0b635230.jpg	new_whale
...	...	...
2734	471cc7c1.jpg	w_fe49bc4

	Image	Id
189	04b3714e.jpg	w_37dd956
9167	ef4108d9.jpg	w_0d39a68
2747	476f249a.jpg	w_3197568
2047	3535398b.jpg	w_6e8486d
7849	cb93a0b0.jpg	w_9ea2cc3
2558	42d640db.jpg	w_fac9864
9274	f1b24b92.jpg	w_fe95ab8
8666	e16eed44.jpg	w_1e68ef5
6396	a651c7c4.jpg	w_2c55303
3385	57efc103.jpg	w_addcafa
4555	768a1cf7.jpg	w_db0ad01
1184	1ebd8ea1.jpg	w_8867074
6420	a6dc7463.jpg	w_fba3bde
5051	8389019f.jpg	w_987a36f
5311	8a3fb9df.jpg	w_bc9dc37
2433	3f419c58.jpg	w_1306632
6949	b4442482.jpg	w_c00534d
769	13359607.jpg	w_7e8b270
1685	2c8f7197.jpg	w_6b4af70
8322	d8b470f4.jpg	w_26f9f95
5578	90ab92e3.jpg	new_whale
4426	73393f00.jpg	w_0981144
466	0b7dbf66.jpg	w_97f5054
6265	a300e9ee.jpg	w_1f6e1db
5734	9490698e.jpg	w_6c899ff
5191	87055451.jpg	w_19c005a
5390	8c37aa0c.jpg	w_a254eb0
860	152fb267.jpg	w_45b90d9
7270	bc7482e2.jpg	w_02facde

7880 rows × 2 columns

In [11]:
```
data_val
```

Out[11]:

	Image	Id
8920	e871b226.jpg	w_38158d6
9839	ffbca206.jpg	w_73b26b7
1851	3054d682.jpg	w_0acce53
6334	a49b519c.jpg	w_7dee51b
8516	dd5bcda3.jpg	w_813c5be
416	0a254aa7.jpg	w_9b401eb
8830	e5d8bc42.jpg	w_f93d780
6369	a584067a.jpg	w_66c1b54
7194	bad83ce8.jpg	w_853c1f7
8444	db9e8d8f.jpg	w_44f0fa2
3032	4f447f94.jpg	new_whale
6170	a0d45211.jpg	w_53064a6
6322	a451963f.jpg	w_3166a4d
6065	9e365d39.jpg	w_fe49bc4
5956	9b1232a8.jpg	w_d26cc27
7964	cebf233a.jpg	new_whale
7664	c6f8ee24.jpg	w_4a81594
106	02a66def.jpg	w_b14007c
7131	b8f8c2c9.jpg	w_931ade2
33	00b588d6.jpg	w_25871da
5944	9ac90cec.jpg	w_a2633d4
8695	e21a39c8.jpg	w_6d274b2
388	0942db2a.jpg	w_f792125
8224	d5c9c1f2.jpg	new_whale
3055	4fc624a7.jpg	w_b6410bc
3100	50ada087.jpg	w_d9055d1
2973	4db19916.jpg	w_d8eae88
2455	3fbb89ef.jpg	w_8d83172
3006	4e91f0de.jpg	w_4875b75
8028	d034eb40.jpg	w_7763134
...	...	...
6340	a4b61383.jpg	w_1addcc2

	Image	Id
8711	e2a325c4.jpg	w_89ca343
5706	93e38b2a.jpg	w_7248590
2288	3b4b1d4a.jpg	w_dfbfe10
1055	1b05bbb0.jpg	w_fe6c1f3
233	05859f6e.jpg	w_5a29f9d
3773	62c8b6cc.jpg	w_6092d5c
6216	a1e5a5e8.jpg	w_7895123
5266	88e9d696.jpg	w_573eb8f
6953	b45e909d.jpg	w_7923fdc
4376	72156898.jpg	w_ae07541
6495	a887bbbe.jpg	w_72dfe51
8992	ea26cb5f.jpg	w_7dc3fae
4374	72046206.jpg	w_4e68ddc
811	142364d6.jpg	w_4b7b80b
5467	8d9b4a76.jpg	w_32a920b
2867	4b16a4ff.jpg	w_f1a4389
8257	d6e6bce6.jpg	new_whale
4775	7bf807c6.jpg	w_83df8d5
2045	352d8e07.jpg	w_a2eb1bb
9807	fec336f2.jpg	w_1ec267f
5697	93c063ac.jpg	new_whale
7743	c8d44ff3.jpg	new_whale
6017	9cf24fb5.jpg	w_5817e08
2517	41c3e932.jpg	w_5102893
6914	b34a73a4.jpg	w_5bc5e63
3257	54c9b3ed.jpg	w_98baff9
111	02c82510.jpg	w_a8b5c0f
2884	4b6cee90.jpg	w_bd1dbed
8774	e4a44402.jpg	w_4ebeafb

1970 rows × 2 columns

In [12]:
```
classes = train_data_label.Id.unique()
```

In [13]:
```
len(classes)
```

Out[13]: 4251

In [14]:
```
classes_Id_map = {}
```

In [15]:
```
for i in range(len(classes)):
    classes_Id_map[i] = classes[i]
```

In [16]:
```
classes_Id_map[3]
```

Out[16]: 'w_19e5482'

In [17]:
```
Id_classes_map = {}
```

In [18]:
```
for i in range(len(classes)):
    Id_classes_map[classes[i]] = i
```

In [19]:
```
series = train_data_label['Id']

label_series = train_data_label['Id'].value_counts()
```

