**5th Place Solution to Kaggle Google Universal Image Embedding Competition**

Shingo Yokoi, Noriaki Ota, Shinsuke Yamaoka
NS Solutions Corp.
{yokoi.shingo.a6q, ota.noriaki.4qp, yamaoka.shinsuke.5ke}@jp.nssol.nipponsteel.com

**Abstract**

In this paper, we present our solution, which placed 5th in the kaggle Google Universal Image Embedding Competition in 2022. We use the ViT-H visual encoder of CLIP from the openclip repository as a backbone and train a head model composed of BatchNormalization and Linear layers using ArcFace. The dataset used was a subset of products10K, GLDv2, GPR1200, and Food101. And applying TTA for part of images also improves the score. With this method, we achieve a score of 0.684 on the public and 0.688 on the private leaderboard. Our code is available. [https://github.com/iron01206/kaggle-Google-Univerals-Image-Embedding-Competition-5th-Place-Solution](https://github.com/iron01206/kaggle-Google-Univerals-Image-Embedding-Competition-5th-Place-Solution)

**1. Introduction**

Google Universal Image Embedding Competition [5] is a multi-domain image retrieval task that identifies images that have the same instance as the query image from a set of images from multiple domains (e.g., clothing, packaged goods, landmarks). Previously, the Google Landmark Challenges [3] [4] focused on instance-level recognition (ILR) of a single domain, and this competition has been extended to multiple domains. Our approach used the ViT-H [10] visual encoder of the CLIP [16] model trained on the LAION-2B [18] dataset as the backbone. CLIP is well suited for multidomain tasks such as this one because it has such a high general recognition capability that it can recognize many classes with high accuracy even in a zero-shot task. In this competition, the size of the output embedding vector is limited to 64 dimensions, so compression of the dimension is necessary. Therefore, we used ArcFace [9] to train a projection head composed of two layers, BatchNormalization and Linear. this architecture is based on MAE [13]. The dataset used is the subset of products10K [6], GLDv2 [22], GPR1200 [17], and Food101 [7]. And applying TTA also improves the score. As a result, we achieved 0.688 mAP@5 on the private leaderboard, achieving 5th place.

**2. Method**

**2.1. Architecture**

The architecture we used is shown in Fig. 1. We used the ViT-H visual encoder of CLIP model trained on the LAION-2B dataset as a backbone. This model was released during the competition and is used by most of the top teams. We tried many other models, but no one had a better performance than this model. We think the key point is that it is a large model trained on a larger dataset. The input size of the image for this model is 224x224 (RGB) and the output is a 1024-dimensional vector. The head architecture is based on MAE and is composed of two layers: BatchNormalization(affine=False) and Linear. And head receives a 1027-dimensional vector that is a concatenation of the 1024-dimensional vector that is the output of ViT-H and the original height / 224, original width / 224, and aspect ratio of the input image.

**2.2. Training & Validation**

We used ArcFace (scale=30.0, margin=0.5) for training of head. We tried various other similar methods (MSLoss [21], ContrastiveLoss [12], ProxyAnchorLoss [15], AdaFace [14], CircleLoss [19], NTXent-Loss [20], SubCenterArcFaceLoss [8]), but ArcFace performed the best. We used SAM [11]+AdamW (weight_decay=0.1) for Optimizer. By using a very large weight decay, we can reduce the overfit to the train data and improve the score. For the Learning Rate scheduler, CosineAnnealing (1e-2 to 1e-4, with 1e-4 to 1e-2 3epochs warmup) is used, the batch size is 256, and the epoch is 1000. Also, when training the head, pre-computing and storing the ViT-H outputs made it possible to train at a very fast speed. This can be trained in about 10 hours using the NVIDIA RTX3090. To check the performance of zero-shot, we prepared a test set with only data from classes not existing in train and validation. For submission, we used weights with high MAP@5 scores in the test set. However, this score has a low correlation with the LB score, and we finally considered a high LB score to be the most important.
2.3. Dataset

We used subsets of products10K, GLDv2, GPR1200, and Food101 for training. The number of classes and samples from each dataset is shown in Table 1. We used the clean version of the GLDv2 dataset. This dataset has a very large number of data, so we randomly selected 5000 classes from the classes that have more than 49 samples. All the data from the Products10K dataset were used. The GPR1200 dataset contains 200 classes of data each sampled from six different datasets. We used them all except that from the iNaturalist. The Food101 dataset was randomly removed and the maximum number of samples for each class was 30. We tried many other datasets but did not use them due to lower scores on the public leaderboard. We also tried adjusting the number of data according to the distribution of the types of objects in the test set revealed by the host, but this did not work well.

