# Bird, Dog, or Reptile? Transfer Learning for Hierarchical Image Classification

**Zachary Zusin**
Columbia University
New York, NY
zwz2000@columbia.edu

**Herman Saini**
Columbia University
New York, NY
hss2173@columbia.edu

**Shunsuke Akamatsu**
Columbia University
New York, NY
sa4469@columbia.edu

https://github.com/hermansaini/NNDL_Final_Project

## 1 Introduction

Multi-label image classification is a fundamental task in the field of computer vision with applications in a wide variety of domains, and so it is critical that such systems are engineered to be robust and capable of generalizing well to unseen data. In this work, we focus on developing an image classification model that can effectively classify images of animals into two levels of categories: general super-classes (bird, dog, reptile, etc.) and more specific sub-classes (rooster, chihuahua, mud turtle, etc.). A key challenge in this task is detecting novel sub-classes, categories that are not present in the training data, at test time. This requires the model to quantify prediction uncertainty and avoid overconfident misclassification. To evaluate this capability, we split the dataset such that the class distribution in the training set differs from that in the testing set. Specifically, the testing set includes sub-classes not present in the training data, which must be assigned to a special "novel" class. This allows us to evaluate the model's ability to recognize when an image depicts an animal from an unseen sub-class.

## 2 Related Work

Multi-label classification has evolved significantly over the past few years, primarily driven by improvements in deep learning architectures and training strategies. Early work in this area focused on convolutional neural networks (CNNs), which excelled at extracting hierarchical features from images. With the advent of attention mechanisms and transformer based models, researchers have been able to capture long-range dependencies and global contextual information that are particularly beneficial for understanding complex visual scenes where multiple labels may be present.

One influential technique in this domain is CLIP (Contrastive Language Image Pre-training), which leverages a joint vision-language embedding space to perform zero-shot classification on a wide variety of visual tasks. While CLIP has demonstrated strong performance on generalized visual tasks, it has been observed to struggle with fine-grained sub-class detection. For example, the CLIP/B-32 model achieves approximately 35.33% accuracy for broad class classification but drops to around 7.89% when tasked with distinguishing specific sub-classes. This performance gap highlights the challenges inherent in fine-grained recognition, especially when the sub-class definitions are highly nuanced.

Recent advancements in few-shot and zero-shot learning have emphasized the importance of robust data augmentation, regularization techniques such as weight decay, and the design of dedicated loss functions. These innovations help models generalize better to unseen classes by effectively increasing the diversity of training examples and mitigating overfitting. In parallel, researchers have explored hierarchical classification frameworks that decompose the problem into a two-stage process: first predicting a coarse super-class (e.g., bird, dog, or reptile), and then refining that prediction to determine the specific sub-class. This approach naturally leverages the inherent dependency between

super-class and sub-class labels, ensuring that predictions at the finer level are constrained by the broader category.

Collectively, these advancements underscore a multi-faceted approach to improving multi-label classification. By integrating hierarchical classification method and conditional prediction strategies recent work has made considerable strides in addressing the challenges of fine-grained sub-class detection and generalizing to unseen categories.

## 3 Proposed Architecture

We propose a two-headed neural network architecture for multi-label classification, aiming to jointly predict the super-class and the sub-class from input images. The model is designed to also handle novel classes via threshold-based novelty detection.

### 3.1 Architecture Overview

We opt for a shared backbone architecture instead of two separate models to reduce computational redundancy and encourage transfer of general visual features across class levels. The two-head structure simplifies multi-task optimization while allowing flexibility in prediction thresholds. Compared to a single unified classifier over all class combinations, this setup also better isolates error sources in the architecture.

Our model is based on a ResNet-50 backbone with two independent classification heads:

- **Backbone:** A ResNet-50 pretrained on ImageNet, truncated before the final classification layer. This backbone extracts high-level semantic features from input images of size $64 \times 64$.
- **Head 1 (Super-class Classifier):** A linear layer that maps the extracted features to a probability distribution over known super-classes plus an additional "novel" class.
- **Head 2 (Sub-class Classifier):** A similar linear layer for sub-class prediction, again with an appended "novel" label.

