Project Proposal: Few-Shot Logo Recognition

Cancemi Alessia, s347156
Mincigrucci Gabriele, s358987
Oggero Paolo, s342937

Course: Machine Learning for Vision and Multimedia
Academic Year: 2025-2026

1 The Dataset Intended for Use (with Source)

1.1 dataset description

The dataset chosen for this project is the **LogoDet-3K Dataset** (available at GitHub repository). This dataset was selected because it enables the analysis of both classical Convolutional Neural Networks (CNN) and Region-Based Convolutional Neural Networks (R-CNN), as it provides bounding box annotations for every image. Such annotations allow us to evaluate different types of architectures, which is particularly interesting because it lets us compare how various networks behave and the accuracy they achieve on the same dataset.

LogoDet-3K is composed of **3,000 logo categories**, approximately **200,000 manually annotated logo objects**, and a total of **158,652 images**. The logos are categorized into nine super-categories with varying numbers of images and objects.

Table 1 presents the statistical distribution of the dataset across these super-categories.

Table 1: Statistical overview of the LogoDet-3K dataset by super-category

Root Category	**Sub-Categories**	**Images**	**Objects**
Food	932	53,350	64,276
Clothes	604	31,266	37,601
Necessities	432	24,822	30,643
Others	371	15,513	20,016
Electronic	224	9,675	12,139
Transportation	213	10,445	12,791
Leisure	111	5,685	6,573
Sports	66	3,945	5,041
Medical	47	3,945	5,185
Total	**3,000**	**158,652**	**194,261**

1.2 Dataset Split

For this project, the dataset will be split at the class (brand) level. This means that each brand appears entirely either in the base set, containing training and validation sets (which are disjoint at the brand level as well), or in the novel set (used exclusively for few-shot testing). No brand is shared between these sets, ensuring that the model is evaluated on completely unseen classes, not just unseen images. To keep the project computationally manageable, we will consider working on a subset of LogoDet-3K instead of the full dataset if training becomes too computationally expensive.

Base Brands All brands belonging to the base set are split as follows:

- **Training set:** approximately 70% of the brands
- **Validation set:** 10–20% of the brands
- **Base test set:** remaining brands

This structure allows the network to learn general logo features while retaining a portion of the brands to evaluate performance on seen classes.

Novel Brands (Few-Shot Setting) Novel brands are never shown during training. For each novel brand, we define:

- **Support set:** containing 1 or 5 images (1-shot or 5-shot), used as reference examples
- **Query set:** containing all remaining images of that brand

During evaluation, the model must retrieve all occurrences of a novel brand using only the few support examples provided, ensuring a true few-shot recognition scenario.

2 the architecture to be adopted

2.1 Baseline architecture

A classical **Convolutional Neural Network (CNN)** architecture (e.g., **ResNet-50**) will be used as a baseline model. Transfer learning will be employed by initializing the network with weights pretrained on a large dataset (e.g., **ImageNet**), which allows the model to benefit from previously learned features and accelerate convergence. The network will have to be adapted in order to produce embeddings, rather than class probabilities, since feature embeddings are required to compare similarity in a few-shot recognition task.

2.2 Advanced architecture (optional)

Since the dataset provides bounding box annotations for each logo, **if time and complexity allows us to do so**, a **Region-Based Convolutional Neural Network**

(R-CNN) (e.g., Faster R-CNN with a ResNet-50 backbone) will also be explored using transfer learning. By restricting the network's attention to the regions within the bounding boxes we hope to improve the network capability to produce significative embeddings. Such an approach is particularly advantageous in real-world scenarios where logos appear alongside complex or cluttered backgrounds, as it minimizes noise from other elements in the image while retaining essential logo features.

3 the training setup

Since we are dealing with a few-shot recognition task on images, our setup focuses on data augmentation, transfer learning and different losses to avoid overfitting and obtaining meaningful embeddings.

3.1 Data Preprocessing and Augmentation

To make the model robust against variations in lighting and orientation, we plan to apply various transformations to the input images during training. This is crucial because in the real world, logos are often distorted or occluded.

