

AI Problem Challenge – Creating Logo/Face Recognition Model

Nayeema Nonta

Problem Summary

From the problem statement, the machine learning task is to provide an approach to generate datasets for logo and face recognition. Then propose a model to extract faces/logos, and classify the brand or person.

Solution Overview

For my solution, I have decided to create 2 separate models, one to extract and recognize logos, and one to extract and recognize faces. I believe this approach enables me to easily analyze and monitor the outputs of each model and optimize it for its respective task. The overall solution is shown in Figure 1, where a data set is prepared using publicly available data, then, a baseline model is trained and evaluated. Then, the models should be iteratively improved upon the baseline metrics using other more complex models, hyper-parameter tuning, and data augmentation/normalization techniques. Once the model accuracy is 95% or higher, it can be used to make video predictions. It is important to continuously monitor the train the model. In terms of making predictions from videos, tools such as OpenCV can be used to extract and sample every 10 frames. Blurry frames can also be removed using OpenCV. Once the input is properly processed, it can be fed into the models to generate the predictions.

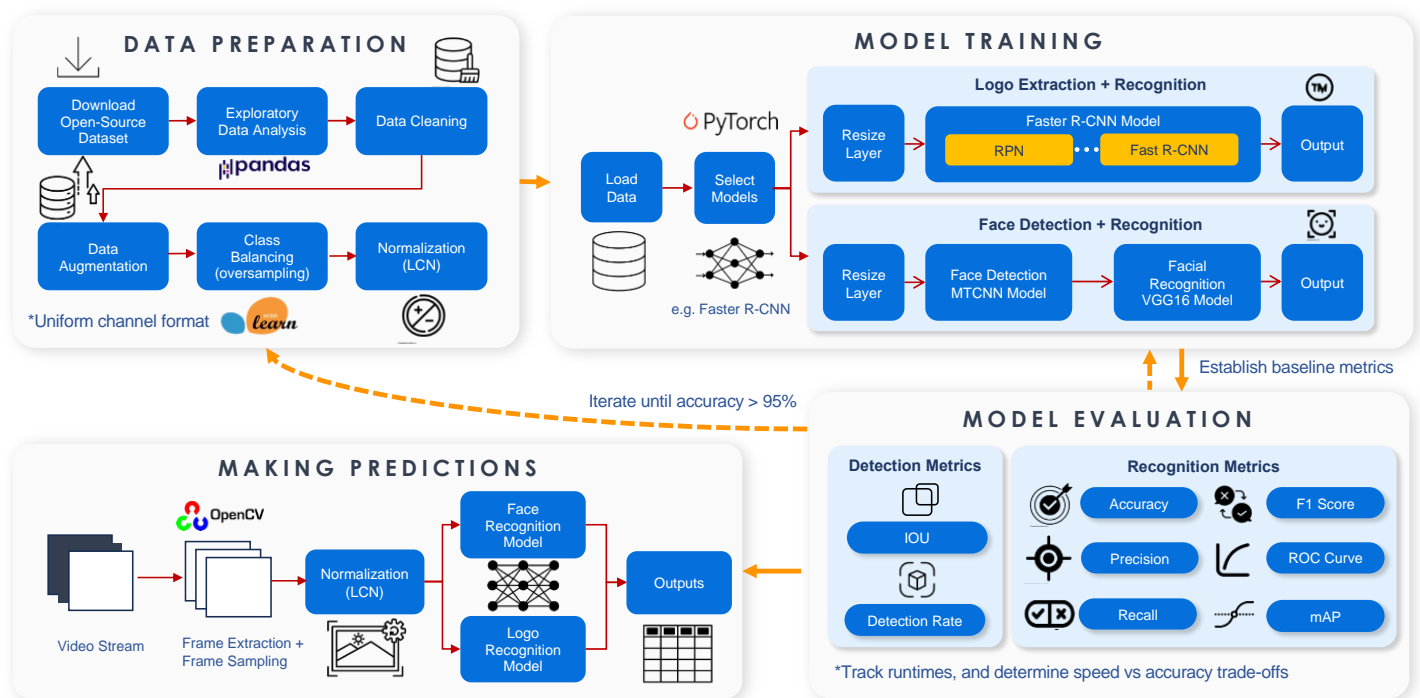


Figure 1: Solution Diagram

Constraints

1. The input will need to be pre-processed into a fixed size (e.g., 224x224) prior to input in the model
2. The input will be pre-processed to have the same number of channels as the model (e.g., RGB)
3. Additional constraints may include resources available to train models, which can be dealt with using approaches such as reducing the batch size and parallel processing

