technische universität
dortmund

Course Report
Machine Learning for Physicists

CINIC-10 Source Classifier: Learning to Tell CIFAR from ImageNet

Akram Aki
born in Hagen, Germany

2025

WG Rhode / Elsässer
Fakultät Physik
Technische Universität Dortmund

First reviewer:	Dr. Carsten Burgard
Second reviewer:	Dr. Cornelius Grunwald
Submission date:	31. July 2025

Abstract

The goal of this project was to build a classifier capable of determining whether an image in the CINIC-10 dataset originates from CIFAR-10 or from ImageNet. To this end, we analyzed statistical and visual differences between the two domains, including RGB mean values, per-class variance and structural image characteristics. We tested various machine learning approaches, including Convolutional Neural Networks (CNNs), Multi-Layer Perceptrons (MLPs) as well as classical models like Random Forests. In addition to training models, we explored interpretability techniques such as visualizing activation maps and analyzing learned convolutional filters. The results show that even without class labels, models can learn to distinguish the source domain of images based on subtle differences in data characteristics. All code and instructions to reproduce the results are available in the associated GitHub repository [1].

Kurzfassung

In diesem Projekt wurde das Ziel verfolgt, einen Klassifikator zu entwickeln, der innerhalb des CINIC-10-Datensatzes erkennen kann, ob ein Bild ursprünglich aus CIFAR-10 oder aus ImageNet stammt. Dafür wurden zunächst die Metadaten ausgewertet und visuelle sowie statistische Unterschiede zwischen den beiden Quellen analysiert, z.B. RGB-Mittelwerte, Varianz pro Klasse und Strukturmerkmale der Bilder. Zur Modellentwicklung wurden verschiedene Machine Learning Ansätze getestet, darunter Convolutional Neural Networks (CNNs), Multi-Layer Perceptrons (MLPs) sowie klassische Verfahren wie Random Forests. Neben der Modellierung wurden auch Ansätze zur Interpretation der Modelle verfolgt, etwa durch Visualisierung einzelner Bilder nach durchlauf des Kernel Filters oder durch die Visualisierung gelernter Filter. Das Projekt zeigt, dass auch ohne explizite Objektklassenunterscheidung Unterschiede in der Datenquelle durch maschinelles Lernen erkannt werden können. Der vollständige Code sowie eine Anleitung für Basis Einstellungen sind im zugehörigen GitHub-Repository [1] zu finden.

Contents

1 Introduction

The automated classification of visual data is one of the central tasks in modern machine learning. Over the past decade, datasets such as **CIFAR-10** and **ImageNet** have played a pivotal role in the development and evaluation of deep learning models. These datasets differ not only in resolution and content diversity, but also in how images are collected, preprocessed and labeled. While CIFAR-10 consists of small, uniformly processed images of objects with no miss labeled images, ImageNet offers a broader and more varied selection of real world scenes. The **CINIC-10 dataset** (CINIC = *CINIC Is Not ImageNet or CIFAR*) merges images from CIFAR-10 and downsampled ImageNet into a unified 10-class structure. It was primarily created to bridge the gap between the simplicity of CIFAR-10 and the complexity of ImageNet, enabling more robust benchmarking. However, this fusion may also introduce domain-specific artifacts, which could prevent certain machine learning approaches from reaching their full potential. In this project, we explore the question:

Can a ML model distinguish the original source of an image in CINIC-10?

To answer this, we analyzed the dataset at multiple levels. First, we investigated low-level statistical properties, such as per-class RGB channel means and variances, to detect structural biases between the two sources. For instance, CIFAR-10 images often appear more "stock-like", meaning well-centered, consistently lit and visually uniform. ImageNet derived images tend to be more diverse, naturalistic and less curated in appearance. Second, we trained a variety of models, ranging from shallow multi-layer perceptrons to convolutional neural networks (CNNs), to predict the image source. We also experimented with classical machine learning methods such as random forests and compared their performance to neural models. Custom data generators were implemented to efficiently stream image batches during training, allowing us to scale to tens of thousands of samples without exceeding memory limits. In addition to evaluating model performance, we investigated interpretability. We visualized feature maps and learned convolutional kernels to better understand how the models identify domain specific features and whether these features reflect meaningful differences between CIFAR-10 and ImageNet samples. This report presents the dataset and the formulation of the source classification task, outlines the modeling approaches and discusses our findings with a focus on accuracy and what the models may have learned about image domain differences.
While this report assumes basic familiarity with machine learning, no deep technical expertise is required to follow the main arguments. Readers unfamiliar with core concepts such as neural networks, convolutional layers or decision trees may refer to the external resources listed in Appendix A. These sources provide brief and accessible introductions as well as hands on learning experiences to the most important topics mentioned throughout the report.

