NNDL Fall 2023 Project Report

Yucheng Wang	Xuezhen Wang	Xinchen Zhang	Kaggle Group Name
yw3720	ww2604	xz3052	LoveZemel♥

1 Introduction

The project is focused on hierarchical image classification across three main entity classes: bird, dog, and reptile. Each of these classes encompasses various sub-classes, such as hawk within birds, golden retriever within dogs, etc. The primary challenge of the project arises during the testing phase, where images belonging to both novel classes and sub-classes are introduced. Consequently, the model must be adept not only at effectively classifying images from the seen classes and sub-classes but also at distinguishing images from classes and sub-classes that were not encountered during its training. Compounding this challenge is the limited amount of training data available, despite a large number of labels for sub-classes. Furthermore, the sample data distribution in the training set may differ significantly from that in the testing set, meaning that the class frequency in the training set may not be the same as in the testing set.

Our proposed methods are designed to tackle the three key challenges mentioned earlier, with the goal of achieving high accuracy in the classification task. To address the issues of insufficient data and distribution shifts, we adopt a few-shot learning framework. In few-shot learning, a classifier is adapted to new classes with only a few examples from each class [16, 10]. In our case, the availability of more data per class compared to a typical few-shot learning scenario leads us to anticipate even better performance.

Our model is inspired by the Prototypical Network [22] for three main reasons. Firstly, our dataset comprises common animals, a typical training source for various large models, allowing our model to potentially leverage these pre-trained models for general feature representation derived from external data. We have modified the Prototypical Network framework to incorporate pre-trained models seamlessly into our pipeline. Secondly, the Prototypical Network, owing to its few-shot learning nature, can calculate a class prototype using only a few instances with the same label, effectively representing the class in the embedding space. This approach helps mitigate the problem of limited data for specific classes or sub-classes caused by distribution shifts, as only a few examples are needed to learn a prototype. Finally, the Prototypical Network maps each image to a embedding space and employs a distance metric for clustering and classification. This methodology provides us with a mechanism to use distance metrics to differentiate novel classes from known ones.

The pipeline of our method is meticulously designed for effective classification. It involves a neural network to perform a non-linear mapping of inputs into an embedding space, potentially integrating pre-trained large models. Within this space, we define a class's prototype as the mean of its support set. For classification, an embedded query point with a known class is simply matched to the nearest class prototype. For query points with novel classes, we introduce two distinct methods, detailed in Section 2.1 and Section 2.2.

The first method, employed during the training phase, focuses on learning the non-linear mapping function. Additionally, we separate an "auxiliary set" from the dataset, containing only seen labels, to establish a threshold. This threshold is critical for differentiating novel instances from seen ones during the testing phase. After training the mapping function, the support set is fed into the network to generate prototypes in the embedding space. These prototypes are then used to predict labels for instances in the "auxiliary set" by mapping them to the embedding space using the learned function. The label of the closest prototype is assigned to each instance. The threshold is determined by the

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

maximum distance in the embedding space between correctly classified instances and their respective prototypes. During testing, we calculate the distance between each mapped instance and the nearest prototype. Instances with a distance exceeding the threshold are classified as novel. For others, we assign the label of their nearest prototype. This approach ensures accurate classification of both known and novel classes. Figure 1 illustrates the inference stage of this method.

