國立陽明交通大學

NATIONAL YANG MING CHIAO TUNG UNIVERSITY

[EEEE30026] Data Science

Final Project - Model Compression and Few-shot Learning for Pokémon Classification

---

Xiang-Jun, You, Yu-Che, Chang

Student ID : 311511056, 311513058

davidyou0121.ee11@nycu.edu.tw, anthony0304.ee11@nycu.edu.tw

National Yang Ming Chiao Tung University

Contents

Abstract

We present a combination of few-shot learning and model compression techniques for Pokémon classification. Few-shot learning enables effective learning from limited examples, while model compression reduces network complexity for efficient deployment on resource-constrained devices. The proposed method involves training a teacher model using supervised contrastive learning and distilling its knowledge to a smaller student model. The student model is fine-tuned and using metric learning during testing. Pokémon images are crawled from the Naver search engine. Experimental results show promising clustering and compression effects, with scope for further improvements through dataset quality enhancement and the increase of model parameters.

1 Introduction

Few-shot learning is a powerful technique that enables neural networks to learn from a small set of examples. This characteristic makes it particularly suitable for scenarios where training data is limited or scarce. Traditional deep learning approaches often require a large amount of labeled data to achieve good performance. However, in real-world situations, obtaining such data can be challenging and expensive.

On the other hand, model compression techniques have gained significant attention in recent years. These techniques aim to reduce the parameters and complexity of neural networks while preserving their performance. By compressing the model, we can make it more efficient and suitable for deployment on devices with limited computational resources, such as mobile phones or embedded systems. This is especially crucial in applications where real-time inference or low-power consumption is required.

In this project, we propose a novel approach that combines the strengths of both few-shot learning and model compression for the task of Pokémon classification. Pokémon is a popular franchise with a vast number of unique characters, and accurately classifying them is a challenging problem. By leveraging few-shot learning, we can train a neural network to recognize new Pokémon species based on a small number of training examples. This allows us to generalize well to unseen Pokémon species, even with limited labeled data.

2 Related Work

2.1 Few-shot learning

Few-shot learning has been extensively studied in the field of computer vision and machine learning. It aims to enable machine learning models to generalize from a limited number of labeled examples, typically referred to as the support set, to classify or recognize new

instances, known as the query set. Several approaches have been proposed to tackle this problem.

One popular approach is meta-learning, which involves training a meta-model to learn how to adapt its parameters quickly to new tasks with few examples. This is typically done by training the model on a large number of tasks, each consisting of a support set and a query set. The model learns to extract task-specific knowledge from the support set and generalize it to the query set.

Another approach is metric learning, which focuses on learning a similarity metric that can compare instances and determine their proximity in the feature space. By leveraging the similarity between support set examples and query set examples, the model can make predictions based on nearest-neighbor or prototype-based methods.

In this project, we want to using supervised contrastive learning and metric learning to do few-shot learning. Our goal is to improve few-shot learning performance and explore the potential of these techniques in practical applications.

2.2 Model compression

Model compression techniques aim to reduce the size and computational complexity of deep neural networks while preserving their performance. This is crucial for deploying models on devices with limited resources, such as mobile devices or embedded systems.

One common technique for model compression is pruning, which involves removing unnecessary connections or parameters from the network. Pruning methods can be based on magnitude pruning, where weights with low magnitudes are pruned, or structured pruning, where entire filters or channels are pruned. Pruning can be performed during training or as a post-processing step.

Another technique is quantization, which reduces the precision of weights and activations from floating-point representation to lower bit-width fixed-point representation. This reduces the memory footprint and computation cost of the network without significant loss in performance.

Knowledge distillation is another popular method for model compression. It involves training a smaller and more lightweight student model to mimic the behavior of a larger and more complex teacher model. The student model learns from the soft targets produced by the teacher model, which are typically the logits or class probabilities. This knowledge transfer helps the student model achieve similar performance to the teacher model while having a smaller size.

Other model compression techniques include low-rank factorization, parameter sharing, and network architecture design specifically optimized for efficiency.

By combining few-shot learning and model compression, we aim to develop a Pokémon classification model that is efficient, accurate, and suitable for real-time scenarios with limited training data and computational resources.

3 Method

Our method consists of two stages: the training stage and the testing stage.

During the training stage, we first use cifar10 to let model to learn how to achieve good clustering effects in the latent space. Afterward, we use the Pokémon images we crawled to do few-shot learning in testing stage.