2 Background and Related Work

Image classification is a foundational task in computer vision and has been extensively studied through standardized benchmark datasets such as **CIFAR-10** [2] and **ImageNet** [3]. CIFAR-10 consists of low-resolution (32x32) images across 10 object categories, designed for rapid prototyping of deep learning models. In contrast, ImageNet offers millions of high-resolution, hierarchically organized images, capturing a wide variety of real-world content.

The **CINIC-10** dataset [4] was introduced to bridge the gap between CIFAR-10 and ImageNet in both scale and diversity. It consists of 270,000 32x32 RGB images equally distributed across the same 10 classes as CIFAR-10. 60,000 datapoints originate from CIFAR-10 and the other 210,000 comprises downsampled ImageNet images mapped to equivalent CIFAR-10 categories. While CINIC-10 was primarily designed for benchmarking and training stability evaluation, the inclusion of two distinct data sources introduces an implicit domain shift within each class.

Domain shift, the phenomenon where data distributions differ between training and testing environments, has been extensively studied in the contexts of domain adaptation and generalization [5]. Prior work has shown that even subtle differences in image statistics, such as background texture, object positioning or lighting, can significantly affect model performance and generalizability [6, 7]. These differences are often referred to as dataset bias and models may learn to exploit such artifacts rather than focusing on semantically meaningful content.

The presence of label noise is another important consideration when working with large-scale datasets. Northcutt et al. [8] identified that datasets like ImageNet may contain a non-trivial amount of mislabeled images, potentially impacting training dynamics and evaluation reliability.

Our work builds on these observations by explicitly formulating a *source classification* task within CINIC-10. Instead of predicting object classes, we focus on whether a model can identify whether an image originated from CIFAR-10 or ImageNet. To our knowledge, this direction, treating source domain as a predictive label, has received relatively little attention. The only closely related work we are aware of is the "Guess the Dataset" experiment proposed by Torralba and Efros in their paper *Unbiased Look at Dataset Bias* [6], where human participants were asked and ML Models trained to classify the dataset origin of an image. Their work emphasizes how visual characteristics unrelated to semantic content, such as lighting, resolution and framing, can allow both humans and models to identify dataset membership, highlighting the presence of strong, exploitable dataset specific artifacts.

3 Dataset Overview

The CINIC-10 dataset combines images from CIFAR-10 and downsampled ImageNet organized into the three standard splits of `train`, `valid`, and `test`. Each split contains an equal number of 32×32 color images from the same ten object categories *airplane*, *automobile*, *bird*, *cat*, *deer*, *dog*, *frog*, *horse*, *ship* and *truck*. Each class in each split consists of 2000 CIFAR-10 images and 7000 ImageNet images, resulting in a total of 9000 images per class per split.

3.1 Preprocessing

To enable our domain classification task we first annotated each image with a `source` label. This label was inferred directly from the image filename. CIFAR-10 images have filenames starting with `cifar-10`, while ImageNet-derived images begin with `n` or a numeric identifier. This naming convention was inferred from the README file of the CINIC-10 GitHub repository [9]. These labels were added to a metadata table stored as a CSV file. Each row in the resulting metadata file contains:

- `split`: one of `train`, `valid` or `test`
- `category`: the object class label (e.g., *cat*, *truck*)
- `filename` and `full_path`: the image filename and relative path
- `source`: either `CIFAR-10` or `ImageNet`

For training, all images were normalized to the $[0, 1]$ range by dividing pixel values by 255. We also implemented a custom data generator class to load batches of images on demand during training, reducing memory usage and supporting efficient training on large datasets. This generator reads image paths and labels from the CSV files, loads them in batches, applies preprocessing, and yields image-label pairs to the model.
This setup allows the training of both deep learning and classical machine learning models on the exact same data structure, ensuring fair comparisons and reproducible results.

3.2 Examples and Visualizations

To illustrate the characteristics and distribution of the CINIC-10 dataset, we include a set of visualizations and image examples.
Figure 1 shows the distribution of images per class and source in the CINIC-10 dataset. Each class in each split contains exactly 9000 images, but the internal composition of those classes is imbalanced with 7000 images coming from ImageNet and 2000 from CIFAR-10. This imbalance could influence model learning and needs to be accounted for during evaluation.