In [20]:
```
Id_classes_map
```

```
Out[20]: {'w_e15442c': 0,
          'w_1287fbc': 1,
          'w_da2efe0': 2,
          'w_19e5482': 3,
          'w_f22f3e3': 4,
          'w_8b1ca89': 5,
          'w_eaad6a8': 6,
          'new_whale': 7,
          'w_3d0bc7a': 8,
          'w_50db782': 9,
          'w_2863d51': 10,
          'w_6dc7db6': 11,
          'w_968f2ca': 12,
          'w_fd1cb9d': 13,
          'w_60759c2': 14,
          'w_ab6bb0a': 15,
          'w_79b42cd': 16,
          'w_c9ba30c': 17,
          'w_e6ec8ee': 18,
          'w_7e5cc5e': 19,
          'w_ab6db0f': 20,
          'w_bc8d634': 21,
          'w_b59c523': 22,
          'w_2850471': 23,
          'w_51969d2': 24,
          'w_4be8a3e': 25,
          'w_c0cfd5b': 26,
          'w_339c8ae': 27,
          'w_7c7a78c': 28,
          'w_25871da': 29,
          'w_97f5054': 30,
          'w_6926f08': 31,
          'w_9c70a11': 32,
          'w_8e4abc9': 33,
          'w_222dcb7': 34,
          'w_36a4e8b': 35,
          'w_886257d': 36,
          'w_a2eb1bb': 37,
          'w_2ac8b4d': 38,
          'w_9ea2cc3': 39,
          'w_af67683': 40,
          'w_7b3f9d1': 41,
          'w_effd0ea': 42,
          'w_96b697f': 43,
          'w_c07f119': 44,
          'w_685466f': 45,
          'w_04c1951': 46,
          'w_307065e': 47,
          'w_b4369cc': 48,
          'w_2173953': 49,
          'w_27cf4e2': 50,
          'w_aed023d': 51,
          'w_af8cad1': 52,
          'w_a837660': 53,
          'w_4e68ddc': 54,
          'w_f19faeb': 55,
          'w_0ead9d7': 56,
```

```
 'w_8c1ec28': 57,
 'w_7ef90f7': 58,
 'w_88e5933': 59,
 'w_f18df18': 60,
 'w_0a71e87': 61,
 'w_f663b5e': 62,
 'w_03f060f': 63,
 'w_f2e9eb1': 64,
 'w_4114553': 65,
 'w_4fdaa2a': 66,
 'w_55d6c4f': 67,
 'w_12c3d3d': 68,
 'w_e74980b': 69,
 'w_778ee6e': 70,
 'w_4b1fad6': 71,
 'w_1c32062': 72,
 'w_67de30b': 73,
 'w_656afeb': 74,
 'w_0de84f0': 75,
 'w_9e559ff': 76,
 'w_392bee3': 77,
 'w_2c6fe6f': 78,
 'w_5384c57': 79,
 'w_e115a81': 80,
 'w_8ba2066': 81,
 'w_0f16be3': 82,
 'w_b729b1f': 83,
 'w_b074cdf': 84,
 'w_73cbacd': 85,
 'w_e05ee50': 86,
 'w_69f4311': 87,
 'w_4fd48e7': 88,
 'w_7276f4b': 89,
 'w_93bf889': 90,
 'w_d142b00': 91,
 'w_e156c87': 92,
 'w_5297ab3': 93,
 'w_73d5489': 94,
 'w_6202983': 95,
 'w_b14007c': 96,
 'w_2a939eb': 97,
 'w_5624f08': 98,
 'w_e59a1f0': 99,
 'w_a8b5c0f': 100,
 'w_a1b985a': 101,
 'w_913a2ad': 102,
 'w_511c464': 103,
 'w_d3c8520': 104,
 'w_6e8486d': 105,
 'w_18df014': 106,
 'w_96b8436': 107,
 'w_681dba6': 108,
 'w_7049d9f': 109,
 'w_cd22b23': 110,
 'w_0b3c02c': 111,
 'w_e8ca955': 112,
 'w_9b401eb': 113,
```

```
'w_fea7fe6': 114,
'w_e07f3d1': 115,
'w_1d2fbc1': 116,
'w_da0b8b5': 117,
'w_ccca0db': 118,
'w_a4f77bd': 119,
'w_711aaa1': 120,
'w_9ae554b': 121,
'w_1274a11': 122,
'w_9dcf002': 123,
'w_d88328d': 124,
'w_125095f': 125,
'w_8b22583': 126,
'w_6b82ccc': 127,
'w_9562910': 128,
'w_1febbf3': 129,
'w_7ad249d': 130,
'w_cf3c233': 131,
'w_eb8429c': 132,
'w_c11932e': 133,
'w_c9e1cdc': 134,
'w_b8f8e69': 135,
'w_e3b2cfa': 136,
'w_71c7322': 137,
'w_3621c49': 138,
'w_bc62c0e': 139,
'w_33909b8': 140,
'w_a31aa30': 141,
'w_717a293': 142,
'w_44f66a7': 143,
'w_86b3de4': 144,
'w_ba1befb': 145,
'w_4fa1bd5': 146,
'w_379ba08': 147,
'w_c708776': 148,
'w_01cbcbf': 149,
'w_962164f': 150,
'w_c926fc3': 151,
'w_5c62a56': 152,
'w_379785d': 153,
'w_8420f42': 154,
'w_2c51d33': 155,
'w_b34793e': 156,
'w_bf14813': 157,
'w_5cb5fc3': 158,
'w_b2f1afa': 159,
'w_5b99089': 160,
'w_2fe43c7': 161,
'w_eec1133': 162,
'w_3dd1a87': 163,
'w_b83b069': 164,
'w_37dd956': 165,
'w_302af0a': 166,
'w_47d2bc6': 167,
'w_f546b0a': 168,
'w_9d5f5cc': 169,
'w_ad8bc47': 170,
```

```
'w_09d7946': 171,
'w_7523d74': 172,
'w_9d48bcf': 173,
'w_1894a1c': 174,
'w_3197568': 175,
'w_a85a3c9': 176,
'w_b6efe77': 177,
'w_1d53d9c': 178,
'w_a71aa99': 179,
'w_c9dd32e': 180,
'w_95874a5': 181,
'w_a18d0dc': 182,
'w_1029f4e': 183,
'w_8464638': 184,
'w_d6e437d': 185,
'w_5d86dc6': 186,
'w_2341d0a': 187,
'w_7f867ec': 188,
'w_da63cba': 189,
'w_f8cd6b6': 190,
'w_db0ad01': 191,
'w_0f54cdf': 192,
'w_a91b997': 193,
'w_af5d11a': 194,
'w_3aa2073': 195,
'w_e4971dc': 196,
'w_fb94a5a': 197,
'w_7c6ad05': 198,
'w_e158680': 199,
'w_5a29f9d': 200,
'w_b318111': 201,
'w_7ab1759': 202,
'w_296749b': 203,
'w_b5144b0': 204,
'w_a2a1378': 205,
'w_41fa033': 206,
'w_b2d937a': 207,
'w_cae7677': 208,
'w_13c6b6b': 209,
'w_b942708': 210,
'w_19ca2c6': 211,
'w_bcb0696': 212,
'w_0324b97': 213,
'w_a8e5d39': 214,
'w_5aa4856': 215,
'w_980bb1a': 216,
'w_dbf4e7f': 217,
'w_7554f44': 218,
'w_b688397': 219,
'w_53859b2': 220,
'w_65e2424': 221,
'w_4d358e5': 222,
'w_3f907d6': 223,
'w_5c07f42': 224,
'w_845f7ca': 225,
'w_a66b2a5': 226,
'w_0471bdf': 227,
```

```
 'w_898eb3b': 228,
 'w_40be51f': 229,
 'w_94e1ed4': 230,
 'w_6c2d7ea': 231,
 'w_b9ee5ec': 232,
 'w_79cde72': 233,
 'w_b96d4e0': 234,
 'w_8f2fb79': 235,
 'w_6398e9d': 236,
 'w_ccfdb00': 237,
 'w_1a70685': 238,
 'w_0e7cb1c': 239,
 'w_af18c29': 240,
 'w_5dc1c2d': 241,
 'w_80171b9': 242,
 'w_1eafe46': 243,
 'w_a14ccaa': 244,
 'w_dee6441': 245,
 'w_f8e6546': 246,
 'w_992c775': 247,
 'w_7f0700f': 248,
 'w_1a229eb': 249,
 'w_046634b': 250,
 'w_1dff010': 251,
 'w_bb79fb8': 252,
 'w_d1bbde9': 253,
 'w_ff7630a': 254,
 'w_98baff9': 255,
 'w_fc433f7': 256,
 'w_6ba46a1': 257,
 'w_fcdc898': 258,
 'w_f63d8b0': 259,
 'w_15d7ecf': 260,
 'w_aa7a302': 261,
 'w_1917275': 262,
 'w_18a1bf2': 263,
 'w_989fbbb': 264,
 'w_0b3b659': 265,
 'w_d96a0cd': 266,
 'w_2e42cc0': 267,
 'w_d6a6360': 268,
 'w_0f8b515': 269,
 'w_517a3bc': 270,
 'w_bbd9409': 271,
 'w_6871147': 272,
 'w_0466071': 273,
 'w_2486665': 274,
 'w_654a5bb': 275,
 'w_4e52a49': 276,
 'w_4e7fc3e': 277,
 'w_3320e76': 278,
 'w_ad6edbe': 279,
 'w_25fdcfb': 280,
 'w_d0e5d1d': 281,
 'w_0d39a68': 282,
 'w_b9c9129': 283,
 'w_ec354a5': 284,
```

```
'w_a7d4668': 285,
'w_a526492': 286,
'w_44a6b62': 287,
'w_7307089': 288,
'w_f0612be': 289,
'w_0b0d88d': 290,
'w_df82632': 291,
'w_9c5ed68': 292,
'w_f6086ac': 293,
'w_aeca19c': 294,
'w_dbda0d6': 295,
'w_fba3bde': 296,
'w_c13a4e3': 297,
'w_1dc6d62': 298,
'w_6d17527': 299,
'w_eb44149': 300,
'w_cd70e8b': 301,
'w_41901e5': 302,
'w_7e841fa': 303,
'w_60c5ba0': 304,
'w_cf00b01': 305,
'w_dcb4be9': 306,
'w_a965f14': 307,
'w_9072f07': 308,
'w_1489751': 309,
'w_e3c119c': 310,
'w_ba8c7bc': 311,
'w_ff70408': 312,
'w_50e125b': 313,
'w_639aed5': 314,
'w_bb2d34d': 315,
'w_79774a5': 316,
'w_77e1ae3': 317,
'w_27da7aa': 318,
'w_d47e2e3': 319,
'w_90f5d3b': 320,
'w_98413c8': 321,
'w_7bcc2d6': 322,
'w_40e3ee7': 323,
'w_5126776': 324,
'w_269b090': 325,
'w_92d55a6': 326,
'w_655bba3': 327,
'w_9621342': 328,
'w_8c3db0a': 329,
'w_2fcb559': 330,
'w_bf4880a': 331,
'w_0b04c08': 332,
'w_f792125': 333,
'w_b6c4efa': 334,
'w_c71bf0e': 335,
'w_6c803bf': 336,
'w_2b443f8': 337,
'w_290f82b': 338,
'w_9948a99': 339,
'w_2b28681': 340,
'w_0a565c5': 341,
```

```
'w_b1d054c': 342,
'w_d28208f': 343,
'w_c00534d': 344,
'w_87cb6bd': 345,
'w_57c14db': 346,
'w_6e742e5': 347,
'w_3b5403b': 348,
'w_4b7b80b': 349,
'w_5c59d12': 350,
'w_fe96bef': 351,
'w_3278f8c': 352,
'w_ce9b95e': 353,
'w_0a0cf7d': 354,
'w_06b6f60': 355,
'w_c786765': 356,
'w_7cca153': 357,
'w_613723e': 358,
'w_8cf18ac': 359,
'w_0acce53': 360,
'w_33fc58d': 361,
'w_8201aa8': 362,
'w_480b2d4': 363,
'w_462a117': 364,
'w_a59905f': 365,
'w_2811cea': 366,
'w_b678944': 367,
'w_a913945': 368,
'w_c622a3f': 369,
'w_17136dc': 370,
'w_3cf3853': 371,
'w_8963cff': 372,
'w_5f68f56': 373,
'w_7104905': 374,
'w_8103039': 375,
'w_ef89b72': 376,
'w_83e0203': 377,
'w_02c2248': 378,
'w_29c9595': 379,
'w_8533569': 380,
'w_0e96943': 381,
'w_a533837': 382,
'w_f9d27ad': 383,
'w_640f91d': 384,
'w_0d48a7d': 385,
'w_32037e2': 386,
'w_48bcad8': 387,
'w_8459e39': 388,
'w_9d136ff': 389,
'w_eb0a6ed': 390,
'w_35c8057': 391,
'w_58c3bf9': 392,
'w_70715c1': 393,
'w_693c9ee': 394,
'w_981a39b': 395,
'w_1c2fb13': 396,
'w_acee288': 397,
'w_569c2c7': 398,
```

```
'w_aaf443d': 399,
'w_a1fe45d': 400,
'w_fa32440': 401,
'w_67e6b75': 402,
'w_3fffc61': 403,
'w_505a141': 404,
'w_0ea659e': 405,
'w_aa446ad': 406,
'w_5e04db0': 407,
'w_b066aa6': 408,
'w_347b648': 409,
'w_0899118': 410,
'w_f6d32e2': 411,
'w_2071a4c': 412,
'w_6af9dd7': 413,
'w_849b126': 414,
'w_f5cc2fd': 415,
'w_414f402': 416,
'w_7028d77': 417,
'w_b29214f': 418,
'w_4c9d3df': 419,
'w_1090920': 420,
'w_ef2f0f0': 421,
'w_a646643': 422,
'w_9967edd': 423,
'w_a21cc97': 424,
'w_5a6d315': 425,
'w_680e011': 426,
'w_96c141f': 427,
'w_02fce90': 428,
'w_26dd948': 429,
'w_d8f848c': 430,
'w_d224115': 431,
'w_e0e133a': 432,
'w_fcd8828': 433,
'w_5a0d250': 434,
'w_dc89c4c': 435,
'w_cece268': 436,
'w_ba53619': 437,
'w_c103b25': 438,
'w_b6a95e0': 439,
'w_4ec7c66': 440,
'w_f9a09c6': 441,
'w_6d274b2': 442,
'w_6f580eb': 443,
'w_07e92ee': 444,
'w_0853262': 445,
'w_ceffa10': 446,
'w_851a7f4': 447,
'w_49b425b': 448,
'w_1596a47': 449,
'w_398aa7f': 450,
'w_8a3449f': 451,
'w_6992521': 452,
'w_8fb79a2': 453,
'w_9989964': 454,
'w_18a854b': 455,
```

```
'w_b074f5e': 456,
'w_eff7e35': 457,
'w_01c2cb0': 458,
'w_1e31d24': 459,
'w_8de6989': 460,
'w_3350425': 461,
'w_3b0894d': 462,
'w_ba39446': 463,
'w_a22afc6': 464,
'w_5c6215c': 465,
'w_83b2cab': 466,
'w_94cd45e': 467,
'w_fd07344': 468,
'w_7ed3719': 469,
'w_28ce17c': 470,
'w_3c2ec7a': 471,
'w_2fd21ec': 472,
'w_549fbeb': 473,
'w_823fcbb': 474,
'w_23dce10': 475,
'w_ab161f7': 476,
'w_0793503': 477,
'w_a34c992': 478,
'w_f4e0748': 479,
'w_77e1d3f': 480,
'w_4696a5a': 481,
'w_c2580ef': 482,
'w_63d10a1': 483,
'w_64d8a6d': 484,
'w_b721e96': 485,
'w_66c1b54': 486,
'w_a0fc1a7': 487,
'w_2d29ddd': 488,
'w_330f897': 489,
'w_f8c3a63': 490,
'w_4855bd3': 491,
'w_9291cdf': 492,
'w_a5a13d0': 493,
'w_9b4a5df': 494,
'w_fa96a3d': 495,
'w_38e4aae': 496,
'w_3fc7f1e': 497,
'w_3d12652': 498,
'w_b6301f8': 499,
'w_f478955': 500,
'w_e685c80': 501,
'w_49bbc79': 502,
'w_19a5685': 503,
'w_67599af': 504,
'w_5f50b5e': 505,
'w_2ce9f0c': 506,
'w_9688819': 507,
'w_efd3f81': 508,
'w_699be3a': 509,
'w_8507226': 510,
'w_fe36b07': 511,
'w_42c631a': 512,
```

```
 'w_2282bb8': 513,
 'w_d4bc10d': 514,
 'w_0ee4d6d': 515,
 'w_d51a79f': 516,
 'w_cf69291': 517,
 'w_690fc6b': 518,
 'w_d5ec83f': 519,
 'w_515838e': 520,
 'w_9771603': 521,
 'w_73b705e': 522,
 'w_01a51a6': 523,
 'w_e39371c': 524,
 'w_8697f5f': 525,
 'w_0c42dba': 526,
 'w_f5b8faf': 527,
 'w_53951e6': 528,
 'w_3fc3d07': 529,
 'w_2e4fecc': 530,
 'w_1a5e7a2': 531,
 'w_95af6a2': 532,
 'w_dc97dc3': 533,
 'w_b04d4f1': 534,
 'w_0f41afe': 535,
 'w_08ddb50': 536,
 'w_ea90267': 537,
 'w_5f9fd41': 538,
 'w_8aab1d4': 539,
 'w_4bf5a06': 540,
 'w_89e159a': 541,
 'w_e4f1fcc': 542,
 'w_0c6dbbe': 543,
 'w_09dd18c': 544,
 'w_a22f338': 545,
 'w_883b284': 546,
 'w_7419e4b': 547,
 'w_c493795': 548,
 'w_f79b3f9': 549,
 'w_7885601': 550,
 'w_81df82e': 551,
 'w_f4b95e7': 552,
 'w_f03ca16': 553,
 'w_ca0ec7c': 554,
 'w_cc009e5': 555,
 'w_25d7f93': 556,
 'w_432b162': 557,
 'w_1e5a146': 558,
 'w_b8dff40': 559,
 'w_74011a2': 560,
 'w_be20b46': 561,
 'w_c0d494d': 562,
 'w_491607b': 563,
 'w_6092d5c': 564,
 'w_76106aa': 565,
 'w_740dfd4': 566,
 'w_95f9670': 567,
 'w_861cc1c': 568,
 'w_8114b1b': 569,
```

```
'w_7f6ce8e': 570,
'w_c167245': 571,
'w_1a5beb9': 572,
'w_3ee8570': 573,
'w_0e4f53c': 574,
'w_c26666e': 575,
'w_c8bf168': 576,
'w_8867074': 577,
'w_686408a': 578,
'w_01319fa': 579,
'w_5436d75': 580,
'w_7c943ab': 581,
'w_19dc50f': 582,
'w_831f124': 583,
'w_fc76ede': 584,
'w_01a99a5': 585,
'w_87050a3': 586,
'w_37bd99a': 587,
'w_3af4e73': 588,
'w_b6689cc': 589,
'w_da9254f': 590,
'w_75378d3': 591,
'w_3380eb0': 592,
'w_6047e81': 593,
'w_f6b952e': 594,
'w_d96c946': 595,
'w_a31fed5': 596,
'w_43be268': 597,
'w_20da4cf': 598,
'w_73c9ba2': 599,
'w_4cedbf8': 600,
'w_65efe4d': 601,
'w_75f217b': 602,
'w_fd218ba': 603,
'w_ffcd98e': 604,
'w_43b50e5': 605,
'w_206e903': 606,
'w_c2ea3f0': 607,
'w_fb8ca1e': 608,
'w_0e737d0': 609,
'w_ed117b3': 610,
'w_ffa78a5': 611,
'w_76d2cce': 612,
'w_33c3ce1': 613,
'w_d9aab0a': 614,
'w_0869575': 615,
'w_718ce15': 616,
'w_35c5d43': 617,
'w_fe054f3': 618,
'w_243e33e': 619,
'w_f90b695': 620,
'w_c99f5ef': 621,
'w_e62a48d': 622,
'w_cb6a206': 623,
'w_4a1e58b': 624,
'w_1f10750': 625,
'w_c06dd6e': 626,
```

```
 'w_002222a': 627,
 'w_b70a0c4': 628,
 'w_80211fd': 629,
 'w_2bea4c4': 630,
 'w_a19eecd': 631,
 'w_7e8b270': 632,
 'w_721d2e9': 633,
 'w_93aee9e': 634,
 'w_2c3f440': 635,
 'w_44ef2b8': 636,
 'w_d36f58c': 637,
 'w_95b6cc3': 638,
 'w_bf3449f': 639,
 'w_64830fa': 640,
 'w_a01d903': 641,
 'w_5ea3410': 642,
 'w_e3f7187': 643,
 'w_29fc08a': 644,
 'w_b0aed4a': 645,
 'w_41f0a7e': 646,
 'w_449a34e': 647,
 'w_fcf5284': 648,
 'w_f16e220': 649,
 'w_8f7b653': 650,
 'w_d37b55b': 651,
 'w_6c99c53': 652,
 'w_577e627': 653,
 'w_3d66298': 654,
 'w_648a9a8': 655,
 'w_99c07e8': 656,
 'w_df15cc8': 657,
 'w_637f363': 658,
 'w_d2be6cc': 659,
 'w_159f36b': 660,
 'w_e55a554': 661,
 'w_dfbfe10': 662,
 'w_c751211': 663,
 'w_8d72578': 664,
 'w_4f38350': 665,
 'w_db2baec': 666,
 'w_099ab25': 667,
 'w_7aaa569': 668,
 'w_19f0b15': 669,
 'w_8bdc211': 670,
 'w_c35b3ae': 671,
 'w_d663f4f': 672,
 'w_59eb8ae': 673,
 'w_8329cea': 674,
 'w_9ca943b': 675,
 'w_4133134': 676,
 'w_68820e8': 677,
 'w_5cc2285': 678,
 'w_2f283f3': 679,
 'w_9a21f34': 680,
 'w_451f507': 681,
 'w_31fb9f0': 682,
 'w_3d8c865': 683,
```

```
'w_18eee6e': 684,
'w_aed8c22': 685,
'w_aed6a9a': 686,
'w_f81c626': 687,
'w_e9f85b7': 688,
'w_9e7021b': 689,
'w_61c1f9c': 690,
'w_00cb685': 691,
'w_490c8a0': 692,
'w_8325a79': 693,
'w_28c8f60': 694,
'w_d84ed05': 695,
'w_8431164': 696,
'w_fca13e2': 697,
'w_1948625': 698,
'w_45b90d9': 699,
'w_f369a80': 700,
'w_d8e752e': 701,
'w_6dc88a6': 702,
'w_b71d930': 703,
'w_89d9c03': 704,
'w_3565288': 705,
'w_028ca0d': 706,
'w_1272a31': 707,
'w_5010531': 708,
'w_f0a4a2a': 709,
'w_f1396bd': 710,
'w_a1a707e': 711,
'w_494dd45': 712,
'w_193b7e3': 713,
'w_9a1d8e7': 714,
'w_4a78bf2': 715,
'w_9c3db0a': 716,
'w_b0b275e': 717,
'w_b478c13': 718,
'w_705e0fb': 719,
'w_7763134': 720,
'w_d7ffaf2': 721,
'w_19212f0': 722,
'w_6710cf8': 723,
'w_af9a47a': 724,
'w_cd01afb': 725,
'w_cb5ea24': 726,
'w_2250fa9': 727,
'w_ee17a08': 728,
'w_46e45e6': 729,
'w_02d5fad': 730,
'w_6545984': 731,
'w_5b1fa1d': 732,
'w_4c8e7ae': 733,
'w_f87b77b': 734,
'w_92585eb': 735,
'w_aae82f8': 736,
'w_73f8bd3': 737,
'w_daeb296': 738,
'w_acc33ca': 739,
'w_bed6af1': 740,
```

```
'w_8b56cb1': 741,
'w_da0f481': 742,
'w_94a4216': 743,
'w_f0f56dc': 744,
'w_2f81b85': 745,
'w_e0799da': 746,
'w_fed0031': 747,
'w_8489554': 748,
'w_4285435': 749,
'w_0a97a25': 750,
'w_e30e97f': 751,
'w_7a2dfd7': 752,
'w_1dc66b7': 753,
'w_5d335c3': 754,
'w_6bbaa92': 755,
'w_89f463f': 756,
'w_bc93e4a': 757,
'w_e73d83c': 758,
'w_17b33ae': 759,
'w_ed6da70': 760,
'w_90ec71a': 761,
'w_dab1fb4': 762,
'w_6e42b8e': 763,
'w_7248590': 764,
'w_864cc78': 765,
'w_f02bf23': 766,
'w_871b1b9': 767,
'w_1d84278': 768,
'w_b970272': 769,
'w_c6fceea': 770,
'w_8b1d546': 771,
'w_e58fbe3': 772,
'w_a1350a7': 773,
'w_7b39dda': 774,
'w_daec6d3': 775,
'w_a0ae961': 776,
'w_e3c0ae5': 777,
'w_e62b5d8': 778,
'w_52b0438': 779,
'w_09c1e0b': 780,
'w_4504167': 781,
'w_2d99a0c': 782,
'w_e4616da': 783,
'w_09f825c': 784,
'w_8158b1f': 785,
'w_ae1db8a': 786,
'w_08f1502': 787,
'w_93d42a1': 788,
'w_14c8f15': 789,
'w_cb38960': 790,
'w_e02fe7b': 791,
'w_a402e24': 792,
'w_57e8a80': 793,
'w_813c5be': 794,
'w_3d2724b': 795,
'w_9146eae': 796,
'w_1ac4c38': 797,
```

```
 'w_2725793': 798,
 'w_ed5a7c6': 799,
 'w_fb4d1f1': 800,
 'w_21dfc18': 801,
 'w_ede13b5': 802,
 'w_a87ac93': 803,
 'w_5966fff': 804,
 'w_aed1e43': 805,
 'w_92d390e': 806,
 'w_86ff8a6': 807,
 'w_7759db0': 808,
 'w_f28de05': 809,
 'w_97bc172': 810,
 'w_f185562': 811,
 'w_74de378': 812,
 'w_00b621b': 813,
 'w_837646b': 814,
 'w_53e9072': 815,
 'w_776ba6e': 816,
 'w_351a1e1': 817,
 'w_c068ed3': 818,
 'w_fe922c3': 819,
 'w_2dbb0fe': 820,
 'w_e0ba9ae': 821,
 'w_c053b99': 822,
 'w_b5c9946': 823,
 'w_680b86e': 824,
 'w_e5ed2d6': 825,
 'w_733fe81': 826,
 'w_ddad87e': 827,
 'w_d3f5be3': 828,
 'w_dab33f6': 829,
 'w_21e178f': 830,
 'w_33b9360': 831,
 'w_c58b474': 832,
 'w_fe6c1f3': 833,
 'w_4791b4c': 834,
 'w_68a7146': 835,
 'w_3b483d3': 836,
 'w_1ed4dde': 837,
 'w_c0f3e88': 838,
 'w_130508d': 839,
 'w_014250a': 840,
 'w_44beff0': 841,
 'w_8e93d0e': 842,
 'w_d3ce445': 843,
 'w_37a7f78': 844,
 'w_e8c35dc': 845,
 'w_57a137f': 846,
 'w_3add848': 847,
 'w_e24a84d': 848,
 'w_e0f1df1': 849,
 'w_ade77e6': 850,
 'w_43e45b6': 851,
 'w_f4219e3': 852,
 'w_80fff4d': 853,
 'w_83714b7': 854,
```

```
'w_7d819df': 855,
'w_da0372d': 856,
'w_b604c18': 857,
'w_91361f0': 858,
'w_8cd7651': 859,
'w_5ba417d': 860,
'w_d9562f1': 861,
'w_aa16da4': 862,
'w_d234e6b': 863,
'w_4ec48a9': 864,
'w_422420c': 865,
'w_dfec7a3': 866,
'w_3c9f80b': 867,
'w_dbc392b': 868,
'w_b0362e2': 869,
'w_4d98ef7': 870,
'w_050bdac': 871,
'w_ebe0ad5': 872,
'w_540fd73': 873,
'w_d6255a4': 874,
'w_993d66c': 875,
'w_17dc953': 876,
'w_1e3ce01': 877,
'w_3db53dc': 878,
'w_0f89e72': 879,
'w_58cb087': 880,
'w_b282692': 881,
'w_2e872af': 882,
'w_33973bf': 883,
'w_136653f': 884,
'w_d345e80': 885,
'w_5a2075e': 886,
'w_5817e08': 887,
'w_7f3a122': 888,
'w_77e3e0f': 889,
'w_bbf20b4': 890,
'w_fe8233d': 891,
'w_e1431ba': 892,
'w_dbc4dcc': 893,
'w_330286e': 894,
'w_1c69443': 895,
'w_8c957e0': 896,
'w_ba17206': 897,
'w_1746c88': 898,
'w_1eaebf2': 899,
'w_0f84bf6': 900,
'w_f283381': 901,
'w_f1b565a': 902,
'w_f790728': 903,
'w_8a2793e': 904,
'w_cd02407': 905,
'w_f460394': 906,
'w_9694b9d': 907,
'w_32602d9': 908,
'w_afde232': 909,
'w_f9083fe': 910,
'w_4f28a45': 911,
```

```
 'w_3d9fb6c': 912,
 'w_7e8fb79': 913,
 'w_ace8c54': 914,
 'w_9f1fafb': 915,
 'w_2e2ba59': 916,
 'w_5dd4772': 917,
 'w_9854838': 918,
 'w_6d05f7f': 919,
 'w_514c62c': 920,
 'w_a2564cf': 921,
 'w_0c8967d': 922,
 'w_2901dbf': 923,
 'w_c1540a3': 924,
 'w_bc5beaa': 925,
 'w_e9cacbf': 926,
 'w_ca40961': 927,
 'w_6e47e0e': 928,
 'w_032d44d': 929,
 'w_06dbe6b': 930,
 'w_f208155': 931,
 'w_338b130': 932,
 'w_ce7c6c0': 933,
 'w_6132293': 934,
 'w_cb32cb8': 935,
 'w_dcb1f2a': 936,
 'w_72e70e5': 937,
 'w_9b804bd': 938,
 'w_930bc39': 939,
 'w_76d5723': 940,
 'w_7f81114': 941,
 'w_c87651d': 942,
 'w_104cc93': 943,
 'w_2658649': 944,
 'w_fbcb6e4': 945,
 'w_67fecca': 946,
 'w_e6d89c0': 947,
 'w_18fbec1': 948,
 'w_ba9bc6d': 949,
 'w_47148ca': 950,
 'w_419226b': 951,
 'w_61e1076': 952,
 'w_5619521': 953,
 'w_96fd936': 954,
 'w_5a1b758': 955,
 'w_29cc48b': 956,
 'w_54fc5b3': 957,
 'w_987a36f': 958,
 'w_deb33de': 959,
 'w_ccf547c': 960,
 'w_741861e': 961,
 'w_2709cfc': 962,
 'w_82c9c67': 963,
 'w_55647bd': 964,
 'w_78f2e92': 965,
 'w_e61dd6d': 966,
 'w_0cc4a2b': 967,
 'w_d6d502a': 968,
```

```
'w_9c61d57': 969,
'w_5d50ea2': 970,
'w_19f7a8b': 971,
'w_5d2734c': 972,
'w_27b9e86': 973,
'w_0eb2886': 974,
'w_b39c722': 975,
'w_edcb241': 976,
'w_cd38536': 977,
'w_8c1e2e4': 978,
'w_50729c4': 979,
'w_ab4bd59': 980,
'w_7ab9a17': 981,
'w_7377b2b': 982,
'w_238bbbf': 983,
'w_735ce7d': 984,
'w_a846944': 985,
'w_522ba14': 986,
'w_1310342': 987,
'w_4659acf': 988,
'w_ddf14ae': 989,
'w_71ec55a': 990,
'w_c5e13f8': 991,
'w_dfd7ee8': 992,
'w_3694c7d': 993,
'w_87782a0': 994,
'w_c8f7bcd': 995,
'w_b7d5069': 996,
'w_6aad777': 997,
'w_dd76ce2': 998,
'w_b4ad62f': 999,
...}
```