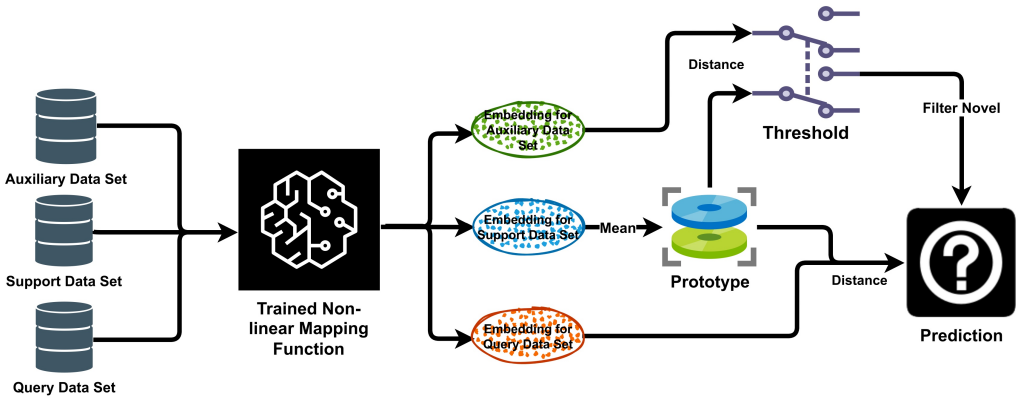

Figure 1: Method 1 Illustration

In our second method, which is a refinement of the first, we initially learn a non-linear mapping function during the early training phase. This function maps input images into an embedding space, where examples belonging to the same class cluster closely together, and those from different classes are more distant. Following the standard prototypical network approach, we start by computing an initial prototype for each class using the support set. Subsequently, the query set is utilized to predict the parameters of a soft K-Means clustering function. This is achieved by inputting the normalized distances between query samples and initial prototypes into a multi-layer perceptron. The resulting K-Means clustering assigns each sample a class-specific mask, which is then used to refine the prototypes. Predictions are made based on the proximity between query samples and these calibrated prototypes, with novel classes being identified through a comparative analysis of the masks computed earlier. Figure 2 illustrates the inference stage of this method.

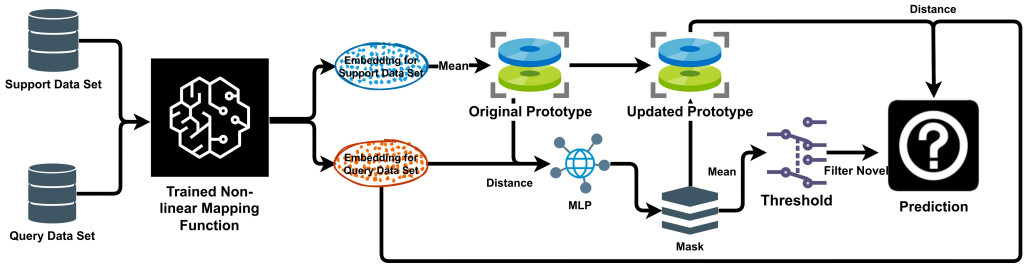

Figure 2: Method 2 Illustration

2 Methods

2.1 Prototypical Model

As outlined in the introduction, we have selected the Prototypical Model concept as our foundational approach. This choice is driven by its capability to project images into a high-dimensional space, enabling the use of distance metrics to distinctively separate novel classes from known ones, which is the core challenge in this project. Its proven efficacy in few-shot learning scenarios, particularly with limited data, aligns well with our project where we are working with a constrained subset of subclass images. Furthermore, the model's inherent flexibility in modifying feature mapping functions allows

us to effectively leverage the strengths of pre-trained models. The original Prototypical Network is designed primarily for image classification within the few-shot learning framework. However, it does not account for the introduction of novel classes during the testing phase [22]. **This section aims to introduce how we modify the original framework so that it's capable of leveraging the pre-trained models and distinguish the novel class from the known ones.**

As shown in Figure 3 modified from Snell et al.'s work [22], the fundamental idea is to learn a metric space where classification is conducted by calculating distances to the prototype representations p_c of each class. Specifically, we compute a *prototype*, an M-dimensional representation $p_c \in \mathbb{R}^M$ for each class c through a function $f_\theta : \mathbb{R}^{size} \to \mathbb{R}^M$ where θ is the parameter to learn. The prototype for each class is defined as the mean vector of the embedded support points from that class:

$$p_c = \frac{1}{|S_c|} \Sigma_{(x_i, y_i) \in S_c} f_\theta(x_i)$$

Leveraging Pre-trained Model. In the original study, a straightforward CNN network serves as the embedding function f_θ. However, in our project, considering the specific characteristics of our dataset, which predominantly includes common animals, we propose a modified approach. Instead of directly inputting x into the embedding function, we first utilize the feature layers from pre-trained large-scale models. These layers, having been trained on extensive datasets, are likely to contain valuable visual information. By inputting x into this pre-trained model, we generate a new input, x'. This new input x' is then processed through a lightweight, fully connected network housing the learnable parameter θ, thereby mapping it to the final metric space. Essentially, our method integrates the pre-trained model with a fully connected network to form the function f_θ. This combination not only serves as an embedding mechanism but also fine-tunes the pre-trained network to better suit our specific task. Specifically, our study incorporates two well-established pre-trained models: ResNet-50 [8] and the Vision Transformer (ViT) [5, 26].

In the original study, provided a function f_θ that maps inputs to an embedding space, and a distance metric $d : \mathbb{R}^M \times \mathbb{R}^M \to [0, +\infty)$, it generate a distribution across all classes for an input x by applying a softmax function to the computed distances between the embedded point and a set of class-specific prototypes within the embedding space:

$$p_\theta(y = k|x) = \frac{\exp(-d(f_\theta(x), p_k))}{\sum_{k'} \exp(-d(f_\theta(x), p_{k'}))}$$

Predicting Novel Class. Nevertheless, this approach does not account for scenarios where the input x does not correspond to any class k identified during training. To manage this situation, in the absence of novel classes, we will evaluate on a local dataset during "auxiliary set testing". Here, a threshold is determined as the maximum distance amongst all correctly classified instances and their respective prototypes. Intuitively, this threshold signifies the extent to which an instance can be separated, according to the distance metric d, from its prototype before it is deemed to no longer be associated with that class. During testing, rather than assigning a label based on the highest $p_\theta(y = k|x)$, we first determine the distance from each instance to its nearest prototype p_k. If the distance $d(f_\theta(x), p_k)$ is less than the threshold established during local testing, we classify the label as novel. If it exceeds the threshold, we assign the label as k.

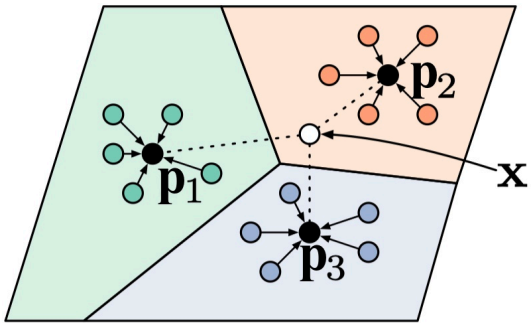

Figure 3: Prototypical Network Illustration

2.2 Refined Prototypical Model

2.2.1 Inspiration from Meta-learning for Semi-supervised Few-shot Classification

The Prototypical Network serves as a crucial methodology for addressing few-shot learning challenges. However, its efficacy in recognizing samples from new classes during the testing phase is limited. Currently, we employ a fixed threshold based on the Euclidean distance, calculated using the sample's features and its corresponding prototype. **This section aims to elaborate on an enhancement of the Prototypical Network, drawing inspiration from a subsequent study, which leverages an unlabeled dataset for semi-supervised learning to augment the Prototypical Network's performance[21].**

Ren et al. examine scenarios where unlabeled samples are present in each episode of a few-shot classification task. They identify two distinct cases: one where all unlabeled samples correspond to the target labels, and another involving samples from various, or *distractor*, classes. These unlabeled datasets contribute to semi-supervised learning, thereby enhancing the robustness of the Prototypical Network.