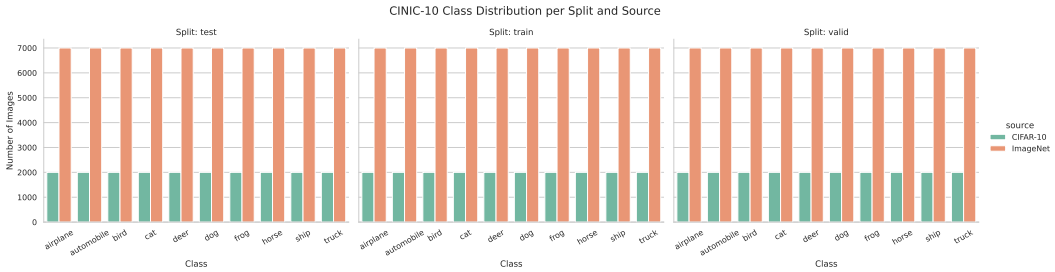


Figure 1: Distribution of classes and sources in the CINIC-10 dataset. Each class contains 9,000 images per split, but within each class, CIFAR-10 contributes 2,000 images and ImageNet 7,000.

In Figure 2, we compare the mean and variance of RGB channel values per class, separated by source. While the differences appear small at first glance, they are systematic and consistent across categories. This suggests the presence of subtle domain-specific color or lighting distributions, which models might exploit when distinguishing between sources.

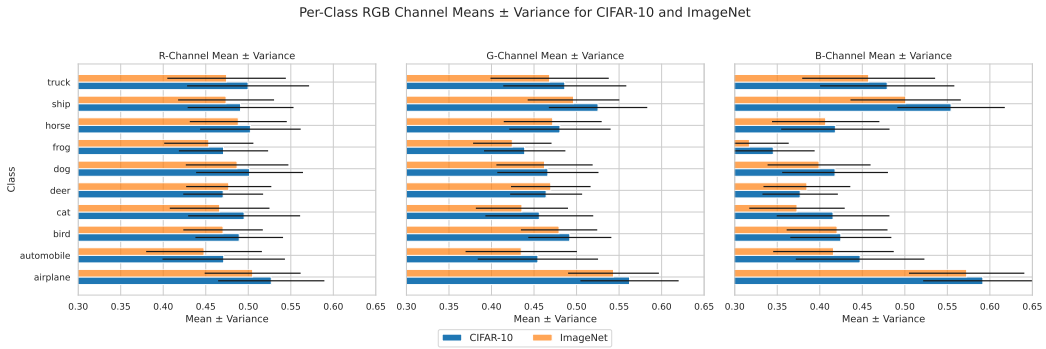


Figure 2: Mean and variance of RGB channels for each class, separated by source. Small but consistent differences between CIFAR-10 and ImageNet images are observable.

To complement these quantitative statistics, we show concrete examples from the dataset. Figure 3 presents a comparison of the *automobile* class from CIFAR-10 and ImageNet. CIFAR-10 images are visually uniform, typically centered and consistently lit. Imagenet appears to be mostly the same even though I can remember a remark at University that Cifar looks more ”stockfootagy”.
To complement the quantitative results, we include visual examples from the dataset. Figure 3 compares samples from the *automobile* class in CIFAR-10 and ImageNet. While both sets feature centered and well-lit vehicles, CIFAR-10 images often appear more uniform and curated. This impression, sometimes likened to ”stock footage,” may partly result from prolonged exposure to similar, looking images during analysis.

Figure 4 shows examples from the *dog* class. Again, CIFAR-10 and ImageNet images appear to look simliar. Interestingly, the final image in the ImageNet row appears to be a fox, which highlights the presence of mislabeled or ambiguous samples in the dataset, as previously discussed by Northcutt et al. [8].

(a) CIFAR-10: *automobile* class

(b) ImageNet: *automobile* class

Figure 3: Example images from the *automobile* class in both CIFAR-10 and ImageNet.

(a) CIFAR-10: *dog* class

(b) ImageNet: *dog* class

Figure 4: Example images from the *dog* class in CIFAR-10 and ImageNet. Notably, the last ImageNet example appears to depict a fox, illustrating the presence of mislabeled data in the dataset as discussed in [8].

These visual and statistical comparisons highlight not only the domain differences between CIFAR-10 and ImageNet but also the types of cues a model might use to differentiate between them.