In [21]:
```
label_series
```

```
Out[21]: new_whale    810
         w_1287fbc     34
         w_98baff9     27
         w_7554f44     26
         w_1eafe46     23
         w_ab4cae2     22
         w_693c9ee     22
         w_fd1cb9d     22
         w_987a36f     21
         w_73d5489     21
         w_43be268     21
         w_f19faeb     20
         w_95874a5     19
         w_9b401eb     19
         w_b7d5069     18
         w_c0d494d     18
         w_0e737d0     17
         w_eb0a6ed     17
         w_18eee6e     17
         w_dbda0d6     17
         w_67de30b     16
         w_b0e05b1     16
         w_6c803bf     16
         w_a59905f     16
         w_17ee910     16
         w_9ca943b     15
         w_ee17a08     15
         w_89e159a     15
         w_cae7677     15
         w_8c1ec28     14
                      ...
         w_8bcf29b      1
         w_6460698      1
         w_4225bb3      1
         w_c07f119      1
         w_ca5abbb      1
         w_8c408dc      1
         w_5f6fb4e      1
         w_34a0eab      1
         w_34c8690      1
         w_945aefe      1
         w_68a7146      1
         w_f7fed13      1
         w_fb270f3      1
         w_a9f41fd      1
         w_b820615      1
         w_c666071      1
         w_01a99a5      1
         w_f801078      1
         w_00d8453      1
         w_008c602      1
         w_5d2734c      1
         w_b1a4f29      1
         w_8963cff      1
         w_29c286a      1
         w_ce269ec      1
         w_397cb24      1
```

```
w_23b01a6      1
w_3050553      1
w_c07076c      1
w_46b211d      1
Name: Id, Length: 4251, dtype: int64
```

In [22]:
```
type(label_series)
```

Out[22]:
```
pandas.core.series.Series
```

In [23]:
```
data_dir = "train"
```

Open Image in RGB Mode

In [26]:
```
class WhaleDataset(td.Dataset):
    def __init__(self, data_dir, label_csv, total_csv,mode,image_size=(2
24, 224)):
        super(WhaleDataset, self).__init__()
        self.image_size = image_size
        self.mode = mode
        self.data = label_csv
        self.images_dir = data_dir
        self.total_data = total_csv
    def __len__(self):
        return len(self.data)
    def __repr__(self):
        return "BirdsDataset(mode={}, image_size={})". \
            format(self.mode, self.image_size)
    def __getitem__(self, idx):
        img_path = os.path.join(self.images_dir, self.data.iloc[idx]['Im
age'])

        img = Image.open(img_path).convert('RGB')
        #stack_img = np.stack((img,)*3, axis=-1)
        #img = Image.fromarray(stack_img, 'RGB')


        transform = tv.transforms.Compose([tv.transforms.Resize(self.ima
ge_size),
                                           tv.transforms.ToTensor(),
                                           tv.transforms.Normalize(mean=
[0.5,0.5,0.5],std=[0.5,0.5,0.5])
                                          # COMPLETE
                                          ])
        x = transform(img)
        d = Id_classes_map[self.data.iloc[idx]['Id']]
        return x, d

    def number_of_classes(self):
        return self.total_data['Id'].nunique()
```

In [30]:
```
def myimshow(image, ax=plt):
    image = image.to('cpu').numpy()
    image = np.moveaxis(image, [0, 1, 2], [2, 0, 1])
    image = (image + 1) / 2
    image[image < 0] = 0
    image[image > 1] = 1
    h = ax.imshow(image)
    ax.axis('off')

    return h
```

In [31]:
```
train_set = WhaleDataset(data_dir = data_dir,
                         label_csv = data_train,
                         total_csv = train_data_label,
                         mode = 'train')
print(data_dir)
```

```
train
```

In [33]:
```
x = train_set.__getitem__(7)
```

In [34]:
```
x
```

Out[34]:
```
(tensor([[[-0.5843, -0.5765, -0.5294,  ..., -0.4039, -0.4510, -0.4510],
          [-0.6078, -0.6078, -0.6000,  ..., -0.4275, -0.4510, -0.4510],
          [-0.6314, -0.6314, -0.6235,  ..., -0.4588, -0.4510, -0.4510],
          ...,
          [-0.4353, -0.4275, -0.4510,  ..., -0.5529, -0.5451, -0.5529],
          [-0.4039, -0.4039, -0.4196,  ..., -0.6000, -0.5765, -0.5843],
          [-0.3804, -0.3804, -0.4353,  ..., -0.6000, -0.5608, -0.576
5]],

         [[-0.1922, -0.1843, -0.1373,  ..., -0.0196, -0.0667, -0.0667],
          [-0.2157, -0.2157, -0.2078,  ..., -0.0431, -0.0667, -0.0667],
          [-0.2392, -0.2392, -0.2314,  ..., -0.0745, -0.0667, -0.0667],
          ...,
          [-0.0431, -0.0353, -0.0588,  ..., -0.0902, -0.0824, -0.0902],
          [-0.0118, -0.0118, -0.0275,  ..., -0.1373, -0.1137, -0.1216],
          [ 0.0118,  0.0118, -0.0431,  ..., -0.1765, -0.1451, -0.145
1]],

         [[ 0.2706,  0.2784,  0.3255,  ...,  0.5373,  0.4745,  0.4745],
          [ 0.2471,  0.2471,  0.2549,  ...,  0.4980,  0.4745,  0.4745],
          [ 0.2235,  0.2235,  0.2314,  ...,  0.4431,  0.4745,  0.4745],
          ...,
          [ 0.4196,  0.4275,  0.4039,  ...,  0.3490,  0.3569,  0.3490],
          [ 0.4510,  0.4510,  0.4353,  ...,  0.3020,  0.3255,  0.3176],
          [ 0.4745,  0.4745,  0.4196,  ...,  0.2784,  0.3098,  0.302
0]]]),
 3480)
```

In [35]:
```
myimshow(x[0])
```

Out[35]: <matplotlib.image.AxesImage at 0x7f2544013ba8>

In [36]:
```
x[1]
```

Out[36]: 3480

In [37]:
```
len(data_val)
```

Out[37]: 1970

In [38]:
```
len(data_train)
```

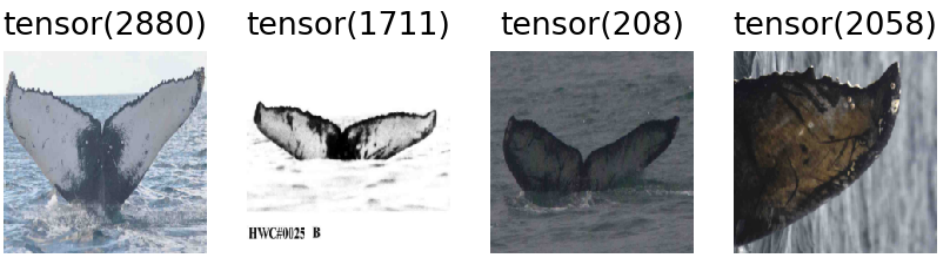
Out[38]: 7880

In [39]:
```
train_loader = td.DataLoader(train_set, batch_size=16, shuffle=True, pin
_memory=True)
```

In [40]:
```
val_set = WhaleDataset(data_dir = data_dir,
                       label_csv=data_val,
                       total_csv = train_data_label,
                       mode = 'val')
```

In [41]:
```
val_loader = td.DataLoader(val_set, batch_size=16, pin_memory=True)
```

In [42]:
```
num_loop = 0
fig = plt.figure()
for img, label in train_loader:
    num_loop += 1
    if (num_loop <= 4):
        plt.subplot(1, 4, num_loop)
        myimshow(img[0])
        plt.title(str(label[0]))
    else:
        break
```



In [43]:
```
import nntools as nt
```

In [45]:
```
class NNClassifier(nt.NeuralNetwork):
    def __init__(self):
        super(NNClassifier, self).__init__()
        self.cross_entropy = nn.CrossEntropyLoss()
    def criterion(self, y, d):
        return self.cross_entropy(y, d)
```

Training VGG16 Model

In [47]:
```
vgg = tv.models.vgg16_bn(pretrained=True)
```

In [48]:
```
vgg.classifier
```

Out[48]:
```
Sequential(
  (0): Linear(in_features=25088, out_features=4096, bias=True)
  (1): ReLU(inplace)
  (2): Dropout(p=0.5)
  (3): Linear(in_features=4096, out_features=4096, bias=True)
  (4): ReLU(inplace)
  (5): Dropout(p=0.5)
  (6): Linear(in_features=4096, out_features=1000, bias=True)
)
```

In [49]:
```
class VGG16Transfer(NNClassifier):
    def __init__(self, num_classes, fine_tuning=False):
        super(VGG16Transfer, self).__init__()
        vgg = tv.models.vgg16_bn(pretrained=True)
        for param in vgg.parameters():
            param.requires_grad = fine_tuning
        self.features = vgg.features
        self.classifier = vgg.classifier
        # COMPLETE
        num_ftrs = vgg.classifier[6].in_features
        self.classifier[6] = nn.Linear(num_ftrs, num_classes)
    def forward(self, x):
        # COMPLETE
        f = self.features(x)
        f = f.view(-1, 25088)
        y = self.classifier(f)
        return y
```