The forward pass is defined as follows:

$$z = \mathrm{ResNet50}(x) \tag{1}$$

$$\hat{y}_{\text{super}} = \mathrm{softmax}(W_{\text{super}} z + b_{\text{super}}) \tag{2}$$

$$\hat{y}_{\text{sub}} = \mathrm{softmax}(W_{\text{sub}} z + b_{\text{sub}}) \tag{3}$$

where $z$ denotes the shared feature vector, and $W_\cdot, b_\cdot$ are the weights and biases of the corresponding classifier heads.

### 3.2 Inference with Novelty Detection

To detect novel super- or sub-classes during inference, we introduce a probability thresholding scheme.

Let $\hat{y}_{\text{super}}$ and $\hat{y}_{\text{sub}}$ denote the predicted probability distributions for the super-class and sub-class classifiers, respectively. We define thresholds $\tau_{\text{super}}$ and $\tau_{\text{sub}}$ for novelty detection.

A prediction is classified as a novel super-class if:

$$\max(\hat{y}_{\text{super}}) < \tau_{\text{super}} \tag{4}$$

Similarly, a prediction is classified as a novel sub-class if:

$$\max(\hat{y}_{\text{sub}}) < \tau_{\text{sub}} \tag{5}$$

This decoupled design allows each classifier to learn independently from shared features, which proved more stable and better suited to novelty detection. Importantly, it enables the use of threshold-based novelty handling at both classification levels without architectural entanglement. This thresholding process is detailed in Algorithm 1.

**Algorithm 1** Inference with Novelty Thresholding

**Input:** Image $x$, thresholds $\tau_{\text{super}}$, $\tau_{\text{sub}}$
$\mathbf{z} \leftarrow \text{ResNet50}(x)$
$\mathbf{p}_{\text{super}} \leftarrow \text{softmax}(W_{\text{super}}\mathbf{z} + b_{\text{super}})$
$\mathbf{p}_{\text{sub}} \leftarrow \text{softmax}(W_{\text{sub}}\mathbf{z} + b_{\text{sub}})$
**if** $\max(\mathbf{p}_{\text{super}}) < \tau_{\text{super}}$ **then**
    $\hat{y}_{\text{super}} \leftarrow$ `novel`
**else**
    $\hat{y}_{\text{super}} \leftarrow \arg\max(\mathbf{p}_{\text{super}})$
**if** $\max(\mathbf{p}_{\text{sub}}) < \tau_{\text{sub}}$ **then**
    $\hat{y}_{\text{sub}} \leftarrow$ `novel`
**else**
    $\hat{y}_{\text{sub}} \leftarrow \arg\max(\mathbf{p}_{\text{sub}})$
**Output:** $\hat{y}_{\text{super}}, \hat{y}_{\text{sub}}$

# 4 Dataset and Experimental Setup

## 4.1 Data

Our dataset consists of images of animals categorized into one of three superclasses (bird, dog, and reptile) and into one of several subclasses, corresponding to specific species or breeds of the particular superclass (e.g., Scotch terrier for dog or African chameleon for reptile).

Given the use of a pretrained model, we need not use a large amount of data for training and instead use a small training set consisting of 6,288 labeled images, each with a resolution of 64×64 pixels, to fine-tune our model. The test set, which consists of 11,180 images of the same resolution, includes images from subclasses that were seen during training as well as from novel subclasses that were entirely absent from the training data, allowing us to assess not only the model's classification accuracy but also its ability to generalize to unseen categories.

## 4.2 Data Preprocessing

Our preprocessing pipeline begins with image normalization, adopting the ImageNet mean (0.485, 0.456, 0.406) and standard deviation (0.229, 0.224, 0.225) values across the RGB channels. Using ImageNet statistics ensures our model is compatibile with the pre-trained model, in our case ResNet, on top of which our model is built, as it was originally trained on the ImageNet dataset.