4 Methodology

This section describes the systematic approach we used to investigate whether a machine learning model can distinguish between CIFAR-10 and ImageNet images within the CINIC-10 dataset. Our methodology spans data preparation, model selection, training procedures and evaluation metrics.

4.1 Problem Formulation

We define the task as a binary classification problem: Given a 32×32 RGB image from the CINIC-10 dataset, can a model predict whether it originates from the CIFAR-10 or ImageNet domain, regardless of its class label? Each image is labeled with its source domain (0 for CIFAR-10, 1 for ImageNet), which serves as the prediction target.

4.2 Data Preparation

The details of the CINIC-10 dataset structure, class distribution and preprocessing steps are summarized in the section 3 (see Figures 1 and 3). All data handling steps were based on the structure described there.

4.3 Exploratory Statistical Analysis

Before training models, we conducted an exploratory analysis of low level image statistics which is explained in subsection 3.2 (See Figure 2 for visualization).

4.4 Model Architectures

To evaluate the learnability of the source domain in CINIC-10, we trained a series of models with increasing complexity. We started with naive baselines, explored classical machine learning methods and ended with deep learning architectures. To establish meaningful lower bounds, we implemented two simple baseline models and trained a Random Forest classifier using flattened image vectors which served as a non-deep-learning benchmark. As a neural baseline, we trained a fully connected model without convolutional layers. We developed a minimal convolutional neural network using Keras to better capture spatial structure. We visualized its learned kernels and activation maps to understand the spatial features it relied on. Subsequently, a slightly more complex model was trained, with hyperparameter optimization based on validation accuracy, to obtain a more realistic yet still lightweight architecture. As a more advanced deep learning model, we trained a ResNet-18 architecture to evaluate how well a deep residual network can distinguish the domain origin. This model serves as an upper bound in terms of complexity and performance. ResNet-18 hyperparameter optimization was omitted due to the model's already sufficient accuracy and the associated time complexity.

4.5 Training Procedure

- The baseline models (Random Guessing, Majority Guessing, Random Forest) were trained using the combined `train` and `valid` splits.
- The rest of the models were trained on the `train` split and validated on the `valid` split.
- The final evaluation was performed on the separate `test` set for all models.
- Neural networks used the `Adam` optimizer with binary cross-entropy loss.

4.6 Evaluation Metrics

To evaluate model performance, we used the following metrics

- **`classification_report` from `scikit-learn`**: This provides precision, recall, F1-score and accuracy for each class [10].
- **Confusion Matrix**: Used to visualize the classification performance and distinguish between CIFAR-10 and ImageNet predictions. We employed the `heatmap` function from the `Seaborn` library for better readability and presentation [11].

5 Models and Results

This section presents a summary of each model's performance and includes key visualizations to illustrate their predictive capabilities. A decision threshold of 0.5 was used for all models producing continuous predictions — i.e., predictions were classified as class 1 if the output was greater than 0.5 and as class 0 otherwise.

5.1 Baselines: Random and Majority Guessing

As a baseline, we implemented two trivial classifiers:

- **Random Guessing**: Each image was randomly labeled as either `CIFAR-10` or `ImageNet` with 50% probability. Over 100 runs, this yielded a mean accuracy of approximately **50%**.
- **Majority Guessing**: Always predicting the dominant class (`ImageNet`) led to a test accuracy of **78%**.

These baselines confirm that a high accuracy can be misleading in imbalanced datasets, emphasizing the need for more informative metrics.

5.2 Random Forest Classifier

Our non-deep learning benchmark was built using scikit-learn's `RandomForestClassifier`. We tested all combinations of the parameters `n_estimators` (100, 150), `max_depth` (None, 10, 20), `min_samples_split` (2, 5, 10), `max_features` (`sqrt`, `log2`) and `criterion` (`gini`, `entropy`, `log_loss`). The best performing model achieved an accuracy of **78.53%**, which is close to majority guessing accuracy. It predicted `ImageNet` in 89,188 out of 90,000 cases, thus strongly favouring the `ImageNet` domain.

5.3 Fully Connected Neural Network (MLP)

The used MLP model used the architecture

- Flatten → Dense(512, ReLU) → BatchNorm → Dropout(0.3)
- Dense(256, ReLU) → BatchNorm → Dropout(0.3)
- Dense(1, Sigmoid)

and reached a test accuracy of **78%**, predicting `ImageNet` for 88,828 out of 90,000 test images. This again aligns with the majority baseline, showing that without spatial features, the model only learns the class imbalance.

