III. Model documentation and write-up

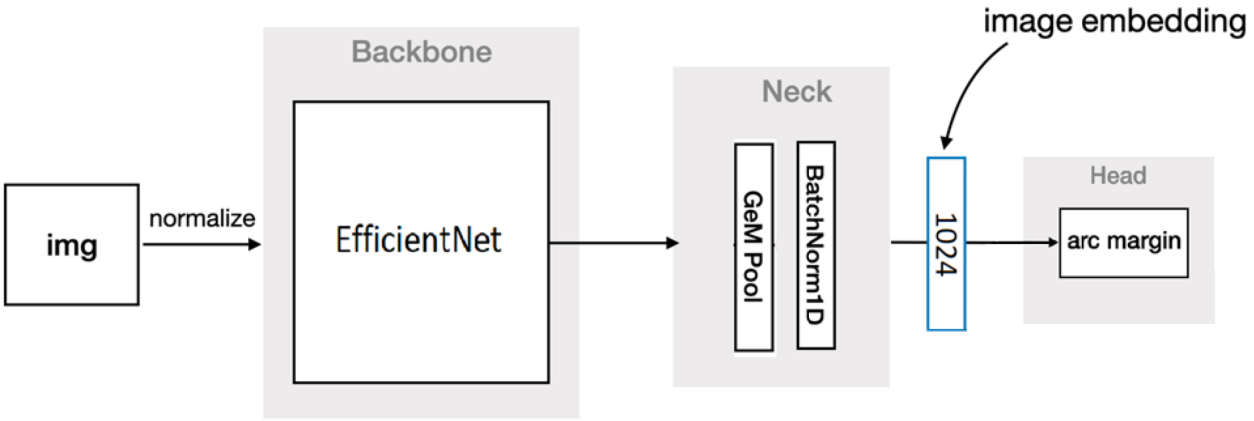
Information included in this section may be shared publicly with challenge results. You can respond to these questions in an e-mail or as an attached file. Please number your responses.

1. Who are you (mini-bio) and what do you do professionally?
   -I graduated from computer science this semester. I am interested in computer vision and NLP.

2. What motivated you to compete in this challenge?
   - Helping the marine life is a very important effort. Also watching cute beluga videos further boosted my motivation.

3. High level summary of your approach: what did you do and why?
   -Solution is based on sub-center ArcFace with Dynamic margins. As a final submission I used ensemble of 7 models which all use EfficientNet backbones. I trained models with different configurations to create diverse ensemble of models. At test time concatenated all 7 model embeddings and flipped embeddings before calculating cosine similarity.

4. Do you have any useful charts, graphs, or visualizations from the process?



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

   Sub-center ArcFace
   -Having only one center for each class causes the following issues:
   *If the intra-class sample variance is high, then it doesn't make sense to enforce compression into a single cluster in the embedding space.
   *For large and noisy datasets, the noisy/bad samples can wrongly generate a large loss value, which impairs the model training.
   Sub-Center ArcFace solves that by introducing sub-centers. The idea is that each class would have multiple class centers. The majority of samples would be contracted to dominant centers, and noisy or hard samples would be pulled to other centers.

   ArcFace with Dynamic margins

-For models to converge better in the presence of heavy imbalance, smaller classes need to have bigger margins as they are harder to learn.

6. Please provide the machine specs and time you used to run your model.
   Models trained on Google Colab using Tesla P100.
   Training time: 10 hours (all models)
   Inference time: 1 hour (all models)

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.)?

8. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?

9. How did you evaluate performance of the model other than the provided metric, if at all?
   -Checking validation score of single fold, on 5 fold cross validation.

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

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?