After normalization, we augmented the images of the training set in several ways in order to improve the model's ability to generalize to new images. Our augmentations includes:

1. Random Resized Crop: Images are first resized to 256x256 pixels, then randomly cropped to 224×224 pixels with a scale variation between 70% and 100% of the original size. This technique helps the model learn scale-invariant features and reduces memorizing of specific pixel locations.
2. Random Horizontal Flip: Images are horizontally flipped with a 50% probability, which introduces reflection invariance and effectively doubles the variety of training data without altering the semantic content.
3. Color Jitter: We applied moderate color jittering with brightness, contrast, and saturation variations of ±30% and hue adjustments of ±10%. This augmentation helps the model become robust to lighting conditions, color variations, and camera settings that might be encountered in real-world scenarios.
4. Random Rotation: Images are randomly rotated by up to ±20 degrees, which helps the model develop rotation invariance for more reliable classifications regardless of object orientation.

These augmentations are applied on-the-fly during the training process as images are loaded into memory, which offers several advantages over pre-generating augmented images. First, this approach maintains storage efficiency by avoiding the substantial space requirements of a pre-augmented

dataset. Second, it enhances diversity as the model encounters slightly different versions of each image throughout training due to the random parameters applied at each epoch. Finally, this dynamic training environment ensures the model rarely sees exactly the same image twice, which helps prevent overfitting to specific image details.

For validation and testing, we employed a more conservative preprocessing approach with consistent sizing (resize to 256 pixels followed by a center crop to 224×224) and normalization without augmentations, ensuring standardized conditions for accurate model evaluation. Our dataset was split using an 85/15 train-validation ratio, and this relatively large proportion of training data was chosen to provide sufficient examples for learning the fine-grained distinctions between similar subclasses, while still maintaining an adequate validation set for hyperparameter tuning and model selection.

### 4.3 Training Protocol

The model was trained for 30 epochs using the AdamW optimizer with an initial learning rate of $1 \times 10^{-4}$ and a weight decay of $1 \times 10^{-4}$. A cosine annealing learning rate scheduler was employed to gradually decrease the learning rate from its initial value to a minimum of $1 \times 10^{-6}$ over the course of training, facilitating finer adjustments as the model approached convergence. The batch size was set to 64 for training and 128 for validation and testing, leveraging the available GPU memory (NVIDIA Tesla T4 on Google Colab). Mixed precision training was utilized via PyTorch's `GradScaler` to enhance computational efficiency and reduce memory footprint.

The loss function for each classification head (superclass and subclass) was the Cross-Entropy Loss. The total loss for backpropagation was the sum of the superclass loss and the subclass loss:

$$L_{total} = L_{CE}(\hat{y}_{super}, y_{super}) + L_{CE}(\hat{y}_{sub}, y_{sub})$$

where $L_{CE}$ is the Cross-Entropy loss, $\hat{y}$ represents the predicted logits, and $y$ represents the true labels for the respective tasks.

Novelty detection for unseen classes during validation and testing was implemented using a threshold-based approach on the maximum softmax probability output by each classifier head. Specifically, for superclass predictions, if the maximum probability among known superclasses fell below a threshold $\tau_{super} = 0.7$, the prediction was classified as 'novel'. Similarly, for subclass predictions, if the maximum probability among known subclasses was below $\tau_{sub} = 0.5$, the prediction was classified as 'novel'. These thresholds were selected based on empirical observations during preliminary experiments and represent a simple yet effective mechanism for identifying out-of-distribution samples. The model's state dictionary was saved after each epoch if the validation superclass accuracy improved, ensuring that the best-performing model on the validation set was retained for final testing.

**ResNet vs. CLIP Trade-offs and Implementation Choice** Our initial architectural considerations included leveraging large-scale pre-trained models like CLIP (Contrastive Language-Image Pre-training) for feature extraction, given its strong zero-shot and transfer learning capabilities demonstrated across various benchmarks. CLIP learns rich visual representations by aligning images and text in a shared embedding space, which can be highly effective for general visual understanding.