3.1 Training stage

During the training stage, we first employ supervised contrastive learning (SCL [1]) to train our teacher model, bringing features with the same label closer together and pushing apart features with different labels, enabling effective clustering in the latent space.

$$\mathcal{L}_{\text{SCL}} = \sum_{i=1}^{2N} \frac{-1}{2N_{\tilde{y}_i} - 1} \sum_{j=1}^{2N} l_{[i \neq j]} \cdot l_{[\tilde{y}_i = \tilde{y}_j]} \cdot \log \frac{\exp\left(z_i \cdot z_{j(i)}/\tau\right)}{\sum_{k=1}^{2N} l_{[k \neq i]} \cdot \exp\left(z_i \cdot z_{j(k)}/\tau\right)} \tag{1}$$

Next, we distill the knowledge from the teacher model to the student model(Figure 1). The loss function (equation2) used includes KL Divergence loss (to learn from the teacher) and supervised contrastive loss (to learn from the ground truth), α is our hyperparameter, we set $\alpha = 0.1$ in our experiment.

$$\text{L}_{\text{student}} = \alpha L_{KD} + (1 - \alpha) L_{SCL} \tag{2}$$

We also use data augmentation in our method, which is a widely used technique that involves applying various transformations or modifications to the original data to create new, synthetic training samples. These transformed samples can help improve the generalization and robustness of the model by introducing variations in the data.

3.2 Testing stage

We utilize the student model trained in the previous stage for few-shot learning using metric learning(Figure 2). We first fine-tune the student model using the support set, and then evaluate its performance on the query set during testing. Based on which category in average of the support set the query set is closest to, we determine its classification as that particular category.

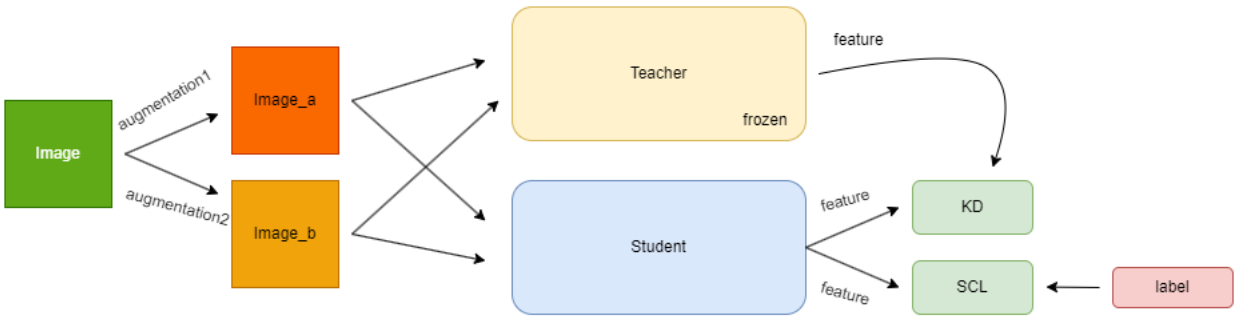

Figure 1: The framework to train our student model.

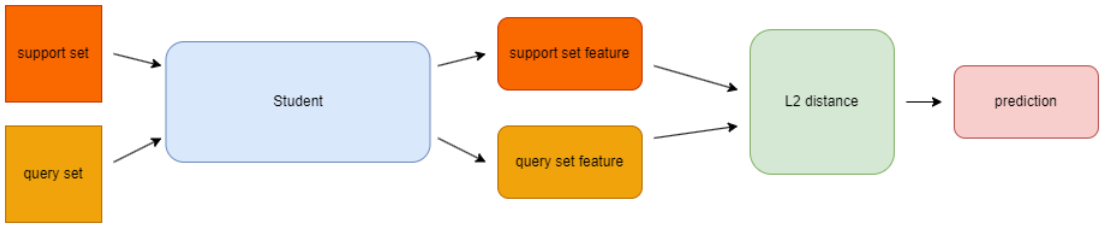

Figure 2: how we do few-shot learning.

3.3 Crawl images

We use the Naver image search engine(Figure3) to crawl our Pokémon images. First, we utilize ChatGPT to list 100 Pokémon names in English. Then, we save those names as a TXT file and write a program to read the names sequentially, searching for them on Naver and downloading ten images for each name.

Figure 3: The website we used to crawl images.

4 Experiment

4.1 Training stage

In training stage, we use cifar10 to be our dataset and resize to 28x28. Our teacher model is Resnet18 and student model is 3 layers CNN model.