Their methodology commences with the fundamental definition of prototypes $\mathbf{p}_c$ introduced above and culminates in a refined prototype $\tilde{\mathbf{p}}_c$, utilizing the unlabeled dataset. The refinement process involves estimating the cluster assignments of the unlabeled samples and subsequently modifying the cluster locations, i.e., the prototypes. Ren et al. employ k-means clustering as the primary method for assigning these unlabeled data. To effectively manage various distractor classes, they propose a strategy for modeling distractors as examples that fall outside a specific vicinity of any legitimate class prototypes, rather than defining distractors via a high-variance, all-encompassing cluster. This approach resonates with the fundamental principle of identifying new classes in our research. They introduced a soft-masking technique to regulate the influence of unlabeled samples, aiming for samples nearer to a prototype to have a reduced masking effect compared to those more distant. Furthermore, they developed a compact neural network to predict the soft thresholds and gradient for each prototype. This MLP determines the involvement of unlabeled samples in prototype adjustment. In the subsequent section, we will delve deeper into how our project references their work and implements modifications tailored to our objectives.

2.2.2 Clustering Using Query Set

In our endeavor, direct application of the methods from the referenced work is infeasible, as we lack access to an unlabeled dataset. Allocating a portion of the training set to serve as an 'unlabeled' subset would result in the underutilization of available label information, leading to suboptimal performance compared to models that exploit the full training set. Nevertheless, the concept of refining prototypes with unlabeled data offers valuable insight. **Instead of utilizing the unlabeled dataset which is not available, we propose to subtly recalibrate the prototypes using the query set, which, while unlabeled, can still provide useful information.** First, same as the procedure we have in the last section, we utilized a function f_θ to mapp all inputs to an embedding space, leveraging the pre-trained models. Subsequently, we adopt the soft k-means refinement technique from the cited paper, calculating normalized distances $\tilde{d}_{j,c}$ between the embeddings of each query sample $\tilde{\mathbf{x}}_j$ and the prototypes $\tilde{\mathbf{p}}_c$:

$$\tilde{d}_{j,c} = \frac{d_{j,c}}{\frac{1}{M}\sum_j d_{j,c}}, \quad \text{where} \quad d_{j,c} = \|f_\theta(\tilde{x}_j) - p_c\|_2^2$$

Subsequently, a multilayer perceptron (MLP) is employed to predict the soft threshold β_c and slope γ_c for each prototype based on a variety of statistics derived from the normalized distances to the prototype:

$$[\beta_c, \gamma_c] = \text{MLP}\left(\left[\min_j \tilde{d}_{j,c}, \quad \max_j \tilde{d}_{j,c}, \quad \text{var}_j(\tilde{d}_{j,c}), \quad \text{skew}_j(\tilde{d}_{j,c}), \quad \text{kurt}_j(\tilde{d}_{j,c})\right]\right)$$

. These parameters serve a dual function. Primarily, they are instrumental in ascertaining whether a sample should influence the adjustment of prototypes. This is executed by computing soft masks $m_{j,c}$ that gauge each example's contribution to the prototype, aligned against the normalized distances threshold:

$$\hat{p}_c = \frac{\sum_i f_\theta(x_i) z_{i,c} + \sum_j f_\theta(\tilde{x}_j)\tilde{z}_{j,c} m_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c} m_{j,c}}, \quad \text{where} \quad m_{j,c} = \sigma(-\gamma_c(\tilde{d}_{j,c} - \beta_c))$$

, with $\sigma(\cdot)$ denoting the sigmoid function. Secondly, they determine the likelihood of a sample being classified into a novel class during testing. This is achieved by calculating the maximum $m_{j,c}$ across all classes for each example $\mathbf{x}_j$ and contrasting this maximum with the average. A higher mask value suggests a stronger affiliation with one of the existing classes, whereas a lower maximum indicates a greater probability of the example being a distractor, which in our context, signifies a novel class. It is critical to note that, due to the absence of distractor classes during training, the MLP is trained solely based on its capacity to modify prototypes. To mitigate overfitting to a zero-distractor scenario, we introduce a dropout layer within the MLP to enhance its robustness.