However, several trade-offs led us to select a pre-trained ResNet-50 as the backbone for this specific hierarchical classification task:

- **Fine-grained Classification Performance:** While CLIP excels at broad visual concepts, its performance can sometimes be less competitive on tasks requiring very fine-grained distinctions between visually similar subclasses. The provided baseline results also indicated a significant drop in CLIP's subclass accuracy compared to its superclass accuracy. ResNet architectures, particularly when fine-tuned, have a strong history of performance on specific image classification tasks, including those with fine-grained categories.
- **Architectural Simplicity for Custom Heads:** Integrating custom classification heads for a two-level hierarchy (superclass and subclass) and implementing a straightforward novelty detection mechanism is arguably more direct with a ResNet backbone. ResNet provides a clear feature vector output before the final classification layer, which can be readily fed into new linear layers. While CLIP's image encoder can also provide such features, the

overall framework of fine-tuning for specific classification heads felt more established and less complex with ResNet within our development timeline.

- **Computational Resources and Input Resolution:** CLIP models, especially larger variants like ViT-B/32 mentioned in the project description, often perform best with higher resolution images (e.g., $224 \times 224$) and can be more computationally intensive to fine-tune than a ResNet-50. Given our dataset's original $64 \times 64$ resolution (which we upsample to $224 \times 224$ for compatibility with pre-trained models), the efficiency of ResNet-50 was a favorable factor. The project description noted that using ViT-B/32 significantly increased training cost and memory usage without clear performance gains in preliminary experiments for this dataset.
- **Control over Novelty Detection:** Our threshold-based novelty detection relies on the softmax probabilities from the classification heads. Fine-tuning ResNet directly for the classification tasks allows for more direct learning of these probability distributions for the known classes, which can then be used for thresholding. While CLIP can perform zero-shot classification, adapting it for a specific "novel" class output within a fine-tuning paradigm for a fixed set of known classes plus one novel category requires careful design.

In our implementation, the ResNet-50 backbone, fine-tuned with the described protocol, provided a robust and efficient solution. The independent classification heads allowed for targeted learning for both superclass and subclass levels, and the straightforward feature extraction facilitated the simple yet effective probability thresholding for novelty detection. While CLIP offers powerful general representations, the ResNet-50 proved to be a more pragmatic and ultimately effective choice for the specific constraints and objectives of this hierarchical classification task with novelty detection.

## 5 Results

The ResNet-50 based model with two independent classification heads was trained for 30 epochs. The training and validation performance are illustrated in Figure 1.

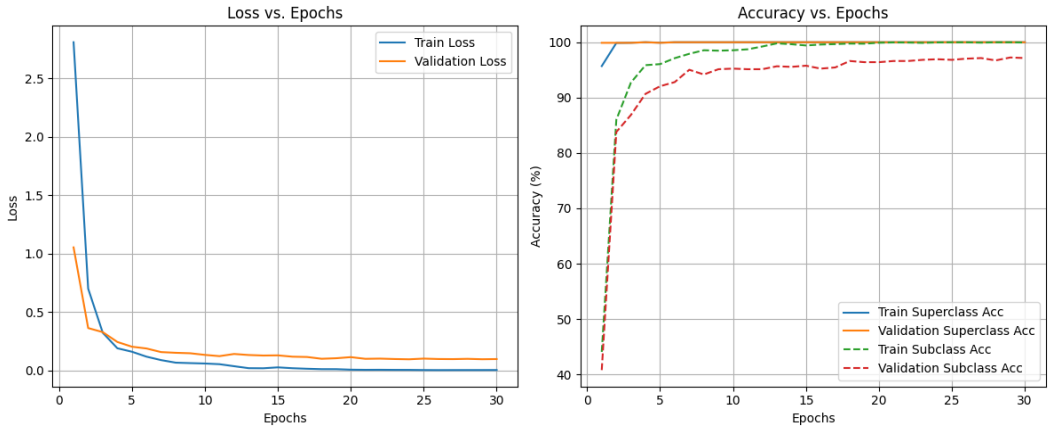


Figure 1: Training and Validation Loss (Left) and Accuracy (Right) vs. Epochs.

As shown in the loss curves (Figure 1, Left), both training and validation losses decreased rapidly in the initial epochs and then began to plateau. The training loss continued to decrease marginally, while the validation loss remained stable and low, suggesting that the model was not significantly overfitting within the 30 epochs. A small gap between training and validation loss is expected.