2.4. Inference tricks

We improved scores by preprocessing the input images and by using TTA (Test Time Augmentation). There are two key points to input image preprocessing: first, it is important to use the same resizing method used when training the ViT-H encoder of CLIP. This means using antialias and BICUBIC for interpolation. The second is to pad the image to a square and then resize it to avoid changing the aspect ratio. Next, when applying TTA, transforms such as flip were not effective, but such as crop or aspect ratio changes were effective. However, because we used a large encoder, we did not have enough time to apply TTA to all images. Therefore, we decided to apply TTA only to the no-square images. And we also applied TTA three times to images predicted to be apparel, packaged, or toy using the classification head trained using the 130K images dataset [1]. Finally, applying a slight center crop improved the scores. This is because important objects are often located in the center of the image, and center cropping has the same effect as increasing the resolution of the images. However, for example, in landmark images, information around objects is often important as well, so excessive center cropping can worsen the score.

### Table 1. Number of classes and samples from each datasets.

| Dataset     | Classes | Samples  |
|-------------|---------|----------|
| GLDv2       | 5000    | 249350   |
| Products10k | 9691    | 141931   |
| GPR1200 landmark | 200   | 2000     |
| GPR1200 face  | 200    | 2000     |
| GPR1200 instre | 200   | 2000     |
| GPR1200 sketch | 200   | 2000     |
| GPR1200 sop   | 200    | 2000     |
| Food101      | 101    | 3030     |

3. Findings

We used the ViT-H encoder and were able to make a competitive model by training only a head. However, we found that the encoder also needs to be finetuned to make a better model. Team cuilab.ai has shown us a great solution to this [2]. We also tried the approach of training with a smaller LR to avoid over-updating the encoder parameters, but it did not improve the score. The next time a similar competition is held, it will be a highly competitive battle.

4. Conclusion

In this paper, we presented our solution to the Google Universal Image Embedding competition. We use the ViT-H visual encoder of CLIP from the openclip repository as a backbone and train a head model composed of BatchNor-
normalization and Linear layers using ArcFace. Tuning the dataset and using some tricks at inference, we reached a final score of 0.684 on the public and 0.688 on the private leaderboard respectively.

References

[1] 130k images (512x512) - universal image embeddings. https://www.kaggle.com/datasets/rhtsingh/130k-images-512x512-universal-image-embeddings.

[2] 1st place solution of google universal image embedding. https://www.kaggle.com/competitions/google-universal-image-embedding/discussion/359316.

[3] Google landmark recognition 2021. https://www.kaggle.com/competitions/landmark-recognition-2021.

[4] Google landmark retrieval 2021. https://www.kaggle.com/competitions/landmark-retrieval-2021.

[5] Google universal image embedding. https://www.kaggle.com/competitions/google-universal-image-embedding.

[6] Yalong Bai, Yuxiang Chen, Wei Yu, Linfang Wang, and Wei Zhang. Products-10k: A large-scale product recognition dataset. CoRR, abs/2008.10545, 2020.

[7] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In European Conference on Computer Vision, 2014.

[8] Jiankang Deng, Jia Guo, Tongliang Liu, Mingming Gong, and Stefanos Zafeiriou. Sub-center arcface: Boosting face recognition by large-scale noisy web faces. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, ECCV (11), volume 12356 of Lecture Notes in Computer Science, pages 741–757. Springer, 2020.

[9] Jiankang Deng, Jia Guo, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. CoRR, abs/1801.07698, 2018.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In International Conference on Learning Representations, 2021.

[11] Pierre Forêt, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. CoRR, abs/2010.01412, 2020.

[12] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06), volume 2, pages 1735–1742, 2006.

[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. CoRR, abs/2111.06377, 2021.

[14] Minchul Kim, Anil K Jain, and Xiaoming Liu. Adaface: Quality adaptive margin for face recognition. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

[15] Sungyeon Kim, Dongwon Kim, Minsu Cho, and Suha Kwak. Proxy anchor loss for deep metric learning. CoRR, abs/2003.13911, 2020.

[16] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. CoRR, abs/2103.00020, 2021.

[17] Konstantin Schall, Kai Uwe Barthel, Nico Hezel, and Klaus Jung. GPRI200: A benchmark for general-purpose content-based image retrieval. CoRR, abs/2111.13122, 2021.

[18] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade W Gordon, Ross Wightman, Mehdi Cheriti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa R Kundrauth, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsiev. LAION-5b: An open large-scale dataset for training next generation image-text models. In Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track, 2022.

[19] Yifan Sun, Changmiao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei. Circle loss: A unified perspective of pair similarity optimization. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 6397–6406, 2020.

[20] Aaiw van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. CoRR, abs/1807.03748, 2018.

[21] Xun Wang, Xintong Han, Weilin Huang, Dengke Dong, and Matthew R. Scott. Multi-similarity loss with general pair weighting for deep metric learning. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 5017–5025, 2019.

[22] Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Premise-landmark dataset v2 - A large-scale benchmark for instance-level recognition and retrieval. CoRR, abs/2004.01804, 2020.