Table 1: Potential Challenges and Risks

Risk	Possible Mitigations
One brand can have multiple logos, the model would need to recognize this	Have these logos as separate classes in the dataset e.g. adidas text, adidas. Ensure the data has different logos for a brand.
The logo recognition model may not perform well with promotional/seasonal logo variations (e.g., pride logos)	Continuously update the model with new logos, try to include as very variations as possible.
Difficult find enough training data for facial recognition of famous people due to privacy concerns	Use publicly available data, and generate synthetic data using data augmentation techniques

Part 1 – Data Collection/Generation

The approaches for generating a dataset to train the models can be reduced to 3 main approaches. The first would be to manually collect and annotate data for logos and faces. The second option would be to use publicly available labeled datasets, with data augmentation (i.e., rotating, scaling, flipping, colour adjustments, etc.) to create variety in samples. In a survey of logo detection research, a major problem to building a large scale logo detection system is lack of data, and the best approach to this problem is data augmentation [1]. Popular datasets for logos and faces are listed in Table A1 and Table A2 respectively in Appendix A. Although many state that it is for research purposes, so use in commercial products may require special permissions. The third option is to use open-source generative AI to create synthetic data for training purposes. The major pros and cons of each of the approaches are listed below in Table 2.

Table 2: Pros and Cons (Including Risks) of Different Data Collection Approaches

Approach	Pros	Cons
Manual Collection + Annotation	<ul style="list-style-type: none">• Able to control the quality of data• Able to ensure bias is reduced as much as possible (all classes represented equally)	<ul style="list-style-type: none">• Time consuming
Online Datasets + Data Augmentation (Best Approach)	<ul style="list-style-type: none">• Fast and easy to access• Already labeled and organized, minimal cleaning and preparation is required• Data augmentation can create variability and increase sample sizes	<ul style="list-style-type: none">• May contain lower quality of data• Data can be biased
Generated data using GenAI	<ul style="list-style-type: none">• Quickly generate large amounts of data, for our specific needs• Can generate a variety of data	<ul style="list-style-type: none">• Outputs can be unrealistic, low quality or completely incorrect• GenAI biases can translate into generated data

While the manual approach provides the best reliability in terms of ensuring data quality, the time and labour intensive aspect of this approach is not justifiable. And while GenAI may be a feasible option to explore, it can be unreliable due to us not being able to easily identify biases in the model. In addition, the data can be unrealistic or incorrect. Therefore, *the best option is to leverage publicly available data* such as Logo-2K+ for logo recognition (which has 2341 classes and at least 50 images for each logo class [2]) and Labeled Faces in the Wild (LFW) for facial recognition. Although the data may be biased or low quality, we can do exploratory data analysis to *identify class imbalances* within the samples and *take measures to counter it* such as oversampling or data augmentation. This approach gives us some control to ensure data quality, while significantly reducing time and costs.

Part 2 – Model Selection

When it comes to the model it must be able to detect/extract the logos and faces from the videos and then make the correct classification. There can be 2 main approaches to handling this process. The first would be to separate the object detection and classification tasks. This can be done by first pre-processing the image and running it through a selective search for object recognition [3] algorithm to find objects of interest, and then apply a classification model. However, this approach is more inefficient and less accurate [4]. It would be much faster to use open-source deep learning models which combine the 2 steps into 1 model. The first step in this process to *select a model which can serve as a strong baseline* and iteratively improve upon it.

Logo Detection and Recognition Model

There are several model implementations available in open-source libraries such as torchvision, which we can train with our custom data. One candidate for logo detection, is the YOLO (You Only Look Once) model. This model is quite fast and performs well for real-time detection, however, it is less accurate when detecting objects which are small and include a lot of background noise [5]. Since we do not need real-time detection for this use case, another good approach to result in high accuracy is to use the *Faster R-CNN model which is slower, but provides higher accuracy and precision than YOLO* [4][5]. This is a unified model which combines the RPN (Region Proposal Network) which is then used by the Fast R-CNN architecture [5]. The third option is a SSD (Single Shot Detector) such as the SSD300, which instead of being a 2 stage model, performs the detection in a single shot, making it both fast and accurate [6].

Given the analysis of candidate models, I propose to use the Faster R-CNN model as a baseline model. The 2 stage approach provides higher interpretability, since we can analyze the detection and feature-extraction + classification steps separately. Starting with a relatively simpler model will help us understand the value provided by the more complex models. Therefore, I would start with this baseline, get early feedback, and set benchmarks. Then other sophisticated models like SSD300 can be attempted to improve on the baseline metrics.