3.1.1 Resizing and Normalization

The LogoDet-3K dataset contains images with different resolutions; therefore, we will need to standardize the input size by resizing the images. Finally, we will normalize the pixel values using the standard ImageNet mean and standard deviation, which is required to correctly utilize the pre-trained weights.

3.1.2 Augmentation

To help the network learn to recognize logos regardless of orientation, perspective, or lighting, we will use multiple data augmentation techniques, including:

- Random horizontal flipping
- Random rotations
- Color jittering (brightness and contrast adjustments)

more techniques will be analyzed during training, if necessary.

3.2 Learning Strategy

Our training strategy relies on **transfer learning** to accelerate convergence and reduce the amount of required labeled data. Initially, early layers of the network (which extract basic features such as edges and textures) will be frozen and only the final layers will be trained. The exact layers to be frozen will be decided during training to allow us to compare the different results. We will also consider un-freezing part of the layers only after a set number of epochs to avoid overfitting.

We analyze two main training approaches:

3.2.1 Contrastive Loss Training

The first approach involves training the network using a **contrastive loss**, where **pairs of images** are fed into the network during each training step. Each pair is labeled as *similar* (same brand) or *dissimilar* (different brands). The network processes both images producing embeddings for each image, the **contrastive loss** is then calculated based on the distance between embeddings. For similar pairs, the loss encourages embeddings to be close together, while for dissimilar pairs, the loss pushes embeddings to be at least a predefined *margin* apart. Over time, this allows the network to learn an embedding space where images from the same brand cluster together, and images from different brands are well-separated. During inference, only the embeddings of images are computed, and distances between embeddings can be used for our few-shot task.

3.2.2 Triplet Loss Training

The second approach employs a triplet loss to train the network, which also focuses on learning a metric space where similarity corresponds to distance. In this setup, each training step uses a triplet of images: an anchor image, a positive image from the same brand, and a negative image from a different brand. The network computes embeddings for all three images and the triplet loss then encourages the distance between the anchor and positive embeddings to be smaller than the distance between the anchor and negative embeddings by at least a predefined margin. This method explicitly enforces relative similarity constraints, which helps produce a more discriminative and robust embedding space, making it particularly suitable for few-shot recognition and retrieval tasks, where new classes may appear at test time.

4 Evaluation Metrics

To evaluate the state of the training, we employ a suite of metrics monitoring distinct aspects related to classical few-shot recognition.

1. **Recall and Precision:** We explicitly monitor raw Precision and Recall to maintain a fundamental understanding of the classifier's performance. While derived metrics provide specific insights, tracking these raw values allows us to observe the immediate trade-off between the system's exactness (the absence of false positives) and its completeness (the absence of false negatives) without the abstraction of complex ranking or thresholding layers.

2. **F1 Score:** This metric serves as the harmonic mean of Precision and Recall. It provides a single, unified score that balances both concerns, ensuring that an

increase in Precision does not come at a disproportionate cost to Recall.

3. **Recall at Fixed Precision (R@P):** this is the recall score obtained when the system threshold is tuned to achieve a specific precision percentage. This metric is used specifically to address the problem of false positives.

4. **Discriminant Ratio (J):** We employ this metric for cluster validation within the generated d-dimensional embedding space. J is defined as the ratio between inter-class variance (S_B) and intra-class variance (S_W). We aim to maximize J, as a high value indicates the model has learned a "generalized" representation: producing highly similar vectors for images of the same logo (low intra-class variation) while creating distinct vectors for logos of different classes (high inter-class variance).

5. **Mean Average Precision (mAP):** This metric is adopted to track the quality of the generated rankings. In a retrieval context, it is insufficient to merely identify the top-1 match; the system must effectively rank all relevant instances (e.g., the 2nd or 3rd matches). mAP is optimal for evaluating the precision of the ranking across the entire query set. We prioritize this metric in the evaluation part for a particular advantage:

    - **Threshold Independence:** The selection of a specific similarity threshold is often arbitrary and brittle. Unlike binary classification metrics, mAP summarizes performance over the entire recall spectrum, effectively averaging precision across all possible thresholds.