III. Model documentation and write-up

Information included in this section may be shared publicly with challenge results. You can respond to these questions in an e-mail or as an attached file. Please number your responses.

1. Who are you (mini-bio) and what do you do professionally? If you are on a team, please complete this block for each member of the team.

   I'm an independent data scientist.

2. What motivated you to compete in this challenge?

   I recently participated in a similar competition where the task was to build a face recognition model for turtles. I hoped to apply the lessons learned here.

3. High level summary of your approach: what did you do and why?

   I started out with a simple classification approach - ImageNet pretrained *EfficientNet* noisy student model, with label smoothing loss, fastai for the training with default augmentations and one cycle lr scheduler. The training set was noisy - there were 788 unique whale IDs in the dataset, in contrast to the expected population of roughly 300 Cook Inlet belugas. After trying a few things to reduce the effect of this inconsistency, what worked best was simply dropping whale IDS with less than 4 samples. I ended up with a smaller dataset (383 classes), which surprisingly produced better results (CV and LB) than the complete dataset.

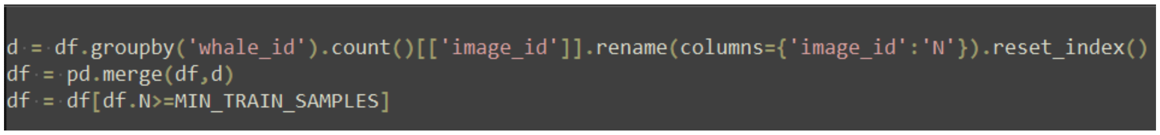
   I came across the Deep Orthogonal Local and Global (DOLG) framework, which has been successful in recent image retrieval competitions. I used models and weights from the classification step as backbone for the DOLG framework, switched the loss function to ArcFace with adaptive margins and trained with discriminative learning rates.

   My final solution was an ensemble of 15 models (3 x 5 fold cv), trained with different image sizes and on different subsets of the dataset. During inference, I applied horizontal flip augmentation, which slightly boosted the score.

4. Do you have any useful charts, graphs, or visualizations from the process?
   No

5. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.

   Filtering out classes with less than 4 samples to reduce dataset noise



```
d = df.groupby('whale_id').count()[['image_id']].rename(columns={'image_id':'N'}).reset_index()
df = pd.merge(df,d)
df = df[df.N>=MIN_TRAIN_SAMPLES]
```

6. Please provide the machine specs and time you used to run your model.
    - CPU (model): Intel Xeon @ 2.30GHz, 4vCPUs
    - GPU (model or N/A): Tesla P100
    - Memory (GB): 16
    - OS: Ubuntu 21.04
    - Train duration: ~75 hours
    - Inference duration: ~3 hours

7. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?
    No

8. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?
    No

9. How did you evaluate performance of the model other than the provided metric, if at all?
    Monitored accuracy and mean average precision (*mAP@20*) during the pre-training step

10. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

    Metric learning - Siamese networks, triplet loss with online mining, contrastive loss, listwise loss
    Swin Transformer + EfficientNet hybrid models, ther imagenet pretrained models
    k-reciprocal nearest neighbor re-ranking
    Additional augmentations - blur, noise, CLAHE, random graying, random erasing

11. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?

    I tried a lot of things based on recent image ranking/ retrieval research that should have worked but didn't or seemed not to. Most of the time, improvements in local testing didn't reflect on the leaderboard. In retrospect, I should have put more effort in building a better CV strategy.