3 Experiments

3.1 Dataset Preprocessing

The training dataset comprises a total of 6,322 images, each sized at 32×32 pixels. This dataset includes 1,850 bird images, 2,085 dog images, and 2,388 reptile images. These images are further categorized into 87 sub-classes, with the largest sub-class containing 102 images and the smallest sub-class containing 49 images. To ensure unbiased feature mapping across all super-classes and sub-classes, we created two augmented datasets for super-class and sub-class training, respectively. Augmentation involved horizontal and vertical flipping to equalize the number of images per label. Consequently, the super-class augmented dataset consists of 7,164 images, with an equal distribution of 2,388 images for each super-class. Similarly, the sub-class augmented dataset contains 8,874 images, with 102 images for each sub-class. Additionally, we normalized the data to achieve a mean of 0 and a standard deviation of 1, further enhancing the robustness of our training process.

3.2 Experiment Setup

The training and testing for super-class and sub-class label is separated. We divided the dataset into three segments: 70% for the training set, 20% for the validation set, and 10% for the testing set for super-class labels. For the training and validation sets, they were equally split into two halves, one forming the support set and the other the query set. We utilized a support set of 100 images and a query set of 100 images for each class in every batch. The test set, also serving as the auxiliary set, was used to evaluate the model in scenarios where there are no images from novel classes. For the sub-class labels, the dataset was similarly divided into three parts, with 80% for training, 10% for validation, and 10% for testing. In each iteration, the model was fed with 5 support images and 5 query images for each class.

For both methods, we conducted tests using three different deep learning models: a vanilla Convolutional Neural Network, a pre-trained Resnet 50[8], and the Vision Transformer (ViT) [5, 26]. The CNN model follows the embedding architecture used by Vinyals et al[25], comprising four convolutional blocks. Each block includes a 128-filter 3× 3 convolution, a batch normalization layer[9], a ReLU nonlinearity, and a 2×2 max-pooling layer, culminating in a 96-dimensional output space. The pre-trained Resnet model selected is the Resnet 50 from the PyTorch library, with its last fully connected layer removed. Additionally, we added two fully connected layers for training, each consisting of a linear function, a ReLU nonlinearity, and a dropout layer with a dropout rate of 0.5. The first layer outputs 4096 dimensions, while the final layer yields an embedding of size 2048. The selected ViT is the small version with 30.1M parameters. As the model necessitates an input image size of 224×224, we included an additional PyTorch embedded resize function at the beginning of the model. Similar to the Resnet, we removed the last layer and added two fully connected layers, each with outputs of dimensions 1024 and 512, respectively.

3.3 Results

Our findings are detailed in Table 1, where we present the validation accuracy, the test accuracy on a locally separated test set without unseen labels, and the test accuracy on a Kaggle test set with unseen labels for super-class classification. These tables enumerate the accuracies achieved by three distinct feature mapping functions employed to map images into our adapted embedding space. These functions include a simple CNN network, a pre-trained ResNet-50 followed by a Multi-Layer Perceptron (MLP), and a pre-trained Vision Transformer (Vit) followed by an MLP. Notably, for the leader-board competition, our best result for super-class classification was an accuracy of 83.5%,

achieved using the revised prototypical method with a pre-trained Vit followed by an MLP. Similarly, the peak accuracy for sub-class classification was 79.8%, attained by both the original and revised prototypical methods, each utilizing a pre-trained Vit followed by an MLP.

