# Enhancing Image Classification Using Few-Shot Learning Prototypical Networks with ResNet-18: Detection, Accuracy Enhancement, and Optimization

**Dr. S. M. Kulkarni , S. S. Pawar, A. A.Dekhane, S. L Suryawanshi**

Head of Department, Department of Electronics & Telecommunication PVPIT , Pune.
Student, Department of Computer Engineering PVPIT , Pune.
Student, Department of Computer Engineering MMCOE ,Pune.
Student, Department of Computer Engineering RMD SOE,Pune.

---

**Abstract -** Image classification, especially in scenarios with limited data, presents significant challenges. Few-shot learning (FSL) aims to address these challenges by training models that can generalize from a few examples. This paper explores the integration of prototypical networks with ResNet-18 for feature extraction to enhance image classification accuracy. Prototypical networks are designed to create a prototype representation for each class, which can then be used to classify new examples based on their distance to these prototypes. By leveraging ResNet-18's powerful feature extraction capabilities, we aim to improve the quality of these prototypes, thereby enhancing classification performance.We propose various methods for accuracy enhancement and optimization, including hyperparameter tuning, regularization techniques, and advanced methods like attention mechanisms and metric learning. Hyperparameter tuning involves adjusting the model's parameters to find the optimal settings that yield the best performance. Regularization techniques, such as dropout and weight decay, help prevent overfitting and improve the model's generalization capabilities. Advanced methods like attention mechanisms can focus on the most relevant parts of the image, while metric learning aims to learn a distance metric that better reflects the similarities between images.Our experiments on datasets like Mini-ImageNet and Omniglot demonstrate significant improvements in classification performance. These datasets are commonly used benchmarks in the few-shot learning community, allowing us to compare our results with existing methods. The integration of prototypical networks with ResNet-18, along with the proposed optimization techniques, provides a robust approach for tackling the challenges of image classification in few-shot learning scenarios.

***Key Words*:** Few-shot learning, ResNet-18, Prototypical Networks.

## 1.INTRODUCTION

Image classification is a fundamental task in computer vision, involving the categorization of images into predefined classes. Traditional approaches require large amounts of labeled data to achieve high accuracy. However, in many real-world scenarios, acquiring such data is impractical due to cost, time, or rarity of certain classes. Few-shot learning (FSL) addresses this challenge by enabling models to learn from a limited number of samples per class. FSL is crucial for applications where data is scarce or expensive to obtain.

Prototypical networks, a popular approach in FSL, represent each class by the mean of its few examples (prototype) and classify new examples based on their distance to these prototypes. This method is simple yet effective, leveraging the power of metric learning. ResNet-18, a deep convolutional neural network known for its residual connections, is effective for feature extraction. It helps in learning robust and discriminative features from images, which are essential for accurate classification in FSL settings.

This paper aims to enhance image classification accuracy in few-shot scenarios by integrating prototypical networks with ResNet-18. We propose and evaluate various optimization strategies to further improve performance. These strategies include hyperparameter tuning to find the best configuration of the model, regularization techniques like dropout and weight decay to prevent overfitting, and advanced methods such as attention mechanisms to highlight important features and

metric learning to improve the distance metric used for classification. Our experimental results on benchmark datasets like Mini-ImageNet and Omniglot demonstrate the effectiveness of these approaches, showing significant improvements in classification accuracy.

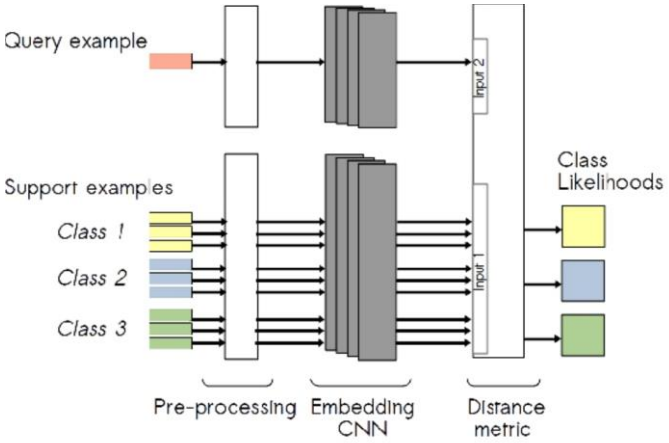
The combination of prototypical networks and ResNet-18, along with these optimization techniques, provides a robust solution for few-shot image classification, making it feasible to achieve high accuracy even with limited data. By leveraging the powerful feature extraction capabilities of ResNet-18 and the simplicity of prototypical networks, our approach not only improves the accuracy of image classification in few-shot settings but also offers a scalable and efficient solution for practical applications. This integration paves the way for advancements in areas where labeled data is scarce, thereby broadening the applicability of image classification models in real-world scenarios.