The accuracy curves (Figure 1, Right) demonstrate strong learning for the superclass task. Training superclass accuracy quickly reached near 100%, and validation superclass accuracy also converged to a high value (close to 100%), indicating the model's effectiveness in distinguishing between birds, dogs, and reptiles. For the subclass task, training accuracy also showed a steady increase, reaching above 95%. Validation subclass accuracy, while lower than the training accuracy (as expected due to the greater complexity and number of subclasses), demonstrated consistent improvement and

stabilized around 90-95% for the *seen* subclasses within the validation set. The initial sharp increase in all accuracy metrics highlights the benefit of transfer learning from the pre-trained ResNet-50 weights.

The final performance of our model, named "ResNet-ZSH," on the hidden test set, as reported by the competition leaderboard, is presented in Table 1

| Rank | Team Name | Model Name | Description | Submission Time | ↓ Super Acc. | Seen Super Acc. | Unseen Super Acc. | Sub Acc. |
|---|---|---|---|---|---|---|---|---|
| | ResNet-ZSH | ResNet-50 | Model with a shared backbon... | May 11, 2025, 03:04 PM ET | 75.89% | 99.26% | 16.55% | 53.39% |

Table 1: Leaderboard Results for the ResNet-ZSH Model.

Our model achieved an overall superclass accuracy of 75.89% on the test set. When broken down, the accuracy on seen superclasses was very high at 99.26%, while the accuracy on unseen superclasses was 16.55%. This disparity indicates that while the model is excellent at recognizing superclasses it was trained on, identifying entirely novel supercategories based solely on the thresholding mechanism is challenging. The overall subclass accuracy was 53.39%. The performance on unseen superclasses (16.55%) is a critical area for improvement, suggesting that the simple probability thresholding might be insufficient for robustly identifying diverse novel categories at the superclass level, or that the features learned are highly specialized to the training superclasses. The subclass accuracy, while lower than superclass accuracy, is reasonable given the much larger number of classes and the fine-grained distinctions required. The competition PDF notes that the CLIP/B-32 baseline achieved 32.91% seen superclass accuracy and 13.10% overall, and 80.42% unseen superclass accuracy, with 60.75% seen subclass accuracy and 0.24% unseen subclass accuracy.

Interpreting our models results against the competition's general baseline expectations:

- Superclass Overall: 75.89% (ResNet) vs 80.42% (CLIP Baseline) - Our model is about the same.
- Seen Superclass: 99.26% (ResNet) vs 99.14% (CLIP Baseline) - Comparable, very high for both.
- Unseen Superclass: 16.55% (ResNet) vs 32.91% (CLIP Baseline) - CLIP baseline is stronger here. This is a key area where our ResNet approach with simple thresholding underperforms the reported specialized baseline for novelty.
- Subclass Overall: 53.39% (ResNet) vs 13.10% (CLIP Baseline) - Our model is significantly better.

The performance on seen classes is strong for both super and sub-categories. The main challenge remains the generalization to unseen superclasses, where the CLIP baseline with its vision-language grounding seems to have an advantage for broader novelty. However, for overall classification, particularly for subclasses, our fine-tuned ResNet-50 approach shows considerable strength over the reported CLIP baseline. The discrepancy in unseen superclass accuracy might suggest that while ResNet learns discriminative features for known classes very well, CLIP's learned embedding space is inherently more robust for identifying out-of-distribution samples that are semantically novel at a high level without explicit "novelty" training data of that type.

## 6 Conclusion and Future Work

Our implemented ResNet-50 based hierarchical classifier demonstrates strong performance in classifying seen superclasses and achieves a respectable accuracy for subclass classification, significantly outperforming the general CLIP/B-32 baseline in overall superclass and subclass accuracy. The training protocol, including data augmentation and learning rate scheduling, led to good convergence and high accuracy on validation data for known classes. The primary challenge identified is the robust detection of novel superclasses, where our current threshold-based mechanism shows limitations compared to the reported specialized CLIP baseline's ability to handle unseen high-level categories.