Super Class		Validation Accuracy	Test Accuracy (w.o novel)	Test Accuracy (with novel)
Prototypical	CNN	92.1%	92.0%	59.4%
	Pre-trained Resnet	95.7%	94.8%	59.5%
	Pre-trained Vit	99.5%	99.3%	53.5%
Revised Prototypical	CNN	94.3%	91.4%	61.0%
	Pre-trained Resnet	97.3%	95.6%	67.1%
	Pre-trained Vit	99.3%	99.4%	**83.5%**
Sub Class		Validation Accuracy	Test Accuracy (w.o novel)	Test Accuracy (with novel)
Prototypical	CNN	52.2%	65.8%	73.7%
	Pre-trained Resnet	68.1%	71.5%	37.4%
	Pre-trained Vit	86.4%	87.5%	**79.8%**
Revised Prototypical	CNN	86.3%	75.6%	72.9%
	Pre-trained Resnet	78.1%	73.9%	76.6%
	Pre-trained Vit	91.3%	86.8%	**79.8%**

Table 1: Results for Super-class/Sub-class Classification

Good Embedding is All You Need. The analysis reveals that the highest classification accuracies are consistently obtained when employing the pre-trained Vision Transformer (Vit) model as part of the feature mapping function. This approach benefits from the transfer of knowledge from other tasks, which is crucial given that the effectiveness of our framework—primarily the identification of a representative prototype for each class—hinges on learning robust embedding features for each instance.

Robustness for Threshold Computation. It is noteworthy that the use of pre-trained models does not enhance the performance of the prototypical method for super-class classification. Moreover, incorporating ResNet-50 results in a marked decrease in performance for the prototypical method in sub-class classification. These declines in performance are exclusively observed in the prototypical method, which employs a hard threshold to differentiate between novel and known classes. This observation points to a significant consideration: the difference in threshold computation between the two models, suggesting that the thresholding approach in the prototypical method may require adaptation or reevaluation when different underlying models are used. Evidently, the threshold computation within the prototypical method exhibits a lack of robustness, as it solely relies on an auxiliary set—separated from the training set—to determine the threshold. This approach overlooks potentially valuable information present in the test set. Consequently, even if the model achieves a high-quality embedding, the instability of this threshold can impair its ability to effectively differentiate between novel and seen classes. In contrast, the threshold computation in the revised prototypical method incorporates information from the test set, aiming to indirectly derive a more robust threshold through a Multi-Layer Perceptron (MLP). This methodological refinement allows for a more dynamic and context-aware threshold setting, potentially enhancing the model's capability to distinguish novel classes from known ones with greater accuracy and reliability.

4 Related Work

Few-shot Learning. Few-shot image classification is a challenging image classification problem facing a lack of dataset but aims to achieve the human level of recognition. Recent approaches focus on efficiently learning from a limited number of samples and generalize well to new, unseen data [23] [20] [21]. The overall goal is to mimic human learning efficiency, enabling the model to recognize and classify objects from very few examples. One significant approach in this domain is deep metric learning [13]. This involves learning feature embeddings, class representations, and distance or similarity measures. Notable methods include the Siamese Convolutional Neural Network [15], the BiDirectional Matching Network [6], and the Transductive Propagation Network [14].

Another notable development is the extension of prototypical networks in semi-supervised few-shot classification. This approach involves graph neural networks used for semi-supervised and active learning [7], and the relation network that extends the matching network and prototypical network [24]. These models, along with others like the category traversal module (CTM) [11] and the deep nearest neighbor neural network [12], emphasizes the importance of feature embedding in improving classification in few-shot learning scenarios. In our model, our first approach, detailed in Section 2.1, draws inspiration from the Prototypical Network [23] and benefits from existing large models' feature representations. It calculates class prototypes using minimal instances, addressing the data scarcity issue by mapping each image to a embedding space and introduce a new distance metric for clustering and classification. We extend the prototypical model by incorporating semi-supervised few-shot classification according to Ren's meta learning approach[21] and k-means clustering[2], as shown in Section 2.2, enhancing the model's ability to accurately classify images with novel classes.

Novel Class Detection. Image classification focusing on generating unseen classes has led to the development of innovative approaches. One such approach is the Pairwise Classification Network [1], which learns intra-class and inter-class differences from seen classes using a binary classification model. This model is later transferred to unseen classes as guidance for uncovering these classes through clustering. This approach also utilizes an auto-encoder to learn unsupervised representations from unlabeled examples, preventing overfitting to seen classes and improving representation for unseen classes. Besides, using hierarchical clustering [18] to identify the number of clusters in rejected examples is also an essential step as the number of clusters is unknown in open environments.