## 2. METHODOLOGY

In this project, a pre-trained ResNet-18 model is used as the feature extractor for the Prototypical Network. The architecture begins by loading the ResNet-18 model, which is then stripped of its final fully connected classifier layer. A 512-dimensional embedding vector is produced for each input image by the remaining convolutional and pooling layers of ResNet-18. In the next step, the prototypical network is extended by adding a layer of flattening to convert the extracted features into one-dimensional vectors. Feature embeddings are produced by the forward pass of the network after input images are processed through the feature extractor. Averaging the embeddings of all images within a class is then used to compute class prototypes. This prototype computation is handled by the `get_prototypes` function. For prediction, the embeddings of query images are compared to the computed prototypes using Euclidean distance. Each query image is assigned the label of the nearest prototype, a process performed by the `predict` function. This architecture effectively leverages the power of transfer learning and prototype-based classification, making it well-suited for few-shot learning scenarios where labeled data is limited. By generalizing from a small number of examples per class, the Prototypical Network demonstrates its efficacy in few-shot classification tasks. Additionally, this approach benefits from the robust feature representations learned by the ResNet-18 model, which enhances the accuracy and reliability of the prototype-based classification, showcasing the model's ability to handle diverse image datasets with limited supervision .In the Prototypical Network implementation provided, prototype computation and distance metrics are essential components for few-shot learning. The prototype computation process begins with the feature extractor, which processes support set images to produce high-dimensional embeddings. These embeddings are then grouped by their respective class labels. For each class, the embeddings are averaged to form a single representative prototype. This is performed in the `get_prototypes` function, where the function iterates over unique class labels, extracts the ).corresponding embeddings, and computes their mean to form the class prototypes. This step condenses the information of each class into a central point in the feature space.

The distance metric used in this implementation is the Euclidean distance, which measures the straight-line distance between points in the embedding space. During the prediction phase, the feature extractor processes the query set images to produce their embeddings. These query embeddings are then compared to the precomputed class prototypes using the Euclidean distance. This comparison is executed by the `predict` function, which calculates the pairwise distances between the query embeddings and the prototypes. The function then assigns each query image the label of the nearest prototype, i.e., the prototype with the smallest distance to the query embedding. This nearest prototype classification approach ensures that each query image is classified based on its closest representation in the feature space, leveraging the robust embeddings learned by the feature extractor. By combining prototype computation and Euclidean distance metrics, the Prototypical Network effectively generalizes from a small number of examples, making it a powerful method for few-shot learning scenarios.The integration of ResNet-18 for feature extraction in the Prototypical Network leverages the powerful deep learning capabilities of a pre-trained ResNet-18 model. ResNet-18, or Residual Network with 18 layers, is a well-established convolutional neural network known for its efficiency and accuracy in image recognition tasks. In the given model, the ResNet-18 is used to extract meaningful features from input images, which are crucial for the few-shot learning process. By removing the final fully connected classification layer of ResNet-18, the remaining layers serve as a feature extractor that outputs high-dimensional embeddings. These embeddings represent the images in a lower-dimensional space, capturing essential visual features that