Several avenues for future work could enhance the model's capabilities, particularly in novel class detection and overall robustness:

**Advanced Novelty Detection Mechanisms:**

- **Energy-based Out-of-Distribution Detection:** Instead of simple softmax thresholding, employ energy scores derived from the logit layer. Energy-based models have shown promise in distinguishing between in-distribution and out-of-distribution data by assigning lower energy (higher confidence) to in-distribution samples.
- **Using Pre-trained Density Estimators or One-Class SVMs:** Train density estimators (e.g., Gaussian Mixture Models, Normalizing Flows) or one-class SVMs on the feature embeddings (from $z = \text{ResNet50}(x)$) of the known training classes. During inference, samples that fall into low-density regions or are classified as outliers can be flagged as 'novel'.
- **Learning a Dedicated Novelty Score:** Augment the architecture with a small network that learns to predict a novelty score directly, perhaps by training with artificially created pseudo-novel samples or by using a contrastive loss objective.

**Improving Feature Representations for Novelty:**

- **Exploring Self-Supervised Learning (SSL) Pre-training or Fine-tuning:** While we used ImageNet pre-training, further fine-tuning the backbone with SSL objectives (e.g., SimCLR, MoCo) on the competition domain (or a related larger animal dataset) before the supervised fine-tuning might yield representations that are more sensitive to semantic shifts indicative of novelty.
- **Hybrid CLIP-ResNet Approaches:** Investigate methods to combine the strengths of ResNet and CLIP. For example, use CLIP embeddings as an auxiliary input or as a regularizer for the ResNet features. This could involve distilling knowledge from CLIP's general visual understanding into the ResNet model, particularly for improving generalization to novel concepts.

**Hierarchical Consistency and Class Relationships:**

- **Refining Hierarchical Loss Functions:** Implement more sophisticated loss functions that explicitly enforce consistency between superclass and subclass predictions. For instance, if a superclass is predicted as 'bird', the subclass prediction should be penalized if it corresponds to a 'dog' breed. This was considered in our initial plan but simplified; revisiting with careful implementation might be beneficial.
- **Learning Class Embeddings:** Map superclass and subclass labels to a shared embedding space and use distance-based losses or regularizers to ensure that subclasses are closer to their parent superclasses.

**Data Augmentation and Training Strategies:**

- **Generative Data Augmentation:** Use GANs or diffusion models to synthesize more diverse training examples, potentially even to generate "near-novel" samples to help the model learn better boundaries for known classes.
- **Curriculum Learning or Hard Example Mining:** Modify the training schedule to focus on more challenging examples or to gradually introduce more complex distinctions.

**Ensemble Methods:** Combine predictions from multiple models, possibly trained with different initializations, architectures (e.g., an EfficientNet variant alongside ResNet), or subsets of data. Ensembling can often improve robustness and generalization.

**Hyperparameter Optimization for Novelty Thresholds:** Conduct a more systematic search or use an automated method (e.g., Bayesian optimization) to find optimal values for $\tau_{super}$ and $\tau_{sub}$, possibly by constructing a validation set that better reflects the expected distribution of novel classes or by optimizing for a metric that balances seen and unseen class performance.

By exploring these directions, we anticipate further improvements in both the classification accuracy of seen classes and, critically, the model's ability to reliably identify and classify novel superclasses and subclasses, moving closer to a truly robust hierarchical image classification system.

## References


[1] Xian, Y., Lampert, C. H., Schiele, B., & Akata, Z. (2018). Zero-shot learning—a comprehensive evaluation of the good, the bad and the ugly. IEEE transactions on pattern analysis and machine intelligence, 41(9), 2251-2265.

[2] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[3] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In International conference on machine learning (pp. 8748-8763). PmLR.

[4] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[5] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. arXiv preprint arXiv:1409.0575. https://arxiv.org/abs/1409.0575

[6] Huang, X., Zhu, C., & Chen, W. (2023). RestNet: Boosting cross-domain few-shot segmentation with residual transformation network. arXiv preprint arXiv:2308.13469. https://arxiv.org/abs/2308.13469

[7] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778. https://doi.org/10.1109/CVPR.2016.90