5.4 Minimal CNN for Visualization

For interpretability, we trained a minimal convolutional network designed to extract and visualize learned kernels and feature maps:

- Conv2D(2 filters)(3x3 Kernel) → MaxPooling2D → Flatten → Dense(1, Sigmoid)

Despite its simplicity, this model achieved a **96%** test accuracy, with F1-scores of **0.91** for `CIFAR-10` and **0.97** for `ImageNet`. However, due to its limited use in drawing insights, we include its kernel and filtered image visualizations only in the Appendix B.

5.5 CNN with Hyperparameter Optimization

We then developed a moderately complex (compared to the model before) CNN and optimized its hyperparameters using the optimization framework `Keras Tuner` [12]. The search space included Conv layer filter count and kernel size, use of a second Conv layer, Dropout rate, Dense layer units and the Learning rate.
Out of many configurations, the best CNN achieved the following on the test set

- **Accuracy**: 98.19%
- **F1-score (CIFAR-10)**: 0.96
- **F1-score (ImageNet)**: 0.99

and the training performance on the training and validation set over multiple epochs is shown in Figure 5.

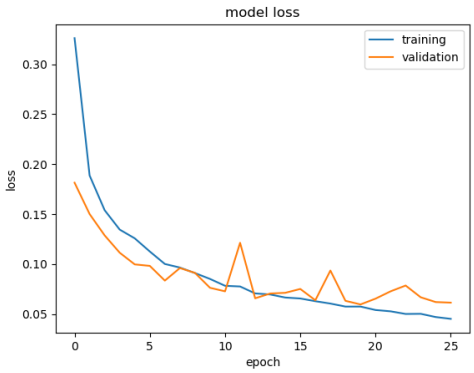


Figure 5: Training and validation loss of the CNN with Parameter ID P00.

To better understand how different hyperparameter configurations influenced the validation accuracy of our CNN model, Figure 6 visualizes the validation accuracy across tested configurations. A sample of the configuration details is listed in Table 1 in Appendix C.

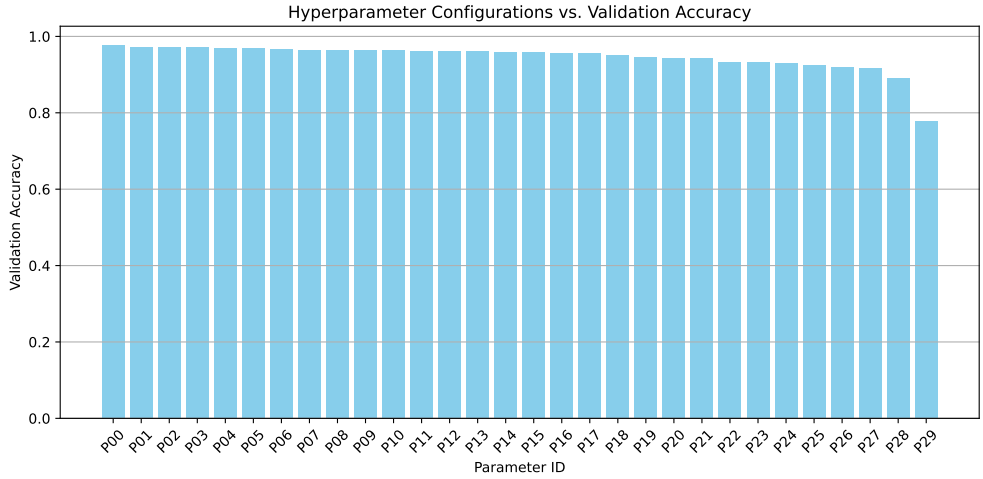


Figure 6: Validation accuracy for selected hyperparameter configurations tested with Keras Tuner. Each configuration ID corresponds to one model setup.