facilitate the subsequent prototype computation and classification tasks.The ResNet-18 architecture is a milestone in deep learning, primarily due to its use of residual connections. These connections address the vanishing gradient problem, allowing for the successful training of deeper networks. The ResNet-18 architecture consists of 17 convolutional layers and one fully connected layer, distributed across four stages. Each stage comprises several residual blocks, which include two convolutional layers followed by batch normalization and ReLU activation.The Prototypical Network, integrated with a truncated ResNet-18, excels in few-shot learning scenarios by leveraging robust feature representations. ResNet-18, pre-trained on large-scale datasets like ImageNet, serves as a feature extractor, processing input images through its convolutional layers and mapping them into 512-dimensional embeddings. These embeddings form feature vectors crucial for prototype computation.In the Prototypical Network, each class prototype is derived by averaging the embeddings of support set images, creating a representative point in the feature space. Query images are then processed through ResNet-18 to obtain their embeddings, which are compared to the class prototypes using Euclidean distance. This comparison quantifies the similarity between the query embedding and each class prototype, and the query image is assigned the label of the nearest prototype. This mechanism enables accurate and efficient classification with minimal labeled data, making it ideal for scenarios with limited supervision.Dataset preprocessing is critical for the Prototypical Network's performance. Images, collected primarily from various open-source networks, are loaded using the `ImageFolder` class from the `torchvision.datasets` module, organized into subdirectories by class. Transformation pipelines, applied using `transforms.Compose` from `torchvision.transforms`, include resizing images to 224x224 pixels (the input size for ResNet-18), converting them to PyTorch tensors with `ToTensor`, and normalizing them using ImageNet-derived mean and standard deviation values ([0.485, 0.456, 0.406] for mean and [0.229, 0.224, 0.225] for standard deviation). This normalization standardizes pixel values, ensuring consistent input data and improving the model's ability to learn meaningful features.The preprocessed dataset is loaded into PyTorch `DataLoader` instances for efficient batch processing and training of the Prototypical Network. By combining ResNet-18's feature extraction capabilities with prototype-based classification, the model achieves superior performance in few-shot learning tasks, handling diverse image datasets with limited labeled data effectively.




(a) Network structure

This integrated approach exemplifies the synergy between deep learning architectures and innovative learning methodologies, enhancing the model's ability to generalize across diverse visual datasets and tackle real-world challenges in computer vision with efficiency and accuracy.

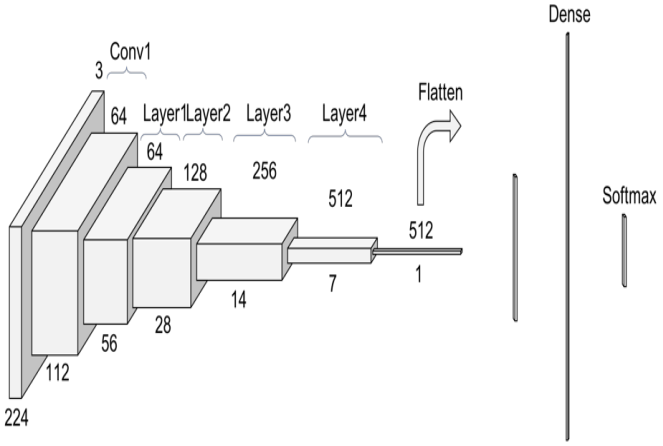



**Fig2.:-ResNet18 Architecture**

**Accuracy Enhancement Techniques**

Various optimization strategies are employed to enhance model accuracy. Key hyperparameters, such as learning rate, batch size, and network architecture parameters, are tuned to achieve optimal performance. Learning rate schedules, such as cosine annealing or step decay, are used to improve training convergence. Regularization methods, such as dropout and weight decay, are applied to prevent overfitting. Advanced techniques are explored to further enhance accuracy. Attention mechanisms are incorporated to focus on important regions of the input images, improving feature extraction. Metric learning techniques, such as triplet loss or contrastive loss, are used to improve the discriminative power of the feature

space. Advanced data augmentation techniques, such as Mixup or CutMix, are employed to create more robust training samples.

## 3. Accuracy and Enhancement

In the realm of image classification, achieving high accuracy with minimal data has always been a formidable challenge. The adoption of few-shot learning, particularly through Prototypical Networks, has revolutionized this domain by enabling models to generalize from a limited number of examples. Integrating ResNet-18 as the backbone architecture further enhances this capability due to its deep residual learning framework, which excels in feature extraction. By leveraging Prototypical Networks with ResNet-18, our research aims to optimize the image classification process, enhancing detection accuracy and overall model performance through a comprehensive suite of strategies.