In [50]:
```
num_classes = train_set.number_of_classes()
```

In [51]:
```
vgg16 = VGG16Transfer(num_classes)
```

In [52]:
```
vgg16
```

```
Out[52]: VGG16Transfer(
           (cross_entropy): CrossEntropyLoss()
           (features): Sequential(
             (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1,
         1))
             (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_ru
         nning_stats=True)
             (2): ReLU(inplace)
             (3): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1,
         1))
             (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_ru
         nning_stats=True)
             (5): ReLU(inplace)
             (6): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil
         _mode=False)
             (7): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1,
         1))
             (8): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_r
         unning_stats=True)
             (9): ReLU(inplace)
             (10): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (11): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (12): ReLU(inplace)
             (13): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
         l_mode=False)
             (14): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (15): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (16): ReLU(inplace)
             (17): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (18): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (19): ReLU(inplace)
             (20): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (21): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (22): ReLU(inplace)
             (23): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
         l_mode=False)
             (24): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (25): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (26): ReLU(inplace)
             (27): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (28): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (29): ReLU(inplace)
             (30): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (31): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
```

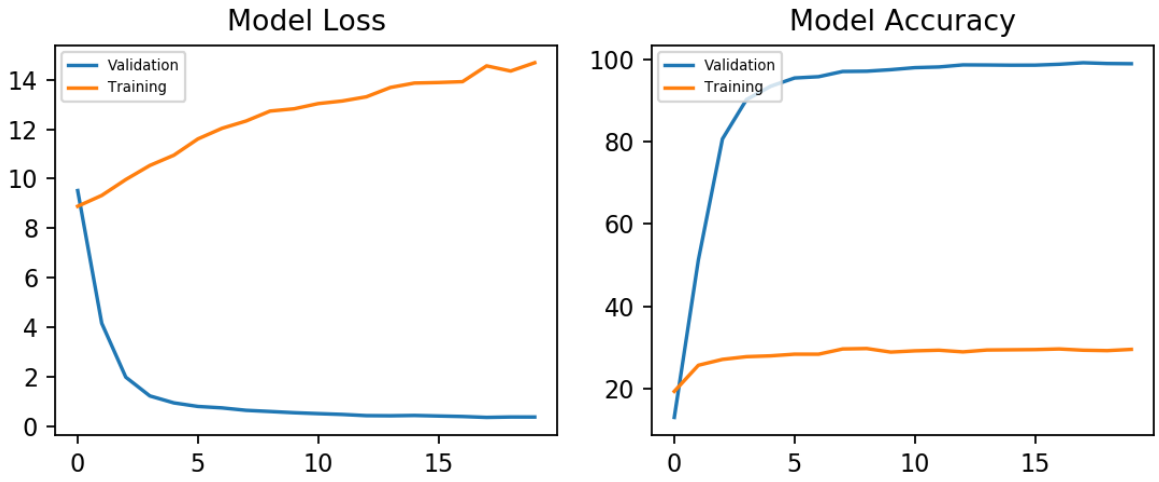
```
running_stats=True)
    (32): ReLU(inplace)
    (33): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
l_mode=False)
    (34): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (35): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (36): ReLU(inplace)
    (37): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (38): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (39): ReLU(inplace)
    (40): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (41): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (42): ReLU(inplace)
    (43): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
l_mode=False)
  )
  (classifier): Sequential(
    (0): Linear(in_features=25088, out_features=4096, bias=True)
    (1): ReLU(inplace)
    (2): Dropout(p=0.5)
    (3): Linear(in_features=4096, out_features=4096, bias=True)
    (4): ReLU(inplace)
    (5): Dropout(p=0.5)
    (6): Linear(in_features=4096, out_features=4251, bias=True)
  )
)
```

In [56]:
```
class ClassificationStatsManager(nt.StatsManager):

    def __init__(self):
        super(ClassificationStatsManager, self).__init__()
    def init(self):
        super(ClassificationStatsManager, self).init()
        self.running_accuracy = 0
    def accumulate(self, loss, x, y, d):
        super(ClassificationStatsManager, self).accumulate(loss, x, y, d
)
        topK_rpob, l = torch.topk(y, 5)
        batchSize = d.size()[0]
        count = 0
        for i in range(batchSize):
            if(d[i] in l[i]):
                count += 1

        self.running_accuracy += count / batchSize

    def summarize(self):
        loss = super(ClassificationStatsManager, self).summarize()
        accuracy = 100 * self.running_accuracy / self.number_update# COM
PLETE
        return {'loss': loss, 'accuracy': accuracy}
```

In [57]:
```
lr = 1e-3
net = VGG16Transfer(num_classes)
net = net.to(device)
adam = torch.optim.Adam(net.parameters(), lr=lr)
stats_manager = ClassificationStatsManager()
exp1 = nt.Experiment(net, train_set, val_set, adam, stats_manager,
                output_dir="whaleclass1_new1", perform_validation_during
_training=True)
```

In [58]:
```
def plot(exp, fig, axes):
    axes[0].clear()
    axes[1].clear()
    axes[0].plot([exp.history[k][0]['loss'] for k in range(exp.epoch)],
                label="training loss")
    axes[0].plot([exp.history[k][1]['loss'] for k in range(exp.epoch)],
label="evaluation loss")
    axes[0].legend(('Validation','Training'), fontsize=6, loc=0)
    axes[0].title.set_text('Model Loss')

    axes[1].plot([exp.history[k][0]['accuracy'] for k in range(exp.epoch
)],
                label="training accuracy")
    axes[1].plot([exp.history[k][1]['accuracy'] for k in range(exp.epoch
)], label="evaluation accuracy")
    # COMPLETE
    axes[1].legend(('Validation','Training'), fontsize=6, loc=2)
    axes[1].title.set_text('Model Accuracy')

    plt.tight_layout()
    fig.canvas.draw()
```

In [59]:
```
fig, axes = plt.subplots(ncols=2, figsize=(7, 3))
exp1.run(num_epochs=20, plot=lambda exp: plot(exp, fig=fig, axes=axes))
```




```
Start/Continue training from epoch 20
Finish training for 20 epochs
```

In [60]:
```
net.eval()
```

```
Out[60]: VGG16Transfer(
           (cross_entropy): CrossEntropyLoss()
           (features): Sequential(
             (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1,
         1))
             (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_ru
         nning_stats=True)
             (2): ReLU(inplace)
             (3): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1,
         1))
             (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_ru
         nning_stats=True)
             (5): ReLU(inplace)
             (6): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil
         _mode=False)
             (7): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1,
         1))
             (8): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_r
         unning_stats=True)
             (9): ReLU(inplace)
             (10): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (11): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (12): ReLU(inplace)
             (13): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
         l_mode=False)
             (14): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (15): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (16): ReLU(inplace)
             (17): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (18): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (19): ReLU(inplace)
             (20): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (21): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (22): ReLU(inplace)
             (23): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
         l_mode=False)
             (24): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (25): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (26): ReLU(inplace)
             (27): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (28): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
         running_stats=True)
             (29): ReLU(inplace)
             (30): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
         (1, 1))
             (31): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
```

```
running_stats=True)
    (32): ReLU(inplace)
    (33): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
l_mode=False)
    (34): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (35): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (36): ReLU(inplace)
    (37): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (38): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (39): ReLU(inplace)
    (40): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (41): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (42): ReLU(inplace)
    (43): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
l_mode=False)
  )
  (classifier): Sequential(
    (0): Linear(in_features=25088, out_features=4096, bias=True)
    (1): ReLU(inplace)
    (2): Dropout(p=0.5)
    (3): Linear(in_features=4096, out_features=4096, bias=True)
    (4): ReLU(inplace)
    (5): Dropout(p=0.5)
    (6): Linear(in_features=4096, out_features=4251, bias=True)
  )
)
```

In [61]:
```
print(exp1.evaluate())
```

```
{'loss': 14.685720079313448, 'accuracy': 29.522357723577237}
```

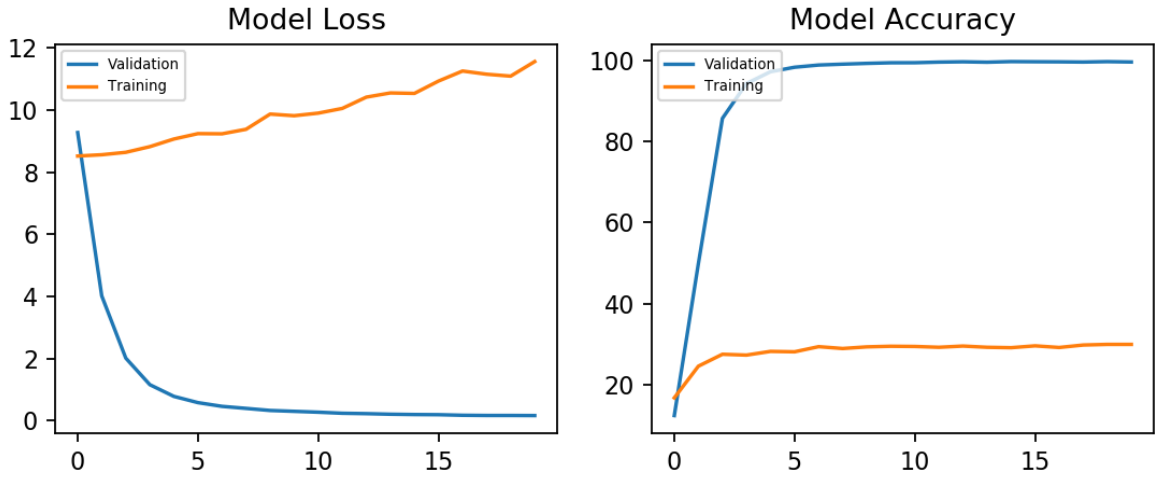
VGG16 Accuracy is 29.52% (Open image in RGB Mode, without doing Data Augmentation)

Training ResNet Model

In [70]:
```
class ClassificationStatsManager(nt.StatsManager):

    def __init__(self):
        super(ClassificationStatsManager, self).__init__()
    def init(self):
        super(ClassificationStatsManager, self).init()
        self.running_accuracy = 0
    def accumulate(self, loss, x, y, d):
        super(ClassificationStatsManager, self).accumulate(loss, x, y, d
)
        topK_rpob, l = torch.topk(y, 5)
        batchSize = d.size()[0]
        count = 0
        for i in range(batchSize):
            if(d[i] in l[i]):
                count += 1

        self.running_accuracy += count / batchSize

    def summarize(self):
        loss = super(ClassificationStatsManager, self).summarize()
        accuracy = 100 * self.running_accuracy / self.number_update# COM
PLETE
        return {'loss': loss, 'accuracy': accuracy}
```

In [71]:
```
class Resnet18Transfer (NNClassifier):
    def __init__(self, num_classes, fine_tuning=False):
        super(Resnet18Transfer, self).__init__()
        resnet = tv.models.resnet18(pretrained=True)
        for param in resnet.parameters():
            param.requires_grad = fine_tuning

        self.model = resnet
        num_ftrs = resnet.fc.in_features
        self.model.fc = nn.Linear(num_ftrs, num_classes)

    def forward(self, x):
        y = self.model(x)
        return y
```

In [73]:
```
fig2, axes2 = plt.subplots(ncols=2, figsize=(7, 3))

lr = 1e-3
net = Resnet18Transfer(num_classes)
net = net.to(device)
adam = torch.optim.Adam(net.parameters(), lr=lr)
stats_manager = ClassificationStatsManager()
exp2 = nt.Experiment(net, train_set, val_set, adam, stats_manager,output
_dir="whaleclass2_new_final", perform_validation_during_training=True)
exp2.run(num_epochs=20, plot=lambda exp: plot(exp, fig=fig2, axes=axes2
))
```




```
Start/Continue training from epoch 0
Epoch 1 (Time: 191.71s)
Epoch 2 (Time: 199.38s)
Epoch 3 (Time: 187.57s)
Epoch 4 (Time: 191.23s)
Epoch 5 (Time: 207.38s)
Epoch 6 (Time: 199.83s)
Epoch 7 (Time: 195.57s)
Epoch 8 (Time: 192.79s)
Epoch 9 (Time: 202.32s)
Epoch 10 (Time: 205.47s)
Epoch 11 (Time: 208.89s)
Epoch 12 (Time: 206.53s)
Epoch 13 (Time: 200.80s)
Epoch 14 (Time: 183.06s)
Epoch 15 (Time: 189.10s)
Epoch 16 (Time: 195.62s)
Epoch 17 (Time: 194.72s)
Epoch 18 (Time: 195.62s)
Epoch 19 (Time: 192.97s)
Epoch 20 (Time: 200.95s)
Finish training for 20 epochs
```

In [74]:
```
net = Resnet18Transfer(train_set.number_of_classes()).to(device)
```

In [75]:
```
print(exp2.evaluate())
```

```
{'loss': 11.561711218298935, 'accuracy': 29.878048780487806}
```