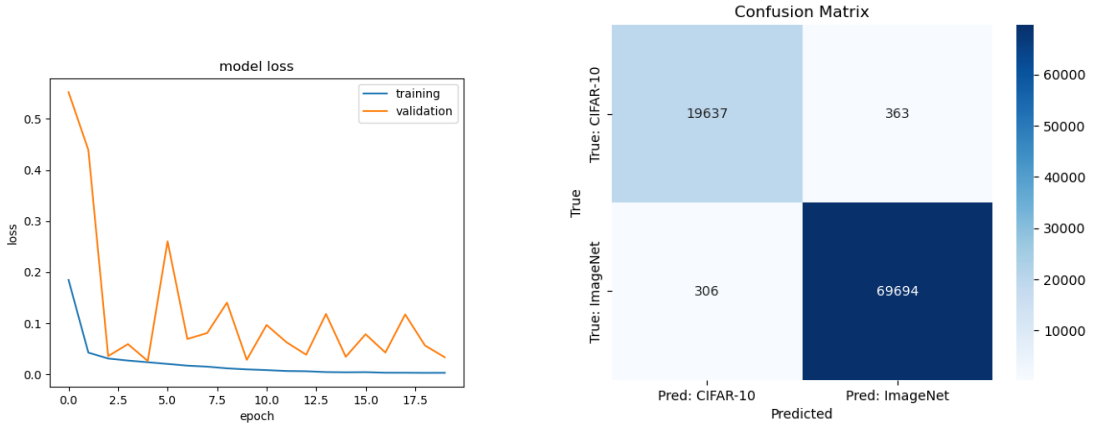
This model demonstrated that with enough capacity and tuning, even modest CNNs can learn highly predictive domain-specific features.

5.6 ResNet-18

To evaluate how well a deeper architecture performs, we trained a **ResNet-18** model [13] on the same binary classification task. Without any hyperparameter tuning, the model reached:

- **Accuracy**: 99.26%
- **F1-score (CIFAR-10)**: 0.98
- **F1-score (ImageNet)**: 1.00

Figure 7a illustrates the loss and validation performance over the training epochs, suggesting the models capabilities being bigger then needed. To further analyze performance, we include Figure 7b, which presents the confusion matrix for test predictions. The nearly perfect separation between CIFAR-10 and ImageNet images supports the conclusion that this deep model can exploit source-specific characteristics with high reliability.



(a) Loss and validation accuracy of ResNet-18 during training.

(b) Confusion matrix of ResNet-18 predictions on the test data.

Figure 7: Training behavior and evaluation of ResNet-18 model.

These results confirm that deep architectures like ResNet-18 can achieve near perfect source discrimination, setting a strong upper performance bound for this task.

6 Discussion

Our experiments show that the CINIC-10 dataset contains strong source identifiable signals. Even simple models could reliably tell apart CIFAR-10 and ImageNet images. The Random Forest classifier and Dense Neural Network, trained on flattened pixels, performed close to a majority baseline and heavily favored predicting ImageNet. This suggests that the differences between the two domains rely on spatial structures rather than just basic statistics.

More complex models, such as CNNs and ResNet-18, were able to separate the domains almost perfectly. The ResNet-18 model reached 99.26% accuracy, highlighting how easily deep networks can pick up on structural cues tied to the image source. This confirms the presence of domain shift in CINIC-10 and raises concerns for tasks where domain independence is important.

To explore how scaling affects model performance, we tried upscaling and downscaling the CIFAR-10 and ImageNet images using the same method described in the CINIC-10 repository [9]. We tested multiple scenarios: rescaling only one of the domains, both or none. Training on mixed scaled and non-scaled data was still effective, though convergence slowed for small CNNs. Surprisingly for models trained on the normal dataset and up- and down- scaling only the CIFAR-10 test images slightly improved performance, while rescaling ImageNet images reduced accuracy. We had expected the opposite effect. This unexpected result suggests that interpolation artifacts might have made CIFAR-10 images more easily identifiable. Further testing is needed to understand this better and was not included in the main part of this report due to insufficient testing.

We also visualized filters from a minimal CNN model that achieved 96% accuracy. However, the visualizations offered limited insight, suggesting that the model relied on subtle differences not easily interpreted. These are included in Appendix B.

Even though CIFAR-10 and ImageNet images appear visually similar, models clearly learn to distinguish them. This challenges the assumption that combining datasets with matching resolution and labels removes domain specific cues. It underlines the importance of domain adaptation techniques and careful dataset design for fair and reliable model training.