Transfer Learning. In image classification scenarios with limited training data, transfer learning has been a focal point of recent research due to its effectiveness. It intersects with many areas, including semi-supervised learning [29], multi-view learning [27], and multi-task learning [28]. In the real world, there are many scenarios where the amounts of training data are extremely limited due to originally rare datasets, privacy concerns, or expensive cost to collect sufficient data. Large CNNs such as ImageNetDeep [4] are often overfitting when trained from random initialization on standard datasets. Deep transfer learning [19], however, can greatly improve the performance. It is especially pertinent in image classification. In real-world scenarios such as sparse medical image classification [17], deep transfer learning has shown particular advantages than CNN.

5 Limitations and Future Works

Determination of Threshold In the methods we have explored, the determination of the threshold either relies on the auxiliary set (Sec 2.1), or relies on a comparative analysis of all testing samples (Sec 2.2). Both methods have limitations. In the prototypical method, the threshold is entirely decided by auxiliary set, which might not be representative in some case, as we discussed in the Sec 3.3. While the refined Prototypical Model offers a robust evaluation of each sample's affiliation with the prototype, the fundamental approach to establishing the threshold remains a horizontal comparison across different test samples. This method implies a need for approximate knowledge of the test sample distribution to achieve satisfactory results. A promising direction for future improvement is the application of Deep Hybrid Models (DHMs), as proposed in 'Deep Hybrid Models for Out-of-Distribution Detection' by Cao et al[3]. DHMs offer a sophisticated approach to model uncertainty, utilizing a combination of Spectral Normalization and Normalizing Flows. This method factorizes uncertainty into aleatoric and epistemic components, enabling the model to more effectively discern whether samples conform to the training data distribution. By integrating such an approach, we can potentially move beyond reliance on comparative thresholds and enhance the robustness of our OoD detection capabilities without the prerequisite of detailed test distribution knowledge.

6 Conclusion

In this project, we address the complex task of hierarchical image classification, specifically incorporating novel classes in test time, through the innovative application and modification of Prototypical Networks. Our approach, focusing on representing classes by their means in a neural network-trained representation space, efficiently utilizes the strengths of pre-trained models. This strategy is particularly effective in dealing with limited data availability and shifting distributions. We introduce a novel threshold concept to effectively distinguish between novel and seen classes, optimizing classification

performance. Despite its simplicity, our method achieves superior results, demonstrating significant potential for application in image classification involving novel classes. Future efforts will be directed towards enhancing this methodology, striking a balance between simplicity and effectiveness.

References

[1] Kyohei Atarashi et al. "A deep neural network for pairwise classification: Enabling feature conjunctions and ensuring symmetry". In: *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I 21*. Springer. 2017, pp. 83–95.

[2] John Burkardt. "K-means clustering". In: *Virginia Tech, Advanced Research Computing, Interdisciplinary Center for Applied Mathematics* (2009).

[3] Senqi Cao and Zhongfei Zhang. "Deep hybrid models for out-of-distribution detection". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 4733–4743.

[4] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[5] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ICLR* (2021).

[6] Wen Fu, Li Zhou, and Jie Chen. "Bidirectional matching prototypical network for few-shot image classification". In: *IEEE Signal Processing Letters* 29 (2022), pp. 982–986.

[7] Victor Garcia and Joan Bruna. "Few-shot learning with graph neural networks". In: *arXiv preprint arXiv:1711.04043* (2017).

[8] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[9] Sergey Ioffe and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. pmlr. 2015, pp. 448–456.

[10] Brenden M. Lake et al. "One shot learning of simple visual concepts". In: *Proceedings of the 33th Annual Meeting of the Cognitive Science Society, CogSci 2011, Boston, Massachusetts, USA, July 20-23, 2011*. Ed. by Laura A. Carlson, Christoph Hölscher, and Thomas F. Shipley. cognitivesciencesociety.org, 2011. URL: https://mindmodeling.org/cogsci2011/papers/0601/index.html.