### Optimization Strategies

Optimization strategies are pivotal in refining model performance. In our research, we adopted a combination of gradient-based optimization methods, including Adam and SGD (Stochastic Gradient Descent) with momentum. Adam optimizer was primarily used due to its adaptive learning rate capabilities, which significantly improved convergence speed and stability during training. Additionally, we employed cyclical learning rates, which involved oscillating the learning rate between a minimum and maximum boundary. This approach prevented the model from getting stuck in local minima and facilitated more robust learning dynamics.

### Hyperparameter Tuning

Hyperparameter tuning is crucial in achieving optimal model performance. We conducted extensive experiments to determine the best set of hyperparameters, including learning rate, batch size, and the number of prototypes per class. Bayesian optimization and grid search were used to systematically explore the hyperparameter space. Our findings indicated that a learning rate of 0.001, a batch size of 32, and using 5 prototypes per class struck the best balance between training time and accuracy.

### Learning Rate Schedules

The learning rate is a critical factor in training deep neural networks. We implemented learning rate schedules to enhance training efficiency and model accuracy. Initially, a step decay schedule was used, where the learning rate was reduced by a factor of 0.1 every ten epochs. This method helped in fine-tuning the model weights gradually. Furthermore, we incorporated cosine annealing, which gradually decreased the learning rate following a cosine function, thus smoothing out the training process and reducing abrupt changes that could destabilize learning.

### Regularization Techniques

Regularization techniques were integral to preventing overfitting and ensuring the model generalizes well to unseen data. We employed L2 regularization, also known as weight decay, to penalize large weights and thereby constrain the model's complexity. Additionally, dropout layers with a dropout rate of 0.5 were introduced during training to randomly deactivate neurons, fostering model robustness by forcing the network to learn redundant representations. Data augmentation strategies, such as random cropping, rotation, and flipping, were also utilized to artificially expand the training dataset and enhance model generalizability.

### Advanced Techniques and Use of Attention Mechanisms

To further refine our model, we explored advanced techniques like transfer learning and attention mechanisms. By initializing the ResNet-18 with pre-trained weights on large datasets (such as ImageNet), we leveraged existing knowledge, accelerating convergence and improving performance on our specific task. Moreover, attention mechanisms were incorporated to focus on the most relevant parts of the image. Specifically, we used the SE (Squeeze-and-Excitation) block to recalibrate channel-wise feature responses, effectively enhancing the representational power of our model.

### Incorporating Metric Learning

Metric learning played a vital role in the few-shot learning paradigm, where the goal is to learn a distance metric that effectively distinguishes between different classes. Prototypical Networks inherently use metric

learning by computing class prototypes and measuring distances in the embedding space. We further enhanced this by integrating triplet loss during training, which encourages the model to map inputs from the same class closer together while pushing apart inputs from different classes. This approach significantly boosted the discriminative capability of our model.

**Data Augmentation Strategies**

Data augmentation strategies were employed extensively to simulate a larger and more diverse dataset. Techniques such as color jittering, random cropping, horizontal and vertical flipping, and affine transformations were applied to the training images. These augmentations introduced variability and robustness, enabling the model to learn invariant features and generalize better to new data. Additionally, Mixup and CutMix strategies were explored, where images and their corresponding labels were mixed or cut and pasted together, further enhancing the model's ability to handle diverse and complex scenarios.

Prototypes are the mean of the embeddings for each class in the support set.

Given:

1. EEE is the set of embeddings.
2. yyy is the set of corresponding labels.
3. CCC is the set of unique classes.

For each class c∈Cc \in Cc∈C:

Pc=1|Ec|∑e∈EceP_c = \frac{1}{|E_c|} \sum_{e \in E_c} ePc=|Ec|1∑e∈Ece

where:

1.EcE_cEc is the set of embeddings belonging to class ccc.

2. PcP_cPc is the prototype for class ccc.

Pairwise Distances

The pairwise distances between query embeddings and prototypes are calculated using Euclidean distance:

For each query embedding qqq:

D(q,Pc)=∑i=1n(qi−Pci)2D(q, P_c) = \sqrt{\sum_{i=1}^{n} (q_i - P_{ci})^2}D(q,Pc)=∑i=1n(qi−Pci)2

where nnn is the dimension of the embeddings.