Face Detection and Recognition Model

For face detection and recognition, it is much more complex to detect faces under varied lighting, angles, etc. and correctly identify the person. For this task it is better to create a pipeline to detect the faces then do facial recognition. A model like MTCNN (multi-task cascaded convolutional neural network) can be used to detect the face and be combined with popular facial recognition models such as VGG-Face or FaceNet. The MTCNN, uses a 3 phase deep neural network architecture to detect faces. It creates a bounding box around the face, while also predicting the locations of facial landmarks for facial alignment [6]. Facial alignment helps orient the images in a standard way into the facial recognition model, which is shown to increase the accuracy. For the facial recognition part, we can use an open source models from PyTorch such as VGG-Face to establish a baseline. This is a good baseline since it is a relatively simple architecture consisting of 16 layers. Once a baseline is established, more complex models such as FaceNet, which uses an inception architecture, and outputs an 128 dimension embedding [6].

Part 3 – Data Pre-Processing

Data used to train models should be cleaned to remove any noise and reduce memory. This means removal of any columns which are not to be used in training the model to ensure faster loading of data, and removing any data which may be missing or incorrect. An important point of consideration in recognition models is the aspect of class balancing. The risk of using open source data sets is that all classes may not be equally represented creating a bias. To counter this, I would use techniques like oversampling to ensure all classes are represented equally. In the case of computer vision, input pre-processing (normalization) can improve the accuracy of models [8], this is because it makes it easier to extract feature maps and improve generalization [9]. There are many normalization techniques, one that I would use is LCN (Local Contrast Normalization), which helps discriminate between adjacent pixels (Figure A1, Appendix A). For logo detection, we can also use data augmentation techniques such as rotating samples to improve the dataset. Lastly it's important to ensure the final format (dimensions) of the data is ready to be input into the respective model. Input dimensions are important in model architectures such as VGG16 (Figure A2, Appendix A) which takes in an input size of 224x224 or 32x32 since the kernel size for the convolution and pooling layers are dependent on the input size.

Part 4 – Model Training and Evaluation

To train the models I would start with a 60/40 split. Where the model would be trained on 60% of the data and test it on the remaining 40%. To evaluate the models, I would use different metrics to measure both detection accuracy (i.e., how accurate are the bounding boxes) and classification accuracy (i.e., was the logo or face correctly identified).

For bounding boxes, I would measure intersection over union (IOU). A higher IOU is better accuracy. An initial threshold of IOU = 0.75 will be set to consider a detection a true positive, then measure the overall detection rate of the model.

General classification metrics can be used such as accuracy, precision, recall, F1 Score, and mean average precision (mAP) to measure the model's performance across all classes. As well as K-Fold cross-validation to ensure the model is generalized. Also, I would make use of an ROC curve to determine the true-positive and false-positive rate trade-off. To ensure an accuracy of 95% I would do hyper-parameter and threshold tuning in the models, add data augmentation or normalization techniques to improve data quality and iteratively improve the model until it results in an accuracy of more than 95%.

Comparison with Competitive Solutions

There are commercial services provided by Amazon, Google, and Microsoft as shown in Table A3 in Appendix A which will meet our task requirements. These solutions however are for a broad range of services, and we have limited customization capabilities. While we can fine-tune their models, we are unable to make any significant changes to it. Since our model will be **task specific** (e.g. logos), it will be more optimized for our use case. This approach is also **cost effective** since it is built using freely available resources and we can easily analyze large volumes of data without paying for 3rd party services. In addition, the existing solutions provide limited flexibility and transparency. Since they use proprietary models, we do not have the capacity to fully understand the biases. Training our own model provides **greater control** to understand and counter biases, as well as scale and train the model continuously as new data becomes available, which is not possible with static commercial solutions. For these reasons, the solution outlined this document is unique.

References

- [1] S. Hou *et al.*, “Deep Learning for Logo Detection: A Survey,” *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 20, no. 3, pp. 1–23, Oct. 2023. doi:10.1145/3611309
- [2] J. Wang *et al.*, “Logo-2K+: A large-scale logo dataset for scalable logo classification,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 6194–6201, Apr. 2020. doi:10.1609/aaai.v34i04.6085
- [3] J. R. Uijlings, K. E. van de Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, Apr. 2013. doi:10.1007/s11263-013-0620-5
- [4] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017. doi:10.1109/tpami.2016.2577031
- [5] A. Sojasingarayar, “Faster R-CNN vs Yolo vs SSD- object detection algorithms,” Medium, <https://medium.com/ibm-data-ai/faster-r-cnn-vs-yolo-vs-ssd-object-detection-algorithms-18badb0e02dc> (accessed Aug. 3, 2024).
- [6] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, “Joint face detection and alignment using multitask cascaded convolutional networks,” *IEEE Signal Processing Letters*, vol. 23, no. 10, pp. 1499–1503, Oct. 2016. doi:10.1109/lsp.2016.2603342
- [7] F. Schroff, D. Kalenichenko, and J. Philbin, “FaceNet: A unified embedding for face recognition and clustering,” *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2015. doi:10.1109/cvpr.2015.7298682
- [8] K. Grm, V. Štruc, A. Artiges, M. Caron, and H. K. Ekenel, “Strengths and weaknesses of deep learning models for face recognition against image degradations,” *IET Biometrics*, vol. 7, no. 1, pp. 81–89, Oct. 2017. doi:10.1049/iet-bmt.2017.0083
- [9] Y. H. Y. Hu, “Normalizations in Neural Networks,” yeephycho, https://yeephycho.github.io/2016/08/03/normalizations_in_neural_networks/ (accessed Aug. 3, 2024).