Data Augmentation Approach For Optimization and Using VGG16 Model

In [76]:
```
label_series = label_series.to_dict()
```

In [77]:
```
label_series
```

```
Out[77]: {'new_whale': 810,
          'w_1287fbc': 34,
          'w_98baff9': 27,
          'w_7554f44': 26,
          'w_1eafe46': 23,
          'w_ab4cae2': 22,
          'w_693c9ee': 22,
          'w_fd1cb9d': 22,
          'w_987a36f': 21,
          'w_73d5489': 21,
          'w_43be268': 21,
          'w_f19faeb': 20,
          'w_95874a5': 19,
          'w_9b401eb': 19,
          'w_b7d5069': 18,
          'w_c0d494d': 18,
          'w_0e737d0': 17,
          'w_eb0a6ed': 17,
          'w_18eee6e': 17,
          'w_dbda0d6': 17,
          'w_67de30b': 16,
          'w_b0e05b1': 16,
          'w_6c803bf': 16,
          'w_a59905f': 16,
          'w_17ee910': 16,
          'w_9ca943b': 15,
          'w_ee17a08': 15,
          'w_89e159a': 15,
          'w_cae7677': 15,
          'w_8c1ec28': 14,
          'w_540fd73': 14,
          'w_7e8b270': 14,
          'w_b074cdf': 14,
          'w_2d99a0c': 14,
          'w_6202983': 13,
          'w_fe49bc4': 13,
          'w_44cccf6': 13,
          'w_bb2d34d': 13,
          'w_f8e6546': 13,
          'w_9ea2cc3': 13,
          'w_4e52a49': 13,
          'w_4b7b80b': 13,
          'w_fba3bde': 13,
          'w_7c7a78c': 12,
          'w_49bbc79': 12,
          'w_b9e00eb': 12,
          'w_7b035cc': 12,
          'w_b678944': 12,
          'w_c9e1cdc': 12,
          'w_2071a4c': 12,
          'w_8c1e2e4': 11,
          'w_b942708': 11,
          'w_da2efe0': 11,
          'w_7311fe4': 11,
          'w_3d66298': 11,
          'w_a254eb0': 11,
          'w_e826a6f': 11,
```

```
'w_1a70685': 10,
'w_38e4aae': 10,
'w_0a97a25': 10,
'w_6af9dd7': 10,
'w_654a5bb': 10,
'w_3d0bc7a': 10,
'w_4fd48e7': 10,
'w_1596a47': 10,
'w_3674103': 10,
'w_97f5054': 9,
'w_c30959a': 9,
'w_8d83172': 9,
'w_ace8c54': 9,
'w_d88328d': 9,
'w_d663f4f': 9,
'w_abe383e': 9,
'w_861cc1c': 9,
'w_fea7fe6': 9,
'w_64f3545': 9,
'w_656afeb': 9,
'w_b4732ef': 9,
'w_59eb8ae': 9,
'w_3af4e73': 9,
'w_fa32440': 9,
'w_813c5be': 9,
'w_d9aab0a': 9,
'w_1a5e7a2': 9,
'w_8867074': 9,
'w_4fb3c95': 9,
'w_e02fe7b': 9,
'w_d36f58c': 8,
'w_1dff010': 8,
'w_dfbfe10': 8,
'w_02facde': 8,
'w_2fd21ec': 8,
'w_7763134': 8,
'w_b6886e5': 8,
'w_2fe43c7': 8,
'w_f81c626': 8,
'w_4659acf': 8,
'w_21e178f': 8,
'w_778ee6e': 8,
'w_44a6b62': 8,
'w_aa16da4': 8,
'w_9ceb05d': 8,
'w_b96d4e0': 8,
'w_d9adb4f': 8,
'w_13c6b6b': 8,
'w_dcb1f2a': 8,
'w_40323ac': 8,
'w_9875c12': 8,
'w_0e4f53c': 8,
'w_c00534d': 8,
'w_3694c7d': 7,
'w_66c1b54': 7,
'w_9771603': 7,
'w_43b50e5': 7,
```

```
 'w_63db6f9': 7,
 'w_8fab53d': 7,
 'w_a837660': 7,
 'w_569c2c7': 7,
 'w_e54feba': 7,
 'w_0beef28': 7,
 'w_3b0894d': 7,
 'w_b6689cc': 7,
 'w_5b99089': 7,
 'w_1489751': 7,
 'w_87050a3': 7,
 'w_d382236': 7,
 'w_11adaae': 7,
 'w_5ba417d': 7,
 'w_67fecca': 7,
 'w_4e68ddc': 7,
 'w_e3c119c': 7,
 'w_47d2bc6': 7,
 'w_41ed8e8': 7,
 'w_b066aa6': 7,
 'w_f18df18': 7,
 'w_c58b474': 7,
 'w_a14ccaa': 7,
 'w_5297ab3': 7,
 'w_d89b29e': 7,
 'w_d9055d1': 7,
 'w_8ba2066': 7,
 'w_a18d0dc': 7,
 'w_434ad6a': 7,
 'w_1a229eb': 7,
 'w_4c9d3df': 7,
 'w_5f82501': 7,
 'w_62c3998': 7,
 'w_715d152': 7,
 'w_5f50b5e': 7,
 'w_fc7cc24': 7,
 'w_2dae424': 6,
 'w_1d53d9c': 6,
 'w_c3d523d': 6,
 'w_8044362': 6,
 'w_7c943ab': 6,
 'w_511c464': 6,
 'w_fd07344': 6,
 'w_1da7080': 6,
 'w_993d66c': 6,
 'w_62f804d': 6,
 'w_0d48a7d': 6,
 'w_35063ed': 6,
 'w_9b67e87': 6,
 'w_2a939eb': 6,
 'w_12cdfbd': 6,
 'w_42f935c': 6,
 'w_ddbf533': 6,
 'w_b688397': 6,
 'w_da63cba': 6,
 'w_886257d': 6,
 'w_5c6215c': 6,
```

```
'w_ea2385d': 6,
'w_ba8c7bc': 6,
'w_b8dff40': 6,
'w_d5ec83f': 6,
'w_7028d77': 6,
'w_b6410bc': 6,
'w_217e78a': 6,
'w_f6fb689': 6,
'w_ad87135': 6,
'w_e61dd6d': 6,
'w_48a98d7': 6,
'w_5a81425': 6,
'w_883557a': 6,
'w_2a18a44': 6,
'w_94cd45e': 6,
'w_d22af00': 6,
'w_33973bf': 6,
'w_4ebeafb': 6,
'w_71764b4': 6,
'w_f86488a': 6,
'w_db2baec': 6,
'w_4855bd3': 6,
'w_6d17527': 6,
'w_f438375': 6,
'w_77e1d3f': 6,
'w_2f54c3c': 6,
'w_8d46cef': 6,
'w_9ec358c': 6,
'w_71c7322': 6,
'w_3411b9f': 6,
'w_d19a884': 6,
'w_f5b8faf': 6,
'w_4a78bf2': 6,
'w_3197568': 6,
'w_367b996': 6,
'w_78f2e92': 6,
'w_9cb529f': 6,
'w_8b56cb1': 6,
'w_b34793e': 6,
'w_32a920b': 6,
'w_045d9fc': 5,
'w_e0efc4f': 5,
'w_d6df554': 5,
'w_ab6db0f': 5,
'w_2e872af': 5,
'w_a965f14': 5,
'w_3026ce2': 5,
'w_90f5d3b': 5,
'w_02d5fad': 5,
'w_3fc3d07': 5,
'w_7759db0': 5,
'w_aae82f8': 5,
'w_ff7630a': 5,
'w_f0f0dbb': 5,
'w_243e33e': 5,
'w_1274a11': 5,
'w_b3e6069': 5,
```

```
 'w_6547e12': 5,
 'w_c0cfd5b': 5,
 'w_ae07541': 5,
 'w_2730966': 5,
 'w_cce912e': 5,
 'w_2f6a962': 5,
 'w_08f1502': 5,
 'w_06a6351': 5,
 'w_6232813': 5,
 'w_eec1133': 5,
 'w_0b04c08': 5,
 'w_b9f5b8b': 5,
 'w_28ce17c': 5,
 'w_57c8da9': 5,
 'w_3e1ba5b': 5,
 'w_53064a6': 5,
 'w_cd1f1ed': 5,
 'w_d47e2e3': 5,
 'w_516bedb': 5,
 'w_d234e6b': 5,
 'w_56bbc91': 5,
 'w_c1b685f': 5,
 'w_c622a3f': 5,
 'w_9948a99': 5,
 'w_b729b1f': 5,
 'w_53859b2': 5,
 'w_9c5ed68': 5,
 'w_9e1e818': 5,
 'w_e73d83c': 5,
 'w_efd3f81': 5,
 'w_bf38f05': 5,
 'w_bc9dc37': 5,
 'w_d2be6cc': 5,
 'w_6092d5c': 5,
 'w_9db9ef5': 5,
 'w_d3ef4b2': 5,
 'w_eac205a': 5,
 'w_0f96780': 5,
 'w_cd88a48': 5,
 'w_817f022': 5,
 'w_286ec5f': 5,
 'w_bbf20b4': 5,
 'w_dc7c5a0': 5,
 'w_0d39a68': 5,
 'w_2a04ceb': 5,
 'w_cf24f84': 5,
 'w_06972d2': 5,
 'w_d7de1ee': 5,
 'w_c11932e': 5,
 'w_8db45b7': 5,
 'w_2c3f440': 5,
 'w_e7a238b': 5,
 'w_cf00b01': 5,
 'w_e964ae8': 5,
 'w_741861e': 5,
 'w_3a7d86d': 5,
 'w_ff1b64c': 5,
```

```
 'w_18a854b': 5,
 'w_a22f338': 5,
 'w_ffa78a5': 5,
 'w_8e451d9': 5,
 'w_6e8486d': 5,
 'w_837646b': 5,
 'w_637f363': 5,
 'w_7885601': 5,
 'w_932c8cd': 5,
 'w_cb7b682': 5,
 'w_ab1c859': 5,
 'w_64830fa': 5,
 'w_93bf889': 5,
 'w_73b705e': 5,
 'w_53951e6': 5,
 'w_1e3ce01': 5,
 'w_e4616da': 5,
 'w_86113c4': 5,
 'w_9122179': 5,
 'w_71e6583': 5,
 'w_72e70e5': 5,
 'w_cef690d': 5,
 'w_efebfe8': 5,
 'w_398aa7f': 5,
 'w_1948625': 5,
 'w_89d9c03': 5,
 'w_18df014': 5,
 'w_143b201': 5,
 'w_0e9f6d9': 4,
 'w_f478955': 4,
 'w_6361632': 4,
 'w_ee948c6': 4,
 'w_aeca19c': 4,
 'w_ca5f17f': 4,
 'w_3172910': 4,
 'w_61d516e': 4,
 'w_dc89c4c': 4,
 'w_1c69443': 4,
 'w_fe95ab8': 4,
 'w_0819271': 4,
 'w_8c1dab2': 4,
 'w_a335e80': 4,
 'w_6badfcf': 4,
 'w_309a2b3': 4,
 'w_f5eb6c6': 4,
 'w_deb33de': 4,
 'w_dad23fa': 4,
 'w_0c70bc3': 4,
 'w_ddf14ae': 4,
 'w_b9ee5ec': 4,
 'w_5436d75': 4,
 'w_0c6dbbe': 4,
 'w_eb47189': 4,
 'w_95b6cc3': 4,
 'w_d1bbde9': 4,
 'w_e30e97f': 4,
 'w_19dc50f': 4,
```

```
 'w_03c84ef': 4,
 'w_a646643': 4,
 'w_df092eb': 4,
 'w_f0d8be1': 4,
 'w_989fbbb': 4,
 'w_33fc58d': 4,
 'w_dbaa2b1': 4,
 'w_5010531': 4,
 'w_711aaa1': 4,
 'w_17e8554': 4,
 'w_414f402': 4,
 'w_77607b5': 4,
 'w_31fb9f0': 4,
 'w_099ab25': 4,
 'w_c027e4e': 4,
 'w_338b130': 4,
 'w_f02bf23': 4,
 'w_f3bd33a': 4,
 'w_c0d1701': 4,
 'w_8459e39': 4,
 'w_3cf3853': 4,
 'w_7419e4b': 4,
 'w_7f40920': 4,
 'w_e62a48d': 4,
 'w_ffda8b2': 4,
 'w_92585eb': 4,
 'w_37dd956': 4,
 'w_3745f59': 4,
 'w_a3dbc8f': 4,
 'w_3ae3603': 4,
 'w_376a413': 4,
 'w_e3c1ec4': 4,
 'w_5384c57': 4,
 'w_a21cc97': 4,
 'w_a913945': 4,
 'w_8103039': 4,
 'w_735ce7d': 4,
 'w_0fea5a3': 4,
 'w_8ecd3a7': 4,
 'w_dfec7a3': 4,
 'w_1c32062': 4,
 'w_44f66a7': 4,
 'w_2a9727c': 4,
 'w_f4e0748': 4,
 'w_f115d53': 4,
 'w_1632307': 4,
 'w_fdc8bf6': 4,
 'w_ef2f0f0': 4,
 'w_2b443f8': 4,
 'w_e7f8e67': 4,
 'w_a524549': 4,
 'w_1dc66b7': 4,
 'w_76d2cce': 4,
 'w_12f2352': 4,
 'w_1f10750': 4,
 'w_f1a4389': 4,
 'w_a25caa9': 4,
```

```
 'w_d3ce445': 4,
 'w_6d4dedc': 4,
 'w_af890ad': 4,
 'w_fe054f3': 4,
 'w_303518a': 4,
 'w_0771d4b': 4,
 'w_d037229': 4,
 'w_92d390e': 4,
 'w_3f907d6': 4,
 'w_1ac4c38': 4,
 'w_b6c4efa': 4,
 'w_48dd419': 4,
 'w_462a117': 4,
 'w_964c1b3': 4,
 'w_33b9360': 4,
 'w_d141590': 4,
 'w_c27d036': 4,
 'w_96fd936': 4,
 'w_a526492': 4,
 'w_718ce15': 4,
 'w_aa7e5f4': 4,
 'w_94e1ed4': 4,
 'w_532fb22': 4,
 'w_3a9ee71': 4,
 'w_9c3db0a': 4,
 'w_3061bdd': 4,
 'w_f83b1f9': 4,
 'w_147b62b': 4,
 'w_552a16e': 4,
 'w_61e1076': 4,
 'w_3fe1eb9': 4,
 'w_2863d51': 4,
 'w_853c1f7': 4,
 'w_1ae1386': 4,
 'w_c57623d': 4,
 'w_60c5e7e': 4,
 'w_a2a1378': 4,
 'w_392bee3': 4,
 'w_6c899ff': 4,
 'w_db474c7': 4,
 'w_16d1b32': 4,
 'w_af67683': 4,
 'w_0bc712b': 4,
 'w_9d704ab': 4,
 'w_bdbec6c': 4,
 'w_6503ccd': 4,
 'w_0408054': 4,
 'w_ca94288': 4,
 'w_6734e40': 4,
 'w_bf77b13': 4,
 'w_3b3b9b2': 4,
 'w_6ef68ed': 4,
 'w_25d7f93': 4,
 'w_59349ea': 4,
 'w_c10ffe9': 4,
 'w_1dd8e68': 4,
 'w_bf4880a': 4,
```

```
 'w_1969a9d': 4,
 'w_44f0fa2': 4,
 'w_5cc2285': 4,
 'w_5d96ba4': 4,
 'w_c8d20aa': 4,
 'w_7538a4c': 4,
 'w_d3c8520': 4,
 'w_5c62a56': 4,
 'w_2b939eb': 4,
 'w_e45bc18': 4,
 'w_79b42cd': 4,
 'w_f93d780': 4,
 'w_b95307d': 4,
 'w_8bde95b': 4,
 'w_c2474a2': 4,
 'w_2f283f3': 4,
 'w_c6be61e': 4,
 'w_ee9c4d7': 4,
 'w_530f87b': 4,
 'w_9577f88': 4,
 'w_733fe81': 4,
 'w_daeb296': 4,
 'w_372ae75': 4,
 'w_e59294d': 4,
 'w_83714b7': 4,
 'w_67599af': 4,
 'w_e3c0ae5': 4,
 'w_19a5685': 4,
 'w_0988bbb': 4,
 'w_86b3f04': 4,
 'w_576db5d': 4,
 'w_68820e8': 4,
 'w_d6a6360': 4,
 'w_721d2e9': 4,
 'w_3f90907': 4,
 'w_2e4fecc': 4,
 'w_dfd7ee8': 4,
 'w_7db10cc': 4,
 'w_6e47e0e': 4,
 'w_5c07f42': 4,
 'w_89317b0': 4,
 'w_5cb5fc3': 4,
 'w_ff70408': 4,
 'w_83df8d5': 4,
 'w_d96a0cd': 4,
 'w_e5250e9': 4,
 'w_d6e437d': 4,
 'w_7dee51b': 4,
 'w_90201e3': 4,
 'w_024358d': 3,
 'w_272259b': 3,
 'w_0f84bf6': 3,
 'w_51969d2': 3,
 'w_fb7a56b': 3,
 'w_4ec48a9': 3,
 'w_5317c46': 3,
 'w_491607b': 3,
```

```
 'w_186bcab': 3,
 'w_1d05772': 3,
 'w_4378542': 3,
 'w_5e57e8b': 3,
 'w_3650949': 3,
 'w_eb8429c': 3,
 'w_a2b8172': 3,
 'w_62ca7c8': 3,
 'w_d8e752e': 3,
 'w_7ba4b5a': 3,
 'w_01b2250': 3,
 'w_f4ac89a': 3,
 'w_cf3c233': 3,
 'w_1310342': 3,
 'w_92be3ca': 3,
 'w_03a2ed7': 3,
 'w_6d05f7f': 3,
 'w_a91b997': 3,
 'w_7a2dfd7': 3,
 'w_a358522': 3,
 'w_be4d5b8': 3,
 'w_fc433f7': 3,
 'w_b890652': 3,
 'w_74de378': 3,
 'w_f64658b': 3,
 'w_490c8a0': 3,
 'w_acc67ea': 3,
 'w_b6efe77': 3,
 'w_17b33ae': 3,
 'w_0d8fb3f': 3,
 'w_99ad599': 3,
 'w_238bbbf': 3,
 'w_3e0f25d': 3,
 'w_07e92ee': 3,
 'w_7377b2b': 3,
 'w_0ee4d6d': 3,
 'w_17136dc': 3,
 'w_b8f8e69': 3,
 'w_4114553': 3,
 'w_60eba40': 3,
 'w_9868b95': 3,
 'w_42dd100': 3,
 'w_3fcb80e': 3,
 'w_7895123': 3,
 'w_41fa033': 3,
 'w_8d9c6fc': 3,
 'w_29fc831': 3,
 'w_2725793': 3,
 'w_9c61d57': 3,
 'w_048f7a9': 3,
 'w_3feb2df': 3,
 'w_0654dd9': 3,
 'w_aef3680': 3,
 'w_0ac6a0a': 3,
 'w_22bcbd6': 3,
 'w_cd70e8b': 3,
 'w_76b6f7c': 3,
```

```
 'w_affdf5d': 3,
 'w_f208155': 3,
 'w_9b6b87d': 3,
 'w_ebf3f26': 3,
 'w_48bcad8': 3,
 'w_e09e886': 3,
 'w_824f286': 3,
 'w_f21f2ec': 3,
 'w_351a1e1': 3,
 'w_ef7755e': 3,
 'w_4c25641': 3,
 'w_f9a09c6': 3,
 'w_3572e7e': 3,
 'w_7c6ad05': 3,
 'w_9d0e84a': 3,
 'w_361e290': 3,
 'w_69185bb': 3,
 'w_3c4062e': 3,
 'w_e700deb': 3,
 'w_3a78626': 3,
 'w_20c671c': 3,
 'w_1431f4b': 3,
 'w_b0362e2': 3,
 'w_5624f08': 3,
 'w_02c9470': 3,
 'w_3621c49': 3,
 'w_e4d3ec3': 3,
 'w_60759c2': 3,
 'w_9146eae': 3,
 'w_5c23454': 3,
 'w_b0aed4a': 3,
 'w_2b930da': 3,
 'w_e62b5d8': 3,
 'w_4f248f3': 3,
 'w_0ead9d7': 3,
 'w_aed1e43': 3,
 'w_def715a': 3,
 'w_e548eb7': 3,
 'w_e64c9a6': 3,
 'w_4863ed9': 3,
 'w_e8a8d85': 3,
 'w_c0f3e88': 3,
 'w_5ae47d9': 3,
 'w_8397970': 3,
 'w_159f36b': 3,
 'w_50729c4': 3,
 'w_cd65880': 3,
 'w_da6d50b': 3,
 'w_2832e90': 3,
 'w_c263d43': 3,
 'w_2216a46': 3,
 'w_e38b2c7': 3,
 'w_85f9040': 3,
 'w_fbcb6e4': 3,
 'w_3d8c865': 3,
 'w_15f29b7': 3,
 'w_dd90e0a': 3,
```

```
'w_37223f9': 3,
'w_9321960': 3,
'w_c2c4f43': 3,
'w_ecb9a79': 3,
'w_dff77b6': 3,
'w_bd61ca9': 3,
'w_14c8f15': 3,
'w_33e89be': 3,
'w_63d10a1': 3,
'w_4a7080a': 3,
'w_b39c722': 3,
'w_5a6d315': 3,
'w_2dcbf82': 3,
'w_3449e4f': 3,
'w_95aad92': 3,
'w_ca8bfb4': 3,
'w_29d2cec': 3,
'w_0aae8c1': 3,
'w_6047e81': 3,
'w_9434f0c': 3,
'w_242a05d': 3,
'w_05b2ddd': 3,
'w_cf092f5': 3,
'w_0e96943': 3,
'w_c529cf1': 3,
'w_e3f51ec': 3,
'w_f104523': 3,
'w_ba17206': 3,
'w_6a3210a': 3,
'w_09f825c': 3,
'w_d94db94': 3,
'w_30f095d': 3,
'w_8431164': 3,
'w_5ad2030': 3,
'w_6556c5c': 3,
'w_89e451c': 3,
'w_08d1ccd': 3,
'w_19f0b15': 3,
'w_ffa7427': 3,
'w_3349c9d': 3,
'w_2d29ddd': 3,
'w_cbedcb0': 3,
'w_78999a6': 3,
'w_7f999ff': 3,
'w_479e3e5': 3,
'w_86b9dfd': 3,
'w_17dc953': 3,
'w_df15cc8': 3,
'w_8861715': 3,
'w_77e1ae3': 3,
'w_ce29d2a': 3,
'w_7285eb3': 3,
'w_8376b7e': 3,
'w_93b64e9': 3,
'w_dd76ce2': 3,
'w_206e903': 3,
'w_3ee8570': 3,
```

```
'w_e69cef0': 3,
'w_aa7a302': 3,
'w_895c722': 3,
'w_d37b55b': 3,
'w_680e011': 3,
'w_57c14db': 3,
'w_a494171': 3,
'w_c133ea8': 3,
'w_f4224b9': 3,
'w_3f91e04': 3,
'w_a687e28': 3,
'w_09d7946': 3,
'w_e158680': 3,
'w_4b04829': 3,
'w_60cc444': 3,
'w_697c72e': 3,
'w_64433af': 3,
'w_9e22ded': 3,
'w_7586198': 3,
'w_e80641a': 3,
'w_fe2742d': 3,
'w_ab6bb0a': 3,
'w_23cd105': 3,
'w_e685c80': 3,
'w_8f1aa27': 3,
'w_de3cab3': 3,
'w_da0b8b5': 3,
'w_fb72090': 3,
'w_8507226': 3,
'w_edf5f77': 3,
'w_2554558': 3,
'w_d02787f': 3,
'w_a7d4668': 3,
'w_d0b237f': 3,
'w_bc62c0e': 3,
'w_17d5eb9': 3,
'w_25871da': 3,
'w_587d2e3': 3,
'w_2709cfc': 3,
'w_7c44934': 3,
'w_f546b0a': 3,
'w_4c8e7ae': 3,
'w_54fc5b3': 3,
'w_ebe0ad5': 3,
'w_d0157fd': 3,
'w_cc759cd': 3,
'w_87e00c8': 3,
'w_abc2eda': 3,
'w_8a31e6c': 3,
'w_eb39613': 3,
'w_23e4e61': 3,
'w_fb4d1f1': 3,
'w_1c2fb13': 3,
'w_ec354a5': 3,
'w_2cde7c0': 3,
'w_9989964': 3,
'w_b329c00': 3,
```

```
 'w_aca9607': 3,
 'w_01a51a6': 3,
 'w_5919cf7': 3,
 'w_3b5403b': 3,
 'w_dd52cfc': 3,
 'w_851a7f4': 3,
 'w_c80d300': 3,
 'w_6f580eb': 3,
 'w_fb2c3fb': 3,
 'w_b33e728': 3,
 'w_be32bec': 3,
 'w_4b83ce2': 3,
 'w_8f593d4': 3,
 'w_4848a3c': 3,
 'w_9caed77': 3,
 'w_ba53619': 3,
 'w_d708c5a': 3,
 'w_50db782': 3,
 'w_79d4855': 3,
 'w_8e23e4c': 3,
 'w_ddad87e': 3,
 'w_bc8d634': 3,
 'w_8b1ca89': 3,
 'w_54c00ad': 3,
 'w_b29214f': 3,
 'w_65e519b': 3,
 'w_42c631a': 3,
 'w_c8f7bcd': 3,
 'w_4be373e': 3,
 'w_8643ba3': 3,
 'w_f7a0fd7': 3,
 'w_f1b565a': 3,
 'w_b05c57f': 3,
 'w_71bfc77': 3,
 'w_cd02407': 3,
 'w_2ac83b0': 3,
 'w_296749b': 3,
 'w_e6f5e09': 3,
 'w_cd22b23': 3,
 'w_0a91f24': 3,
 'w_f5771d1': 3,
 'w_3f365f3': 3,
 'w_28fa29e': 3,
 'w_12c3d3d': 3,
 'w_fb89186': 3,
 'w_5f7c402': 3,
 'w_dbc4dcc': 3,
 'w_b5383b5': 3,
 'w_6d274b2': 3,
 'w_91572bb': 3,
 'w_e0f1df1': 3,
 'w_20e863f': 3,
 'w_a9dd349': 3,
 'w_7a53e0c': 3,
 'w_1638016': 3,
 'w_38b1edd': 3,
 'w_662a132': 3,
```

```
 'w_6710cf8': 3,
 'w_f843a8d': 3,
 'w_b9c9129': 3,
 'w_70a31ce': 3,
 'w_bf8c3d2': 3,
 'w_3d4900b': 3,
 'w_2bba8c8': 3,
 'w_f7e6199': 3,
 'w_9ae554b': 3,
 'w_5b45503': 3,
 'w_33c3ce1': 3,
 'w_9205d40': 3,
 'w_b4ad62f': 3,
 'w_81b402a': 3,
 'w_056be75': 3,
 'w_fd587bc': 3,
 'w_a8ec52f': 3,
 'w_65428d9': 3,
 'w_3bc8a47': 3,
 'w_cd01afb': 3,
 'w_f774c8b': 3,
 'w_2b0028d': 3,
 'w_cb7d1b5': 3,
 'w_70daaba': 3,
 'w_fe5e78b': 3,
 'w_c84551d': 3,
 'w_865c2ba': 3,
 'w_5dc1c2d': 3,
 'w_8bdc211': 3,
 'w_e6ec8ee': 3,
 'w_a5a13d0': 3,
 'w_bf69a19': 3,
 'w_97da401': 3,
 'w_2658649': 3,
 'w_8fd5636': 3,
 'w_8a66718': 3,
 'w_dee1df3': 3,
 'w_23dce10': 3,
 'w_09dd18c': 3,
 'w_c13a4e3': 3,
 'w_844f032': 3,
 'w_c128384': 3,
 'w_b081817': 3,
 'w_3de676c': 3,
 'w_6dbb861': 3,
 'w_c4c8491': 3,
 'w_5cef366': 3,
 'w_d224115': 3,
 'w_62c548b': 3,
 'w_964ce09': 3,
 'w_c06dd6e': 3,
 'w_90ec71a': 3,
 'w_f03ca16': 3,
 'w_1272a31': 3,
 'w_8d5ede1': 3,
 'w_5dd4772': 3,
 'w_4f38350': 3,
```

```
 'w_8b22583': 3,
 'w_e156c87': 3,
 'w_8b15b96': 3,
 'w_b41e229': 3,
 'w_54a5871': 3,
 'w_0a71e87': 3,
 'w_515838e': 3,
 'w_32037e2': 3,
 'w_f3db3f1': 3,
 'w_8201aa8': 3,
 'w_0ebd514': 3,
 'w_1a5beb9': 3,
 'w_307065e': 3,
 'w_fdf60bb': 3,
 'w_b067417': 3,
 'w_a2eb1bb': 3,
 'w_f746a73': 3,
 'w_6b03eb4': 3,
 'w_7f81114': 3,
 'w_3aa2073': 3,
 'w_bbc2a14': 3,
 'w_a3cd405': 3,
 'w_e55a554': 3,
 'w_24212f5': 3,
 'w_b5e3076': 3,
 'w_0bbb3de': 3,
 'w_d0e5d1d': 3,
 'w_fb94a5a': 3,
 'w_fce6ab2': 3,
 'w_92d55a6': 3,
 'w_3f5e7fc': 3,
 'w_5a2075e': 3,
 'w_00cb685': 3,
 'w_073b15e': 3,
 'w_2282bb8': 3,
 'w_f014da3': 3,
 'w_3b99025': 3,
 'w_676ddb0': 3,
 'w_6fcceaf': 3,
 'w_773509f': 3,
 'w_8464638': 3,
 'w_a22afc6': 3,
 'w_2482b4a': 3,
 'w_74e15fc': 3,
 'w_b47a4a1': 3,
 'w_8158b1f': 3,
 'w_7e5cc5e': 3,
 'w_c926fc3': 3,
 'w_34c4927': 3,
 'w_414a0d7': 3,
 'w_74adf0b': 3,
 'w_eff7e35': 3,
 'w_ceffa10': 3,
 'w_7875b79': 3,
 'w_402784d': 3,
 'w_78ba632': 3,
 'w_dcf2001': 3,
```

```
'w_e1431ba': 3,
'w_22bb9b3': 3,
'w_29670e2': 3,
'w_cece268': 3,
'w_e0e5c9e': 3,
'w_759b647': 3,
'w_5966fff': 3,
'w_17964ef': 3,
'w_70715c1': 3,
'w_c708dc7': 3,
'w_b7a832a': 3,
'w_c95a0ea': 3,
'w_230a0de': 3,
'w_0e7cb1c': 3,
'w_e449d3a': 3,
'w_eea20c4': 3,
'w_218b25e': 3,
'w_7e93962': 3,
'w_9151503': 3,
'w_703e39b': 3,
'w_a1237e3': 3,
'w_38158d6': 3,
'w_5810973': 3,
'w_503ff17': 3,
'w_18a1bf2': 3,
'w_e4f1fcc': 3,
'w_96c141f': 3,
'w_febafc1': 3,
'w_f660d7d': 3,
'w_613723e': 3,
'w_0de84f0': 3,
'w_e0818df': 3,
'w_a0ae961': 3,
'w_616ca36': 3,
'w_7459706': 3,
'w_d398969': 3,
'w_15d7ecf': 3,
'w_222dcb7': 3,
'w_a01d903': 3,
'w_48a5b86': 3,
'w_7b770ec': 3,
'w_d428980': 3,
'w_d6255a4': 3,
'w_1ecfe96': 3,
'w_081dd6e': 3,
'w_35c8057': 3,
'w_f4d89bb': 3,
'w_8de6989': 3,
'w_4fdaa2a': 3,
'w_0d049cf': 3,
'w_02bb4cf': 3,
'w_4b47559': 3,
'w_a1350a7': 3,
'w_2fd73d9': 3,
'w_db3621e': 3,
'w_ead305f': 3,
'w_5817e08': 3,
```

```
 'w_6c2d7ea': 3,
 'w_87e8446': 3,
 'w_900f9eb': 3,
 'w_f792125': 3,
 'w_d9fdd15': 3,
 'w_0740d28': 3,
 'w_681dba6': 3,
 'w_b6ed5d2': 3,
 'w_302af0a': 3,
 'w_cf0c062': 3,
 'w_b2d937a': 3,
 'w_80b1c48': 3,
 'w_9a5cd8e': 3,
 'w_c0a4c9d': 3,
 'w_2db01d5': 3,
 'w_8c605d2': 3,
 'w_9d5f5cc': 3,
 'w_41da937': 3,
 'w_5be8e63': 3,
 'w_a533837': 3,
 'w_1911cbb': 3,
 'w_d09e61a': 3,
 'w_6249f5d': 3,
 'w_968f2ca': 3,
 'w_e3caa40': 3,
 'w_2dbb0fe': 3,
 'w_0013924': 3,
 'w_c1e8594': 3,
 'w_44edba9': 2,
 'w_15427c3': 2,
 'w_b6d09e6': 2,
 ...}
```