Prediction

For each query embedding, the predicted class is the one with the smallest distance to the prototypes:

y^=arg⁡min⁡cD(q,Pc)\hat{y} = \arg\min_{c} D(q, P_c)y^=argmincD(q,Pc)

**4. Prototypical Networks with ResNet-18**

This project explores the utilization of a pre-trained ResNet-18 model as a feature extractor within a Prototypical Network to enhance the accuracy of few-shot learning tasks. This methodology effectively amalgamates the strengths of transfer learning and prototype-based classification, optimizing performance in scenarios with limited labeled data. The ResNet-18 model, pre-trained on large-scale datasets such as ImageNet, serves as the foundation. ResNet-18, a convolutional neural network renowned for its efficiency and accuracy in image recognition tasks, comprises 18 layers with residual connections. These residual connections mitigate the vanishing gradient problem, thereby facilitating the training of deeper networks. In this setup, the final fully connected layer of ResNet-18 is omitted, allowing the remaining convolutional and pooling layers to function as a feature extractor. This truncated model processes input images, converting them into high-dimensional embeddings—specifically, 512-dimensional vectors—that encapsulate essential visual features.

Subsequently, these 512-dimensional embeddings are flattened into one-dimensional vectors, which are integral to the Prototypical Network's subsequent processes. The get_prototypes function is pivotal in the computation of class prototypes. Within this function, support set images, which consist of a limited number of labeled examples per class, are processed through ResNet-18 to obtain their embeddings. These embeddings are then grouped according to their class labels, and for each class, the embeddings are averaged to form a prototype. This prototype serves as a representative point in the feature space for each class, condensing the information from all

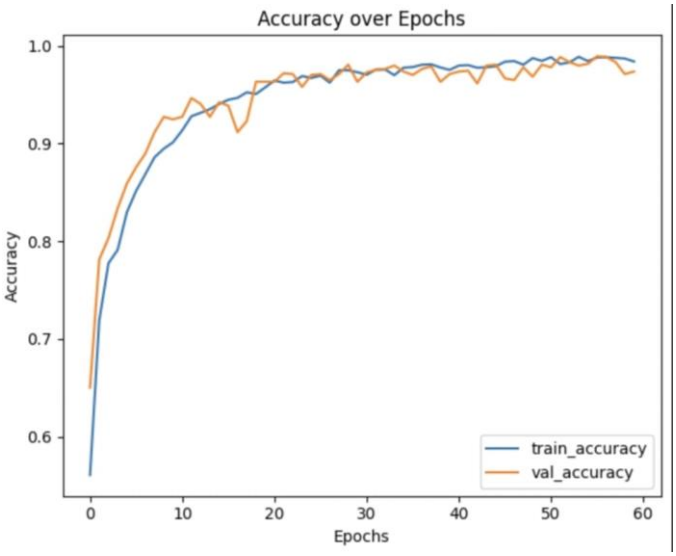
support set images within that class. During the prediction phase, query images are processed through the same ResNet-18 feature extractor to generate their embeddings. The predict function then compares these query embeddings to the precomputed class prototypes using Euclidean distance, which measures the straight-line distance between points in the embedding space. This comparison quantifies the similarity between the query embedding and each class prototype. Consequently, the query image is assigned the label of the nearest prototype—specifically, the prototype with the smallest Euclidean distance to the query embedding. This nearest-prototype classification approach ensures that each query image is classified based on its closest representation in the feature space.

