



Google Universal Image Embedding Challenge

EEE 511, Fall 2022, Team 9

Sumant Kulkarni, Dhanraj Bhosale,
Pratyush Pandey, Shayal Shamsu

The Challenge



Google Universal image embedding challenge is a competition hosted by Kaggle in collaboration with Google research and Google lens.

The specific challenge is to build a single universal image embedding model capable of representing objects from multiple domains at instance level.

The task is to not only determine the generic category of an object (e.g., an arch), but also the specific instance of the object ("Arc de Triomphe de l'Étoile, Paris, France")

This multi domain ILR is the key to real-world visual search applications, such as augmenting cultural exhibits in a museum, organizing photo collections, visual commerce and more.

Introduction



- Traditionally, research on image embedding learning has been conducted with a focus on per-domain models.
- Generally, papers propose generic embedding learning techniques applied to different domains separately rather than developing generic embedding models
- **Instance-Level Recognition (ILR)** is tackled by training a deep learning model with a large set of images.
- Capturing features of all object domains in a single dataset and training a model that can distinguish between them is a challenging task.
- The competition expects a model that extracts feature embedding for the images and submit the model via Kaggle Notebooks.
- Kaggle runs the model on a held-out test set, performs a k-nearest-neighbors lookup, and scores the resulting embedding quality.

Method - Process



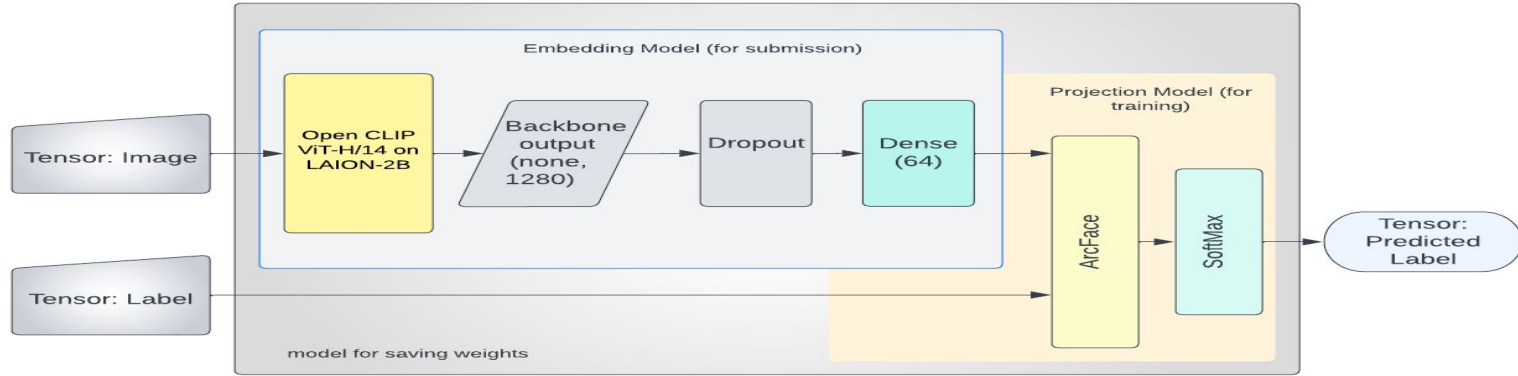
- Research various machine learning techniques for image embeddings
- Finalise reference papers that can be used for multi model image embeddings
- Review top code notebooks in the kaggle competition and select one as reference for implementation
- Identify datasets to be used for training and ascertain their availability
- Implement the deep learning network and generate the submission zip
- Submit the model to Kaggle and obtain the evaluation score
- Fine tune the model iteratively submitting multiple versions for scoring
- Build simulator and validate the performance of the model
- Document the report with results and conclusions

Method - Datasets



- Imagenet - <https://www.image-net.org/index.php> (Available for free to researchers for noncommercial use)
- Products -10K - <https://products-10k.github.io/> (Available for free for non-commercial research and educational purposes)
- Google Landmark Recognition 2021 - <https://www.kaggle.com/competitions/landmarkrecognition-2021/data> (Dataset is part of Kaggle competition in 2021)
- To reduce the dataset size, this dataset has only 50 images per class
- 90% data is used for training while 10% is used for validation.

Method - Model Architecture



Embedded model for Inference (Model Submitted for scoring): Backbone(CLIP) + Dropout + Dense(units=64) + L2 Norm

Training: Backbone(CLIP) + Dropout + Dense(units=64) + ArcFace + Softmax (classes=17691)

Method - Model Architecture



- **Embedded Inference Model:**
 - The CLIP(Contrastive Language-Image Pre-Training) is a neural network developed by OpenAi and is trained on a variety of image-text pairs.
 - OpenCLIP is trained on ViT-H/14 on LAION-2B and has the highest accuracy of 78%
 - Due to 'Zero-Shot' capabilities, CLIP models can be applied to nearly arbitrary visual classification tasks which fits perfectly in our use case.
 - Dropouts are used to prevent overfitting
 - L2 Norm Regularization is used to penalize insignificant parameters by suppressing their weights
- **Training Model:**
 - We replace L2Norm by ArcFace during training in front of the above embedded model for classification. The weights learned by Arcface are not overly dependent on the input dataset and it helps in reducing the training epochs.
 - In ArcFace, the angular margin ($\cos\theta$) is calculated by normalizing features and FC (Dense) layer weights and taking the inner product.
 - The loss is calculated by applying Softmax to $\cos\theta$. Then \arccos is applied to the $\cos\theta$ values and an angular margin of $+m$ is added only for the correct labels.