7 Conclusion and Future Work

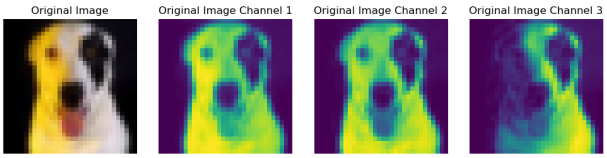
This work explored how easily the source domain of images in CINIC-10 can be learned. Despite having the same resolution and class labels, models, ranging from simple baselines to a ResNet-18, were able to distinguish CIFAR-10 and ImageNet samples with high accuracy. This shows that CINIC-10 preserves domain specific signals, which can mislead performance assessments in tasks that assume domain invariance.
To build on this, future work could involve testing whether humans can also tell the sources apart or repeating the experiments with other combined datasets like CIFAR-10 and STL-10. Another interesting direction would be to experiment with asymmetric scaling strategies, such as upscaling CIFAR-10 and only lightly downscaling ImageNet, to reduce domain cues. Techniques like adversarial domain adaptation or feature alignment could help models focus on class relevant features instead. Finally, interpreting model behavior using tools like saliency maps or SHAP values could offer deeper insights into what cues are being used for source prediction.
In short, this study underscores the importance of evaluating dataset construction and model behavior, especially when generalization, fairness or robustness matter.

A Further Reading

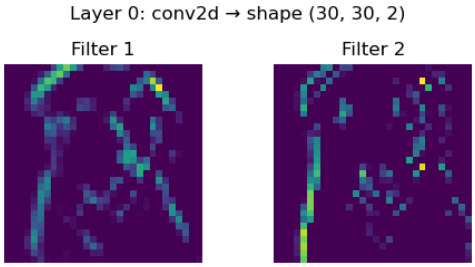
For readers seeking a gentle introduction to the core concepts of machine learning referenced in this report, we recommend the following resources:

- **Neural Networks and Deep Learning (Michael Nielsen)**
  An excellent interactive online book introducing neural networks from scratch.
  `http://neuralnetworksanddeeplearning.com/`
- **Google's Machine Learning Crash Course**
  A practical introduction to ML with TensorFlow, including video lectures and exercises.
  `https://developers.google.com/machine-learning/crash-course`
- **3Blue1Brown: Neural Networks Series (YouTube)**
  Visually intuitive explanations of neural networks and backpropagation.
  `https://shorturl.at/oQq8L`
- **Scikit-learn Documentation: Decision Trees and Random Forests**
  Concise explanations of the implementation method for random Forests used in this Report.
  `https://scikit-learn.org/stable/modules/ensemble.html`
- **Distill.pub: Feature Visualization**
  For understanding what deep neural networks "see."
  `https://distill.pub/2017/feature-visualization/`

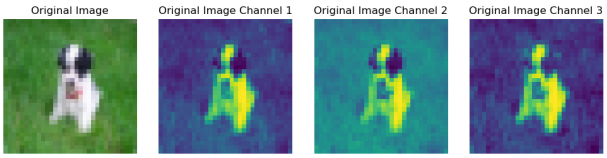
B Filter and Activation Visualizations — Minimal CNN



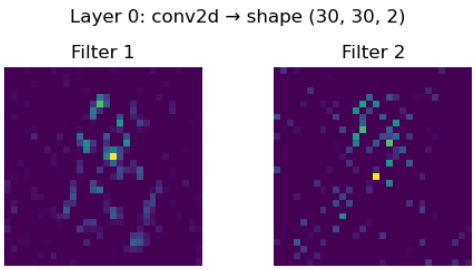
(a) CIFAR-10 image and its RGB channels with R = 1, G = 2 and B = 3.



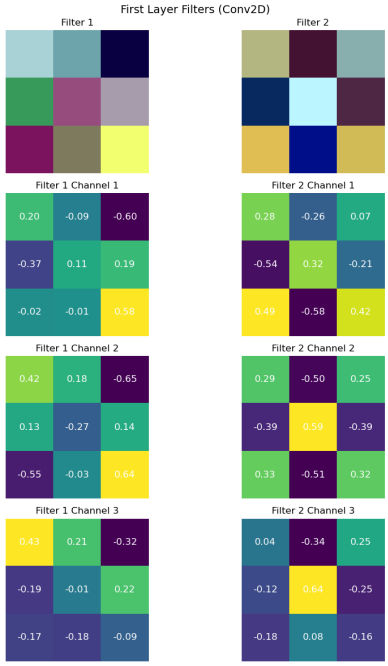
(b) Feature maps after first Conv2D layer with 2 Filters (CIFAR-10).



(c) ImageNet image and its RGB channels with R = 1, G = 2 and B = 3.



(d) Feature maps after first Conv2D layer with 2 Filters (ImageNet).



(e) First Conv2D layer filters of the Minimal Model mentioned in subsection 5.4 (3 channels, 3 × 3) where Channel 1, 2 and 3 correspond to R, G and B in that order.