Appendix A

Table A1: Popular Publicly Available Logo Datasets

Dataset	No. Samples	No. Classes	License	Notes	Link
QMUL-OpenLogo	27083	352	Academic research purpose only	Created by aggregating/refining 7 other datasets	https://hangsu0730.github.io/qmul-openlogo/
WebLogo-2M Dataset	2190757	194	Academic research purpose only	Labeled at image level instead of bounding box	https://weblogo2m.github.io/
Logos-32plus	12312	32	Research Purposes		http://www.ivl.disco.unimib.it/activities/logo-recognition/
BelgaLogos	10000	26	Research Purposes	Contains both ground truth and local level annotations	https://www-sop.inria.fr/members/Alexis.Joly/BelgaLogos/BelgaLogos.html
Logo-2K+	167140	2341	Publicly Available	Contains 10 root categories (e.g., food, clothes, medical)	https://github.com/Wangjing1551/Logo-2k-plus-Dataset

Table A2: Popular Publicly Available Famous People Faces Datasets

Dataset	No. Samples	No. Classes	License	Notes	Link
CelebA	202,599	10,177	Research Purposes		https://mmlab.ie.cuhk.edu.hk/projects/CelebA.html
VGGFace2-HQ	3.31 M	9131	Publicly Available	High quality version of the VGGFace2 dataset. The original dataset may have been taken down due to privacy concerns.	https://github.com/NNNNAI/VGGFace2-HQ?tab=readme-ov-file
Labeled Faces in the Wild (LFW)	13233	5749	Publicly Available	The website warns against bias in the data which may make it unsuitable for commercial use cases due to not having equal representation of all ages and ethnicities. Only 1680 people with 2 or more images.	https://vis-www.cs.umass.edu/lfw/#download

Table A3: Competitive Analysis with Commercial Products




	 Amazon Rekognition	 Google Cloud Vision	 Azure AI Vision	Building Custom Models
High Customization	Limited	Limited	Limited	Tailored solution
Low Cost	No	No	No	Uses free open-source data and models
Transparency	Proprietary models	Proprietary models	Proprietary models	Access to the data, can understand bias, can see and change underlying model architecture



Figure A1: Effects of LCN (Local Contrast Normalization) on an Image

(Source: <https://medium.com/@dibyadas/visualizing-different-normalization-techniques-84ea5cc8c378>)

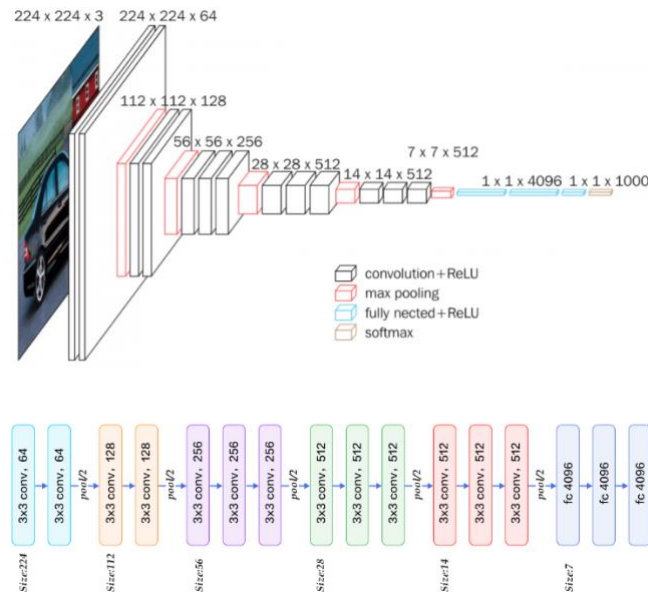


Figure A2: VGG16 Architecture

(Source: <https://www.kaggle.com/code/blurredmachine/vggnet-16-architecture-a-complete-guide>)