In [78]:
```
class AugmentData(td.Dataset):
    def __init__(self, data_dir, label_csv, total_csv,mode,image_size=(2
24, 224)):
        super(AugmentData, self).__init__()
        self.image_size = image_size
        self.mode = mode
        self.data = label_csv
        self.images_dir = data_dir
        self.total_data = total_csv
    def __len__(self):
        return len(self.data)
    def __repr__(self):
        return "BirdsDataset(mode={}, image_size={})". \
            format(self.mode, self.image_size)
    def __getitem__(self, idx):
        img_path = os.path.join(self.images_dir, self.data.iloc[idx]['Im
age'])
#         print(self.images_dir)
#         print(img_path)
        img = Image.open(img_path).convert('RGB')
        transform_list = []

        # data augmentation on the minority class
        transform_list = self.augment_transform(transform_list, idx)


        transform_list.append(tv.transforms.Resize(self.image_size))
        transform_list.append(tv.transforms.ToTensor())
        transform_list.append(
            tv.transforms.Normalize(mean=[0.5,0.5,0.5],std=[0.5,0.5,0.5
]))

        transform = tv.transforms.Compose(transform_list)
        x = transform(img)
        # 1/10 gaussian noise added
        if random.random() < 0.5 :
            x = x + torch.randn_like(x)/10
        d = Id_classes_map[self.data.iloc[idx]['Id']]
        return x, d

    def number_of_classes(self):
        return self.total_data['Id'].nunique()

    def augment_transform(self, transform_list, idx):
        rand_num = random.randint(0,3)
        if rand_num == 0:
            transform_list.append(tv.transforms.ColorJitter(brightness =
0.125))
        elif rand_num == 1:
            transform_list.append(tv.transforms.ColorJitter(contrast =
0.3))
        elif rand_num == 2:
            transform_list.append(tv.transforms.ColorJitter(saturation =
0.3))
        else:
            transform_list.append(tv.transforms.ColorJitter(hue = 0.3))
```

```
        if random.random() < 0.5 :
            transform_list.append(tv.transforms.RandomHorizontalFlip())
        if random.random() < 0.5 :
            transform_list.append(tv.transforms.RandomRotation(25))
        if random.random() < 0.5 :
            transform_list.append(tv.transforms.RandomAffine(degrees = 0
,
                                                             translate =
(0.1,0.15),
                                                             scale = None
,
                                                             shear = None
))
        return transform_list
```