Figure 8: Overview of the input images, intermediate feature maps after the first convolutional layer and learned kernels of the minimal CNN model.

C Hyperparameter Configurations

Table 1: All 30 selected hyperparameter configurations explored during model tuning with Keras Tuner. Each configuration specifies convolutional filter sizes and kernel dimensions of the first convolution layer, dropout rates (DR), dense layer size (DU = Dense Units), learning rate and whether a second convolutional layer was used and its corresponding filter number and kernel size. These settings were evaluated to optimize validation accuracy.

ID	Filt. 1	Kern. 1	Conv 2	Filt. 2	Kern. 2	DR	DU	LR
P00	40	3	True	40	3	0.2	128	0.00034
P01	16	3	True	40	3	0.2	48	0.00023
P02	56	3	False			0.2	96	0.00060
P03	16	3	True	32	3	0.1	64	0.00051
P04	40	3	True	40	3	0.2	128	0.00034
P05	32	5	True	48	5	0.3	96	0.00142
P06	56	3	False			0.2	96	0.00060
P07	8	5	True	8	3	0.4	48	0.00018
P08	40	3	False			0.3	16	0.00156
P09	16	3	True	16	5	0.5	80	0.00074
P10	40	3	False			0.5	80	0.00398
P11	40	3	False			0.5	80	0.00398
P12	40	3	False			0.3	16	0.00156
P13	40	3	False			0.5	32	0.00126
P14	56	5	False			0.3	32	0.00059
P15	64	5	False			0.2	112	0.00012
P16	32	3	False			0.1	48	0.00037
P17	16	5	False			0.4	112	0.00068
P18	40	3	False			0.3	16	0.00156
P19	56	3	False			0.2	96	0.00060
P20	16	3	True	16	5	0.5	80	0.00074
P21	40	3	False			0.5	32	0.00126
P22	40	5	False			0.2	80	0.00140
P23	24	5	True	8	3	0.3	112	0.00023
P24	64	3	False			0.1	80	0.00014
P25	48	3	False			0.2	16	0.00061
P26	48	3	False			0.1	32	0.00021
P27	24	5	False			0.5	64	0.00299
P28	32	3	True	8	5	0.5	112	0.00483
P29	24	5	False			0.3	96	0.00614

References

[1] Akram Aki. *Source Domain Classification in CINIC-10*. https://github.com/AkramAki/cinic10-source-classifier. Accessed: 2025-07-30. 2025.

[2] Alex Krizhevsky. "Learning Multiple Layers of Features from Tiny Images." In: (2009). Technical report. URL: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

[3] Jia Deng et al. "ImageNet: A large-scale hierarchical image database." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2009. URL: https://ieeexplore.ieee.org/document/5206848.

[4] Luke Darlow et al. *CINIC-10 Is Not ImageNet or CIFAR-10*. 2018. URL: https://datashare.ed.ac.uk/handle/10283/3192.

[5] Mingsheng Wang and Weihong Deng. "Deep domain adaptation: A survey." In: *Neurocomputing* 312 (2018), pp. 135–153. URL: https://doi.org/10.1016/j.neucom.2018.05.083.

[6] Antonio Torralba and Alexei A Efros. "Unbiased Look at Dataset Bias." In: *CVPR* (2011). URL: https://people.csail.mit.edu/torralba/publications/datasets_cvpr11.pdf.

[7] Benjamin Recht et al. "Do ImageNet Classifiers Generalize to ImageNet?" In: *International Conference on Machine Learning (ICML)*. 2019. URL: https://arxiv.org/abs/1902.10811.

[8] Curtis G Northcutt, Anish Athalye, and Jonas Mueller. "Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks." In: (2021). URL: https://arxiv.org/abs/2103.14749.

[9] BayesWatch. *CINIC-10: A drop-in replacement for CIFAR-10 (GitHub Repository)*. https://github.com/BayesWatch/cinic-10. Accessed: 2025-07-29. 2018.

[10] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python." In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[11] Michael L. Waskom. "seaborn: statistical data visualization." In: *Journal of Open Source Software* 6.60 (2021), p. 3021. DOI: 10.21105/joss.03021. URL: https://doi.org/10.21105/joss.03021.

[12] Tom O'Malley et al. *KerasTuner*. https://github.com/keras-team/keras-tuner. 2019.

[13] Kaiming He et al. "Identity Mappings in Deep Residual Networks." In: (2016). URL: https://arxiv.org/abs/1603.05027.