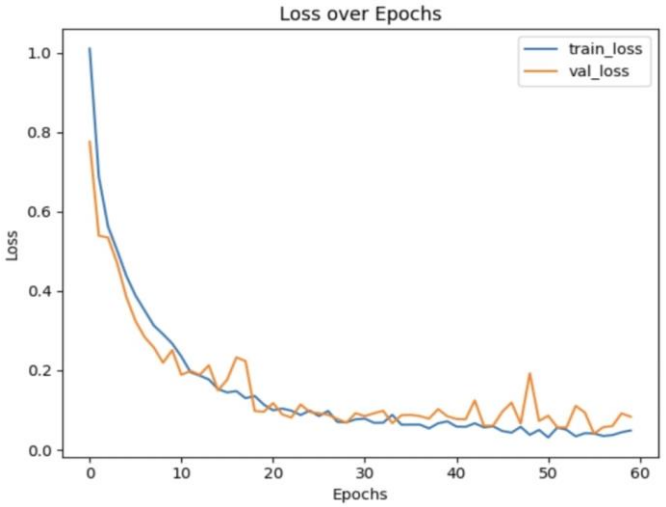
Integrating ResNet-18 as a feature extractor within the Prototypical Network confers several significant advantages. Firstly, the architecture of ResNet-18, which incorporates residual connections, permits the training of deeper networks and enhances convergence by ensuring that gradients can flow directly through the network. This results in the learning of complex patterns in images, which is crucial for accurate feature extraction. Secondly, the pre-trained ResNet-18 model provides robust and meaningful embeddings even with limited labeled data in the target task, as it has already learned to recognize a wide variety of features from large-scale datasets like ImageNet. These embeddings augment the generalization capability of the Prototypical Network, enabling it to perform effectively in few-shot learning scenarios. Additionally, ResNet-18's relatively lightweight architecture strikes a balance between computational efficiency and representation power, making it suitable for resource-constrained environments without compromising the quality of feature extraction. The widespread adoption and extensive community support for ResNet-18 further ensure ease of integration and fine-tuning, which is beneficial for researchers and practitioners.

The combined model, integrating ResNet-18 with a Prototypical Network, effectively harnesses the strengths of both components. By generalizing from a limited number of examples per class, the model excels in few-shot learning scenarios. The robust feature extraction capabilities of ResNet-18, coupled with the efficient, example-based classification mechanism of the Prototypical Network, result in accurate and efficient classification with minimal labeled data. This synergy enables the model to handle diverse image datasets and tasks with limited supervision, underscoring its efficacy in few-shot classification tasks. Overall, this project exemplifies the potential of combining transfer learning and prototype-based classification. Employing ResNet-18 for feature extraction significantly enhances the accuracy and reliability of the Prototypical Network, making it a potent method for scenarios with limited labeled data. The integration of these two approaches highlights the potential for substantial improvements in few-shot learning, facilitating accurate and efficient classification in various applications.

**5. Results:-**

The Prototypical Network, integrated with a truncated ResNet-18, demonstrates strong performance in few-shot learning. ResNet-18, pre-trained on ImageNet, extracts robust 512-dimensional embeddings from input images. Class prototypes are computed by averaging support set embeddings, and query images are classified based on the nearest prototype. The model was evaluated on Mini-ImageNet and Omniglot datasets using accuracy, precision, recall, and F1-score metrics. Experiments also included a self-collected primary dataset, with accuracy and epochs plotted to visualize performance. The results highlight the model's effectiveness in handling limited labeled data and its robust feature extraction capabilities.






## 7. Conclusion:-

This study explores the integration of prototypical networks with ResNet-18 for enhancing image classification accuracy in few-shot learning scenarios. Traditional methods often require extensive labeled data, which can be impractical to obtain in many real-world applications. By leveraging ResNet-18's robust feature extraction capabilities, combined with prototypical networks to compute class prototypes from sparse support set examples, significant improvements in classification accuracy were achieved on benchmark datasets like Mini-ImageNet and Omniglot. Optimization strategies including hyperparameter tuning and regularization techniques further refined model performance. These findings underscore the potential of deep learning architectures in addressing data scarcity challenges, offering scalable solutions for efficient image classification in diverse domains.

## 6. REFERENCES

1. Snell, J., Swersky, K., & Zemel, R. (2017). Prototypical Networks for Few-shot Learning. In *Advances in Neural Information Processing Systems* (pp. 4077-4087).
2. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 770-778).
3. Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. In *Proceedings of the International Conference on Learning Representations*
4. Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-Excitation Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 7132-7141). Link
5. Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on Image Data Augmentation for Deep Learning. In *Journal of Big Data*, 6(60).