In [79]:
```
# calculate the weighted sample
classes_sample_count = [0 for i in range(len(classes))]
for key in label_series:
    classes_sample_count[Id_classes_map[key]] = label_series[key]
```

In [80]:
```
data_train
```

Out[80]:

	Image	Id
4275	70238365.jpg	w_ebf3f26
4533	75e189d4.jpg	w_715c557
1745	2e08f2ba.jpg	w_b48535f
5427	8cea266f.jpg	w_8e92baa
1452	26482485.jpg	w_b3655a6
101	0267139c.jpg	w_e156c87
319	07a58418.jpg	w_4e7fc3e
9218	f0411154.jpg	w_326e389
2742	47575ccc.jpg	w_afe953f
625	0fc63e94.jpg	w_9771603
1922	32754498.jpg	new_whale
9770	fdd5b843.jpg	new_whale
7290	bcf56bfe.jpg	new_whale
3133	515c76d0.jpg	new_whale
1254	2091af92.jpg	w_37dd956
1195	1f1122d6.jpg	w_6132293
2436	3f514e6e.jpg	w_6a16373
4217	6eb7c487.jpg	w_ff7630a
2609	442005d4.jpg	w_1e7bb93
8895	e7b13d2a.jpg	new_whale
259	06252a55.jpg	w_b688397
4429	734430c9.jpg	w_e7f8e67
334	07e96de0.jpg	w_7307089
3071	502c2aeb.jpg	w_90f5d3b
3946	677c2196.jpg	w_8114b1b
8809	e5457edf.jpg	w_cd65880
5654	928ef6f7.jpg	w_b0362e2
7514	c29996d3.jpg	w_ea2385d
8889	e79c6fa7.jpg	w_4a17405
461	0b635230.jpg	new_whale
...	...	...
2734	471cc7c1.jpg	w_fe49bc4

	Image	Id
189	04b3714e.jpg	w_37dd956
9167	ef4108d9.jpg	w_0d39a68
2747	476f249a.jpg	w_3197568
2047	3535398b.jpg	w_6e8486d
7849	cb93a0b0.jpg	w_9ea2cc3
2558	42d640db.jpg	w_fac9864
9274	f1b24b92.jpg	w_fe95ab8
8666	e16eed44.jpg	w_1e68ef5
6396	a651c7c4.jpg	w_2c55303
3385	57efc103.jpg	w_addcafa
4555	768a1cf7.jpg	w_db0ad01
1184	1ebd8ea1.jpg	w_8867074
6420	a6dc7463.jpg	w_fba3bde
5051	8389019f.jpg	w_987a36f
5311	8a3fb9df.jpg	w_bc9dc37
2433	3f419c58.jpg	w_1306632
6949	b4442482.jpg	w_c00534d
769	13359607.jpg	w_7e8b270
1685	2c8f7197.jpg	w_6b4af70
8322	d8b470f4.jpg	w_26f9f95
5578	90ab92e3.jpg	new_whale
4426	73393f00.jpg	w_0981144
466	0b7dbf66.jpg	w_97f5054
6265	a300e9ee.jpg	w_1f6e1db
5734	9490698e.jpg	w_6c899ff
5191	87055451.jpg	w_19c005a
5390	8c37aa0c.jpg	w_a254eb0
860	152fb267.jpg	w_45b90d9
7270	bc7482e2.jpg	w_02facde

7880 rows × 2 columns

In [81]:
```
train_target = [0 for _ in range(len(data_train['Id']))]
```

In [82]:
```
for i in range(len(data_train['Id'])):
    train_target[i] = Id_classes_map[data_train.iloc[i]['Id']]
```

In [83]:
```
train_target
```

```
Out[83]: [1955,
          1876,
          1287,
          3003,
          1089,
          92,
          277,
          3480,
          1725,
          521,
          7,
          7,
          7,
          7,
          165,
          934,
          1687,
          254,
          1776,
          7,
          219,
          2136,
          288,
          320,
          569,
          2363,
          869,
          2545,
          4033,
          7,
          1580,
          83,
          7,
          574,
          1099,
          8,
          837,
          90,
          311,
          120,
          463,
          1104,
          3501,
          2287,
          2651,
          7,
          3371,
          27,
          778,
          2523,
          1037,
          55,
          605,
          2662,
          7,
          1224,
          491,
```

```
7,
13,
2256,
1716,
1204,
2221,
3896,
7,
1938,
618,
1832,
979,
86,
3434,
3565,
1928,
2770,
4146,
1368,
2552,
775,
134,
521,
2293,
1036,
116,
2137,
7,
7,
7,
3574,
2745,
504,
1590,
235,
2046,
655,
3499,
828,
1164,
2236,
826,
159,
1254,
7,
3529,
2909,
516,
77,
1202,
1190,
1202,
1202,
3126,
2383,
1615,
588,
```

```
94,
958,
704,
364,
7,
2795,
4213,
427,
800,
210,
2994,
1473,
2688,
2587,
1331,
7,
2100,
3005,
2030,
3483,
1011,
1193,
1012,
7,
3353,
1700,
541,
759,
4166,
4114,
2134,
1165,
143,
866,
2248,
1677,
2112,
675,
720,
40,
1796,
449,
2727,
2767,
2232,
7,
589,
1473,
84,
1936,
3015,
984,
1967,
2005,
28,
255,
940,
```

```
3394,
2369,
3103,
551,
3910,
3277,
2881,
4165,
577,
559,
2176,
1223,
7,
1415,
7,
1130,
1954,
2147,
3975,
2742,
593,
90,
243,
1023,
7,
513,
2320,
2867,
1074,
1611,
3607,
3433,
4063,
3706,
7,
614,
506,
2197,
1553,
3265,
2986,
211,
2113,
7,
2004,
2904,
174,
7,
1378,
2765,
388,
7,
503,
588,
1268,
3698,
3710,
```

```
3064,
7,
2802,
684,
1496,
763,
279,
2483,
990,
993,
1796,
287,
813,
124,
2195,
801,
1562,
1594,
2998,
1128,
522,
7,
120,
1830,
198,
1939,
4004,
2376,
2005,
589,
181,
782,
4042,
7,
251,
2211,
1272,
1736,
290,
2785,
243,
7,
7,
344,
103,
394,
2923,
198,
7,
71,
202,
2513,
808,
53,
1714,
1439,
1393,
```

```
303,
589,
1187,
1398,
1696,
2721,
473,
1180,
1096,
618,
7,
2,
902,
39,
2765,
3800,
2833,
3581,
446,
2969,
673,
298,
73,
1004,
2350,
1201,
677,
7,
390,
675,
51,
7,
1226,
311,
3034,
3647,
673,
2727,
3750,
1653,
7,
1609,
615,
3883,
7,
1715,
1198,
2115,
2399,
1556,
3839,
175,
3773,
1565,
1477,
2419,
1442,
```

```
3079,
134,
1172,
152,
3510,
7,
4013,
7,
3255,
4170,
3235,
419,
528,
251,
1410,
7,
4190,
2569,
680,
3408,
3506,
1782,
761,
246,
58,
26,
2215,
7,
307,
2921,
2367,
1164,
512,
527,
708,
561,
2222,
412,
402,
1660,
2365,
1955,
2648,
705,
57,
2374,
132,
2015,
1732,
7,
2134,
3834,
2208,
848,
1986,
332,
666,
```

```
7,
2421,
3276,
684,
2863,
65,
7,
7,
3748,
266,
863,
1209,
747,
7,
1637,
84,
1453,
541,
1933,
388,
2707,
451,
3525,
949,
1941,
2007,
1584,
937,
4160,
218,
1535,
66,
1220,
4091,
1702,
241,
1267,
302,
1081,
1585,
79,
84,
1608,
346,
1261,
7,
336,
2635,
3660,
1555,
815,
820,
244,
2522,
4047,
178,
2486,
```

```
1425,
443,
2308,
3680,
1033,
2716,
1126,
2377,
3621,
1912,
965,
561,
1166,
3111,
3141,
7,
770,
2579,
658,
1224,
3515,
2207,
349,
3505,
7,
2190,
3804,
559,
2602,
7,
7,
218,
1607,
2325,
2741,
618,
385,
2025,
7,
445,
36,
209,
3683,
3775,
4061,
684,
1763,
352,
3645,
1,
1434,
1911,
319,
2208,
160,
2548,
1709,
```

```
2674,
2086,
531,
782,
632,
2677,
2782,
93,
1,
3538,
218,
3554,
3955,
7,
2786,
1494,
3712,
424,
7,
1880,
1025,
586,
1576,
654,
3282,
246,
7,
1443,
2602,
7,
1608,
2313,
946,
4238,
796,
171,
597,
920,
8,
275,
1114,
2456,
2222,
1916,
611,
3191,
13,
3650,
1947,
2143,
1022,
614,
635,
7,
571,
1256,
665,
```

```
787,
1,
71,
7,
613,
3300,
2433,
2902,
436,
4046,
3781,
1519,
2494,
1482,
379,
2593,
365,
1971,
210,
113,
1734,
61,
914,
43,
831,
2119,
2478,
3530,
1689,
3558,
988,
3959,
1696,
1690,
2674,
199,
1010,
36,
365,
181,
586,
1273,
2463,
2601,
720,
1218,
1907,
13,
1118,
3636,
181,
54,
2290,
2715,
573,
345,
286,
```

```
7,
787,
144,
4125,
7,
2496,
523,
3474,
1018,
3802,
1234,
1018,
496,
2396,
584,
367,
1993,
1742,
2119,
3271,
7,
13,
2455,
2698,
4066,
5,
2359,
728,
2341,
7,
412,
2235,
1575,
681,
2973,
637,
1261,
7,
2132,
243,
3827,
588,
933,
2191,
7,
1817,
4089,
2783,
371,
371,
544,
307,
996,
94,
2043,
162,
1144,
```

```
2678,
54,
2875,
178,
875,
184,
1260,
2058,
1,
3864,
167,
7,
2949,
160,
2721,
1317,
1695,
860,
1634,
2144,
896,
2065,
496,
63,
457,
939,
1634,
3845,
1358,
2491,
7,
20,
496,
1012,
2019,
419,
1576,
172,
4205,
2033,
1759,
435,
883,
3958,
10,
2570,
1547,
2972,
1716,
611,
3217,
1976,
2999,
2135,
424,
1816,
3015,
```

```
3774,
3977,
650,
188,
701,
7,
797,
1743,
7,
234,
360,
3666,
1475,
3928,
7,
2394,
912,
215,
357,
2348,
1054,
27,
622,
365,
134,
936,
1541,
1939,
4002,
4157,
3539,
3809,
283,
1078,
391,
3903,
1478,
1491,
7,
1341,
3807,
1572,
991,
1187,
3862,
2728,
1128,
2600,
3039,
1787,
7,
3860,
992,
2120,
7,
1079,
4174,
```

```
3397,
49,
2984,
3927,
783,
675,
3355,
3948,
2741,
1285,
1443,
2174,
2465,
2169,
7,
2579,
1209,
441,
3333,
218,
1490,
94,
7,
98,
4098,
266,
1169,
4122,
7,
1290,
344,
2143,
281,
7,
3965,
255,
3911,
2719,
831,
883,
248,
7,
7,
573,
3133,
406,
1714,
425,
4065,
1950,
2066,
608,
3344,
315,
2930,
1147,
2749,
```

```
4149,
68,
2219,
2220,
1543,
1342,
1437,
10,
1152,
161,
1074,
2286,
653,
3574,
2743,
1062,
564,
62,
4209,
7,
364,
1182,
2514,
2992,
847,
296,
2379,
117,
365,
1774,
7,
1177,
827,
29,
1170,
352,
1535,
1298,
4105,
1840,
407,
1015,
1231,
1898,
1304,
465,
616,
3072,
682,
2752,
3171,
113,
2141,
2170,
472,
1107,
398,
```

```
381,
3247,
640,
3520,
1933,
813,
2076,
1640,
2795,
2737,
636,
1926,
3989,
114,
7,
1,
683,
614,
3310,
7,
4076,
7,
715,
295,
2746,
2870,
7,
575,
7,
1403,
527,
1758,
1904,
34,
398,
3338,
7,
919,
95,
3538,
3314,
660,
3760,
4230,
7,
2368,
2492,
729,
1760,
1432,
545,
243,
2617,
1348,
3837,
7,
2681,
```

```
1637,
2124,
804,
2357,
7,
1508,
664,
1669,
1413,
97,
1220,
563,
2620,
2244,
1700,
264,
109,
158,
2704,
2797,
3330,
786,
905,
3344,
1143,
1465,
3245,
4052,
1146,
632,
684,
...]
```

In [84]:
```
classes_Id_map[train_target[0]]
```

Out[84]: 'w_ebf3f26'

random weighted sampler :https://discuss.pytorch.org/t/some-problems-with-weightedrandomsampler/23242 (https://discuss.pytorch.org/t/some-problems-with-weightedrandomsampler/23242)

In [85]:
```
weights = 1. / torch.tensor(classes_sample_count, dtype=torch.float)
samples_weights = weights[train_target]
print(samples_weights)
sampler = td.sampler.WeightedRandomSampler(weights = samples_weights,
                                           num_samples = len(samples_weig
hts),
                                           replacement = True)
```