[11] Hongyang Li et al. "Finding task-relevant features for few-shot learning by category traversal". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 1–10.

[12] Wenbin Li et al. "Revisiting local descriptor based image-to-class measure for few-shot learning". In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019, pp. 7260–7268.

[13] Xiaoxu Li et al. "Deep metric learning for few-shot image classification: A Review of recent developments". In: *Pattern Recognition* (2023), p. 109381.

[14] Yuqing Ma et al. "Transductive Relation-Propagation Network for Few-shot Learning." In: *IJCAI*. Vol. 20. 2020, pp. 804–810.

[15] Atif Mehmood et al. "A deep Siamese convolution neural network for multi-class classification of Alzheimer disease". In: *Brain sciences* 10.2 (2020), p. 84.

[16] Erik G. Miller, Nicholas E. Matsakis, and Paul A. Viola. "Learning from One Example through Shared Densities on Transforms". In: *2000 Conference on Computer Vision and Pattern Recognition (CVPR 2000), 13-15 June 2000, Hilton Head, SC, USA*. IEEE Computer Society, 2000, pp. 1464–1471. DOI: 10.1109/CVPR.2000.855856. URL: https://doi.org/10.1109/CVPR.2000.855856.

[17] Romain Mormont, Pierre Geurts, and Raphaël Marée. "Comparison of deep transfer learning strategies for digital pathology". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2018, pp. 2262–2271.

[18] Frank Nielsen and Frank Nielsen. "Hierarchical clustering". In: *Introduction to HPC with MPI for Data Science* (2016), pp. 195–211.

[19] Jo Plested and Tom Gedeon. "Deep transfer learning for image classification: a survey". In: *arXiv preprint arXiv:2205.09904* (2022).

[20] Sachin Ravi and Hugo Larochelle. "Optimization as a model for few-shot learning". In: *International conference on learning representations*. 2016.

[21] Mengye Ren et al. "Meta-learning for semi-supervised few-shot classification". In: *arXiv preprint arXiv:1803.00676* (2018).

[22] Jake Snell, Kevin Swersky, and Richard S. Zemel. "Prototypical Networks for Few-shot Learning". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*. Ed. by Isabelle Guyon et al. 2017, pp. 4077–4087. URL: https://proceedings.neurips.cc/paper/2017/hash/cb8da6767461f2812ae4290eac7cbc42-Abstract.html.

[23] Jake Snell, Kevin Swersky, and Richard S. Zemel. *Prototypical Networks for Few-shot Learning*. 2017. arXiv: 1703.05175 [cs.LG].

[24] Flood Sung et al. *Learning to Compare: Relation Network for Few-Shot Learning*. 2018. arXiv: 1711.06025 [cs.CV].

[25] Oriol Vinyals et al. "Matching networks for one shot learning". In: *Advances in neural information processing systems* 29 (2016).

[26] Ross Wightman. *PyTorch Image Models*. https://github.com/huggingface/pytorch-image-models. 2019. DOI: 10.5281/zenodo.4414861.

[27] Dan Zhang et al. "Multi-view transfer learning with a large margin approach". In: *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2011, pp. 1208–1216.

[28] Wenlu Zhang et al. "Deep model based transfer and multi-task learning for biological image analysis". In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2015, pp. 1475–1484.

[29] Fuzhen Zhuang et al. *A Comprehensive Survey on Transfer Learning*. 2020. arXiv: 1911.02685 [cs.LG].

A Contribution

- Xuezhen Wang: Code Framework & Experiment, Design and Implement the Prototypical Method, Paper Writing
- Yucheng Wang: Code Framework & Experiment, Design and Implement the Revised Prototypical Method, Paper Writing
- Xinchen Zhang: Code & Experiment, Pre-trained ResNet, Paper Writing