Method - Implementation





- We used Kaggle Notebook for training the model. All required computing resources are made available by Kaggle in the notebook.
- Access to datasets is also easily available in the kaggle environment.
- We built the model using Tensorflow and TPU. It takes about 3 hours to train the model in the environment, for 10 epochs.
- The embedded model with its weights is submitted in zipped format to kaggle for scoring
- We are simulating the performance of this trained model on our laptop by predicting embeddings of test images from various object types.

Results

The model trained and submitted by us for the competition scored 0.678 which is at par with the top 10 score amongst all competitors. It is trained to classify 17691 different object types.

[GitHub Repo Link](#)

Submissions			
You selected 0 of 2 submissions to be evaluated for your final leaderboard score. Since you selected less than 2 submission, Kaggle auto-selected up to 2 submissions from among your public best-scoring unselected submissions for evaluation. The evaluated submission with the best Private Score is used for your final score.			
<input checked="" type="checkbox"/> Submissions evaluated for final score			
<div>All Successful Selected Errors</div> <div>Recent</div>			
Submission and Description	Private Score	Public Score	Selected
<div> [9th place] GUIE: fintune TF(with training) - Version 1 Succeeded (after deadline) · Dhanraj Bhosale · 1mo ago · Try 2</div>	0.678	0.666	<input type="checkbox"/>
<div> Universal Image Embedding ANC Project - Version 2 Succeeded (after deadline) · Pratyush Pramod Pandey · 1mo ago · Notebook Universal Image Embedding ANC Pr...</div>	0.678	0.666	<input type="checkbox"/>

Simulation



- In addition to the score obtained from the Kaggle competition, we developed a simulation using 100 pictures in the Database
- The database has 5 categories namely - Dog, Vehicle, Food, Landmark and Clothes
- We obtain embedding for these database images and also for test image
- Using Euclidean Distance, we find the top 5 similar embeddings and the corresponding image
- In the result, an image of a bull dog was used (Database had images of german shepherds as well)

Simulation Results

Test Image and Top 5 Matches

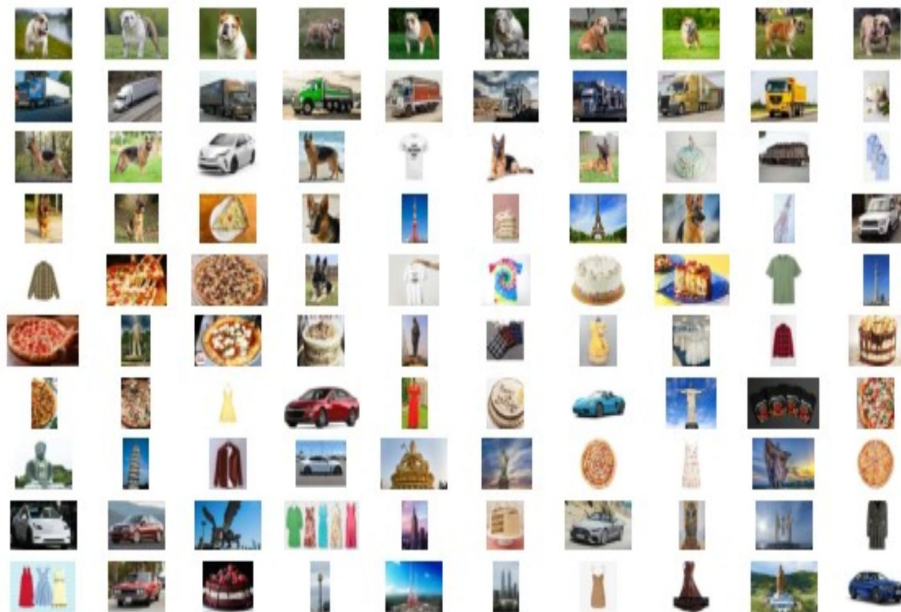


1/1 [=====] - 80s 80s/step

Top 5 Matches are:



Dataset sorted on embedding similarity



Future Enhancements



- Model to take care of corner cases like identifying difference between an actual Eiffel tower and a souvenir or replica of the Eiffel tower
- Enhancing the classification categories beyond the current 17K classes which might make the embeddings even more universal. This would require training with additional datasets
- New approaches for model design may be considered
- New approaches for model training may be used. E.g. Continuous training may be deployed in applications where images are continuously processed like Google lens

References



- Google AI Blog - Introducing the Google Universal Image Embedding Challenge, August 4, 2022, Posted by Bingyi Cao and Mário Lipovský, Software Engineer, Google Lens, <https://ai.googleblog.com/2022/08/introducing-google-universal-image.html>
- Baseline model implementation for the Kaggle universal image embedding - https://github.com/google-research/google-research/tree/master/universal_embedding_challenge
- Training data-efficient image transformers & distillation through attention - <https://arxiv.org/pdf/2012.12877.pdf>
- Transformers for image recognition at scale - <https://arxiv.org/pdf/2010.11929.pdf>
- Reference Code notebook for implementation - <https://www.kaggle.com/code/akihirok/9th-place-guie-fintune-tf-clip-with-training>
- <https://huggingface.co/laion/CLIP-ViT-H-14-laion2B-s32B-b79K>