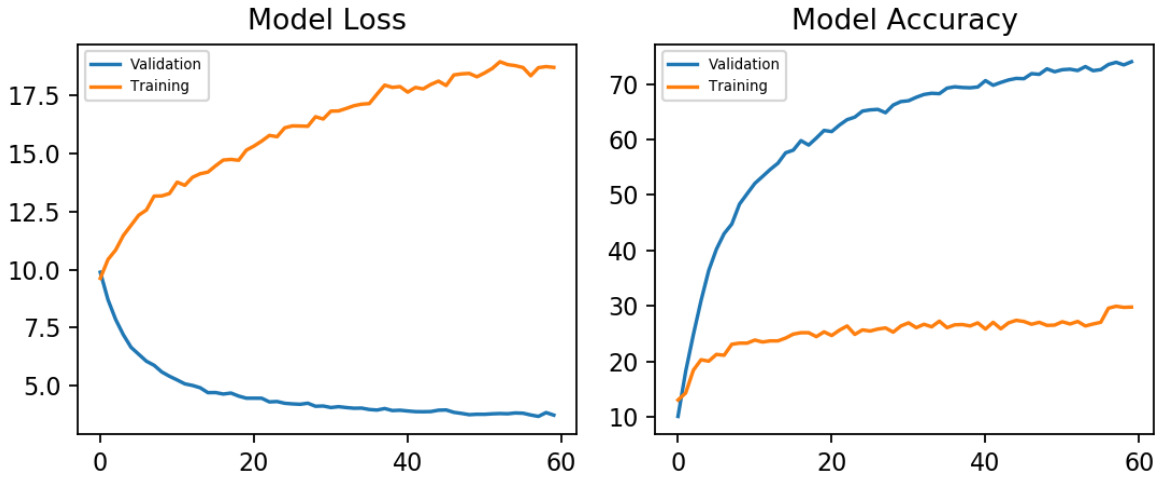
```
tensor([0.3333, 0.5000, 0.5000,  ..., 0.0909, 1.0000, 0.1250])

/opt/conda/lib/python3.6/site-packages/torch/utils/data/sampler.py:115:
UserWarning: To copy construct from a tensor, it is recommended to use
sourceTensor.clone().detach() or sourceTensor.clone().detach().requires
_grad_(True), rather than torch.tensor(sourceTensor).
  self.weights = torch.tensor(weights, dtype=torch.double)
```

In [86]:
```
train_set = AugmentData(data_dir = data_dir,
                        label_csv = data_train,
                        total_csv = train_data_label,
                        mode = 'train')
```

In [87]:
```
train_loader = td.DataLoader(train_set, batch_size=16, sampler=sampler,p
in_memory=True)
```

In [125]:
```
lr = 1e-3
net = VGG16Transfer(num_classes)
net = net.to(device)
adam = torch.optim.Adam(net.parameters(), lr=lr)
stats_manager = ClassificationStatsManager()
exp3 = nt.Experiment(net, train_set, val_set, adam, stats_manager,
                     output_dir="whaleclass_augmented", perform_validation_du
ring_training=True)
```

In [133]:
```
fig, axes = plt.subplots(ncols=2, figsize=(7, 3))
exp3.run(num_epochs=60, plot=lambda exp: plot(exp, fig=fig, axes=axes))
```




```
Start/Continue training from epoch 60
Finish training for 60 epochs
```

In [102]:
```
print(exp3.evaluate())
```

```
{'loss': 18.697418468754467, 'accuracy': 29.776422764227643}
```

Grayscale Image Approach and Using VGG16 Model

In [127]:
```
class WhaleDataset(td.Dataset):
    def __init__(self, data_dir, label_csv, total_csv,mode,image_size=(2
24, 224)):
        super(WhaleDataset, self).__init__()
        self.image_size = image_size
        self.mode = mode
        self.data = label_csv
        self.images_dir = data_dir
        self.total_data = total_csv
    def __len__(self):
        return len(self.data)
    def __repr__(self):
        return "BirdsDataset(mode={}, image_size={})". \
            format(self.mode, self.image_size)
    def __getitem__(self, idx):
        img_path = os.path.join(self.images_dir, self.data.iloc[idx]['Im
age'])

        img = Image.open(img_path).convert('L')
        stack_img = np.stack((img,)*3, axis=-1)
        img = Image.fromarray(stack_img, 'RGB')


        transform = tv.transforms.Compose([tv.transforms.Resize(self.ima
ge_size),
                                           tv.transforms.ToTensor(),
                                           tv.transforms.Normalize(mean=
[0.5,0.5,0.5],std=[0.5,0.5,0.5])
                                          # COMPLETE
                                          ])
        x = transform(img)
        d = Id_classes_map[self.data.iloc[idx]['Id']]
        return x, d

    def number_of_classes(self):
        return self.total_data['Id'].nunique()
```

In [128]:
```
def myimshow(image, ax=plt):
    image = image.to('cpu').numpy()
    image = np.moveaxis(image, [0, 1, 2], [2, 0, 1])
    image = (image + 1) / 2
    image[image < 0] = 0
    image[image > 1] = 1
    h = ax.imshow(image)
    ax.axis('off')

    return h
```

In [129]:
```
train_set = WhaleDataset(data_dir = data_dir,
                         label_csv = data_train,
                         total_csv = train_data_label,
                         mode = 'train')
print(data_dir)
```

```
train
```

In [130]:
```
x_g = train_set.__getitem__(7)
```

In [131]:
```
x_g
```

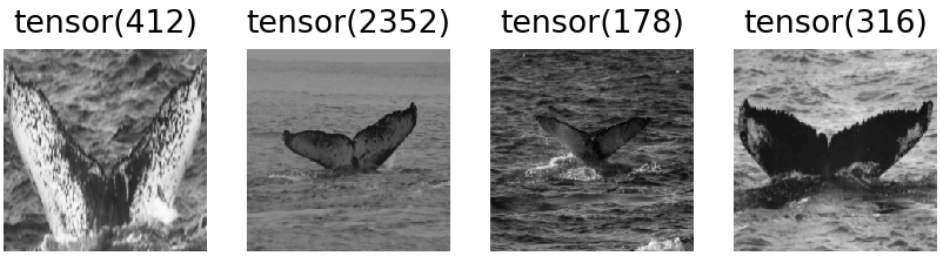
Out[131]:
```
(tensor([[[-0.2627, -0.2549, -0.2078,  ..., -0.0745, -0.1216, -0.1216],
          [-0.2863, -0.2863, -0.2784,  ..., -0.0980, -0.1216, -0.1216],
          [-0.3098, -0.3098, -0.3020,  ..., -0.1373, -0.1216, -0.1216],
          ...,
          [-0.1137, -0.1059, -0.1294,  ..., -0.1843, -0.1765, -0.1843],
          [-0.0824, -0.0824, -0.0980,  ..., -0.2314, -0.2078, -0.2157],
          [-0.0588, -0.0588, -0.1137,  ..., -0.2549, -0.2235, -0.231
4]],

         [[-0.2627, -0.2549, -0.2078,  ..., -0.0745, -0.1216, -0.1216],
          [-0.2863, -0.2863, -0.2784,  ..., -0.0980, -0.1216, -0.1216],
          [-0.3098, -0.3098, -0.3020,  ..., -0.1373, -0.1216, -0.1216],
          ...,
          [-0.1137, -0.1059, -0.1294,  ..., -0.1843, -0.1765, -0.1843],
          [-0.0824, -0.0824, -0.0980,  ..., -0.2314, -0.2078, -0.2157],
          [-0.0588, -0.0588, -0.1137,  ..., -0.2549, -0.2235, -0.231
4]],

         [[-0.2627, -0.2549, -0.2078,  ..., -0.0745, -0.1216, -0.1216],
          [-0.2863, -0.2863, -0.2784,  ..., -0.0980, -0.1216, -0.1216],
          [-0.3098, -0.3098, -0.3020,  ..., -0.1373, -0.1216, -0.1216],
          ...,
          [-0.1137, -0.1059, -0.1294,  ..., -0.1843, -0.1765, -0.1843],
          [-0.0824, -0.0824, -0.0980,  ..., -0.2314, -0.2078, -0.2157],
          [-0.0588, -0.0588, -0.1137,  ..., -0.2549, -0.2235, -0.231
4]]]),
 3480)
```

In [134]:
```
train_loader = td.DataLoader(train_set, batch_size=16, shuffle=True, pin
_memory=True)
```

In [135]:
```
val_set = WhaleDataset(data_dir = data_dir,
                       label_csv=data_val,
                       total_csv = train_data_label,
                       mode = 'val')
```

In [136]:
```
val_loader = td.DataLoader(val_set, batch_size=16, pin_memory=True)
```

In [137]:
```
num_loop = 0
fig = plt.figure()
for img, label in train_loader:
    num_loop += 1
    if (num_loop <= 4):
        plt.subplot(1, 4, num_loop)
        myimshow(img[0])
        plt.title(str(label[0]))
    else:
        break
```



In [138]:
```
class NNClassifier(nt.NeuralNetwork):
    def __init__(self):
        super(NNClassifier, self).__init__()
        self.cross_entropy = nn.CrossEntropyLoss()
    def criterion(self, y, d):
        return self.cross_entropy(y, d)
```

In [139]:
```
vgg = tv.models.vgg16_bn(pretrained=True)
```

In [140]:
```
vgg.classifier
```

Out[140]:
```
Sequential(
  (0): Linear(in_features=25088, out_features=4096, bias=True)
  (1): ReLU(inplace)
  (2): Dropout(p=0.5)
  (3): Linear(in_features=4096, out_features=4096, bias=True)
  (4): ReLU(inplace)
  (5): Dropout(p=0.5)
  (6): Linear(in_features=4096, out_features=1000, bias=True)
)
```

In [141]:
```
class VGG16Transfer(NNClassifier):
    def __init__(self, num_classes, fine_tuning=False):
        super(VGG16Transfer, self).__init__()
        vgg = tv.models.vgg16_bn(pretrained=True)
        for param in vgg.parameters():
            param.requires_grad = fine_tuning
        self.features = vgg.features
        self.classifier = vgg.classifier
        # COMPLETE
        num_ftrs = vgg.classifier[6].in_features
        self.classifier[6] = nn.Linear(num_ftrs, num_classes)
    def forward(self, x):
        # COMPLETE
        f = self.features(x)
        f = f.view(-1, 25088)
        y = self.classifier(f)
        return y
```

In [142]:
```
num_classes = train_set.number_of_classes()
```

In [143]:
```
vgg16 = VGG16Transfer(num_classes)
```

In [144]:
```
vgg16
```

```
Out[144]: VGG16Transfer(
            (cross_entropy): CrossEntropyLoss()
            (features): Sequential(
              (0): Conv2d(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1,
          1))
              (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_ru
          nning_stats=True)
              (2): ReLU(inplace)
              (3): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1,
          1))
              (4): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_ru
          nning_stats=True)
              (5): ReLU(inplace)
              (6): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil
          _mode=False)
              (7): Conv2d(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1,
          1))
              (8): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_r
          unning_stats=True)
              (9): ReLU(inplace)
              (10): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (11): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_
          running_stats=True)
              (12): ReLU(inplace)
              (13): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
          l_mode=False)
              (14): Conv2d(128, 256, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (15): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
          running_stats=True)
              (16): ReLU(inplace)
              (17): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (18): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
          running_stats=True)
              (19): ReLU(inplace)
              (20): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (21): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_
          running_stats=True)
              (22): ReLU(inplace)
              (23): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
          l_mode=False)
              (24): Conv2d(256, 512, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (25): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
          running_stats=True)
              (26): ReLU(inplace)
              (27): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (28): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
          running_stats=True)
              (29): ReLU(inplace)
              (30): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
          (1, 1))
              (31): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
```

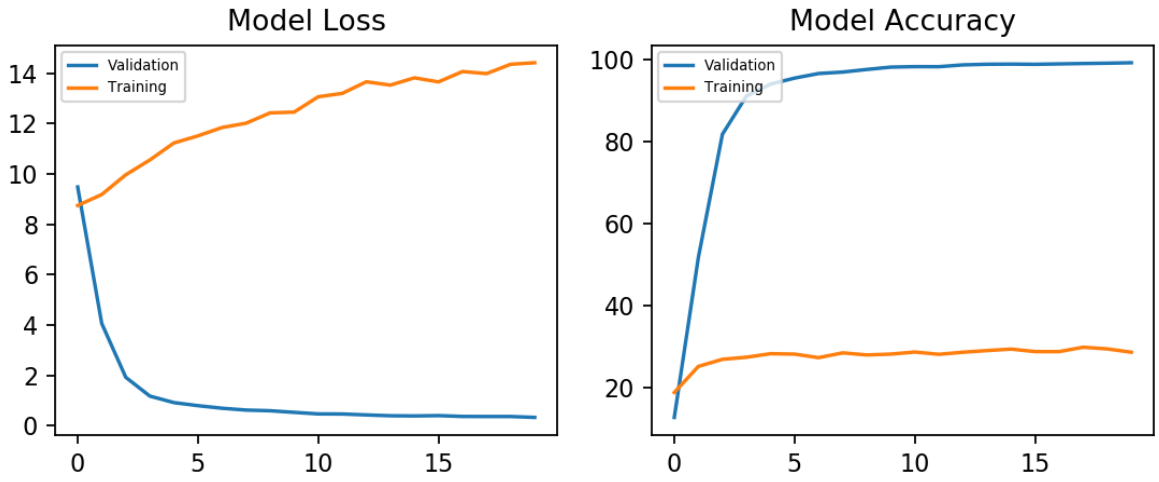
```
running_stats=True)
    (32): ReLU(inplace)
    (33): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
l_mode=False)
    (34): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (35): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (36): ReLU(inplace)
    (37): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (38): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (39): ReLU(inplace)
    (40): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=
(1, 1))
    (41): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_
running_stats=True)
    (42): ReLU(inplace)
    (43): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, cei
l_mode=False)
  )
  (classifier): Sequential(
    (0): Linear(in_features=25088, out_features=4096, bias=True)
    (1): ReLU(inplace)
    (2): Dropout(p=0.5)
    (3): Linear(in_features=4096, out_features=4096, bias=True)
    (4): ReLU(inplace)
    (5): Dropout(p=0.5)
    (6): Linear(in_features=4096, out_features=4251, bias=True)
  )
)
```

In [145]:
```
class ClassificationStatsManager(nt.StatsManager):

    def __init__(self):
        super(ClassificationStatsManager, self).__init__()
    def init(self):
        super(ClassificationStatsManager, self).init()
        self.running_accuracy = 0
    def accumulate(self, loss, x, y, d):
        super(ClassificationStatsManager, self).accumulate(loss, x, y, d
)
        topK_rpob, l = torch.topk(y, 5)
        batchSize = d.size()[0]
        count = 0
        for i in range(batchSize):
            if(d[i] in l[i]):
                count += 1
        #self.running_accuracy += torch.mean((d == l).float())
        self.running_accuracy += count / batchSize

    def summarize(self):
        loss = super(ClassificationStatsManager, self).summarize()
        accuracy = 100 * self.running_accuracy / self.number_update# COM
PLETE
        return {'loss': loss, 'accuracy': accuracy}
```

In [149]:
```
lr = 1e-3
net = VGG16Transfer(num_classes)
net = net.to(device)
adam = torch.optim.Adam(net.parameters(), lr=lr)
stats_manager = ClassificationStatsManager()
exp4 = nt.Experiment(net, train_set, val_set, adam, stats_manager,
                output_dir="whaleclass1_new2", perform_validation_during
_training=True)
```

In [150]:
```
def plot(exp, fig, axes):
    axes[0].clear()
    axes[1].clear()
    axes[0].plot([exp.history[k][0]['loss'] for k in range(exp.epoch)],
                label="training loss")
    axes[0].plot([exp.history[k][1]['loss'] for k in range(exp.epoch)],
label="evaluation loss")
    axes[0].legend(('Validation','Training'), fontsize=6, loc=0)
    axes[0].title.set_text('Model Loss')

    axes[1].plot([exp.history[k][0]['accuracy'] for k in range(exp.epoch
)],
                label="training accuracy")
    axes[1].plot([exp.history[k][1]['accuracy'] for k in range(exp.epoch
)], label="evaluation accuracy")
    # COMPLETE
    axes[1].legend(('Validation','Training'), fontsize=6, loc=2)
    axes[1].title.set_text('Model Accuracy')

    plt.tight_layout()
    fig.canvas.draw()
```

In [151]:
```
fig, axes = plt.subplots(ncols=2, figsize=(7, 3))
exp4.run(num_epochs=20, plot=lambda exp: plot(exp, fig=fig, axes=axes))
```




```
Start/Continue training from epoch 20
Finish training for 20 epochs
```

In [152]:
```
print(exp4.evaluate())
```

```
{'loss': 14.414612611134848, 'accuracy': 28.556910569105693}
```