In Table1 and Figure4, it can be observed that our teacher model has achieved excellent

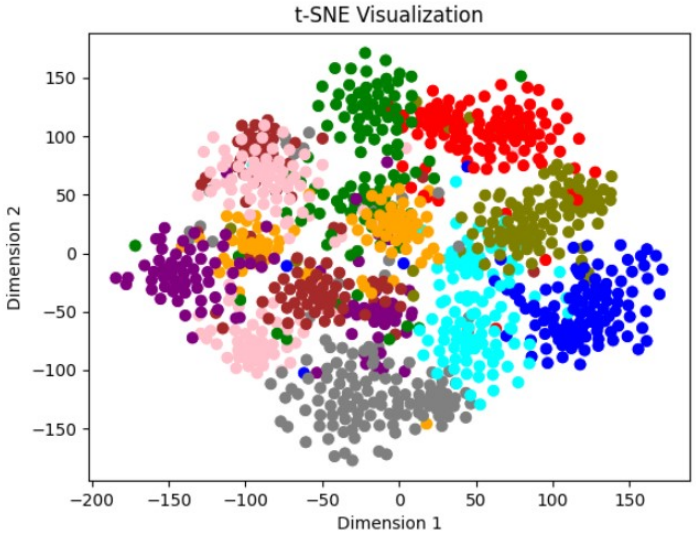

Figure 4: t-SNE visualization of our teacher model after training.

clustering performance in the latent space during the training stage. This allows the student model to learn valuable features effectively.

In Table2 and Figure5, it can be observed that our student model does not exhibit the same strong clustering ability in the latent space as the teacher model. We believe the reason for this lies in the excessive compression of the model parameters, which hinders the distillation of knowledge.

Training time	Epochs	Batch size	Optimizer	Weight decay
4.44 hr	1000	512	AdamW	1e-4

Learning rate scheduler	**Teacher parameter**	**Best accuracy**	**Best loss**
cosine anealing (1e-4 1e-6)	**11.17M**	**0.9104**	**5.1558**

Table 1: Teacher model training setting and result.

4.2 Testing stage

In testing stage, our dataset is use Pokémon images ,and resize to 28x28. Our few-shot setting is 5way-5shot, including 600 tasks, with each task consisting of 5 categories. Each category contains 10 images, and we randomly select 5 images as the support set and another 5 images as the query set.

In Table3 and Figure6, it can be observed that our student model demonstrates effective

Training time	Epochs	Batch size	Optimizer	Weight decay
3.33 hr	1000	512	AdamW	1e-4

Learning rate scheduler	**Student parameter**	**Best accuracy**	**Best loss**
cosine anealing (1e-4 1e-6)	**0.7M**	**0.6811**	**3.1417**

Table 2: Student model training setting and result.

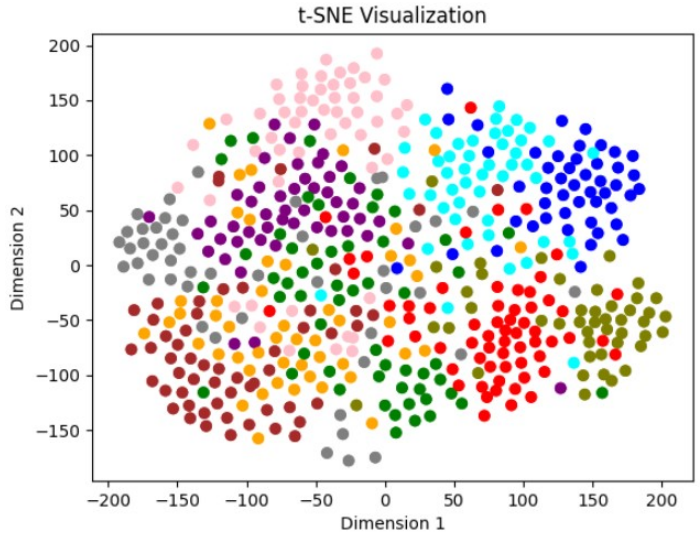

Figure 5: t-SNE visualization of our student model after training.

clustering of the support set, and due to a significant reduction in the number of model parameters, the training time is also considerably faster.

In Table4 and Figure7, it can be observed that our student model fails to accurately classify the query set, resulting in the inability to achieve precise improvements in accuracy. We analyze that the reason behind this is that the clawed images are with too much noise, and we did not train the student model sufficiently during the training stage. In the future, increasing the model parameters may potentially enhance the accuracy.

Finetuning time	Epochs	optimizer	Learning rate scheduler
10 min	500	AdamW	cosine anealing (1e-4 1e-6)

Table 3: student model finetuning setting on support set of Pokémon images.

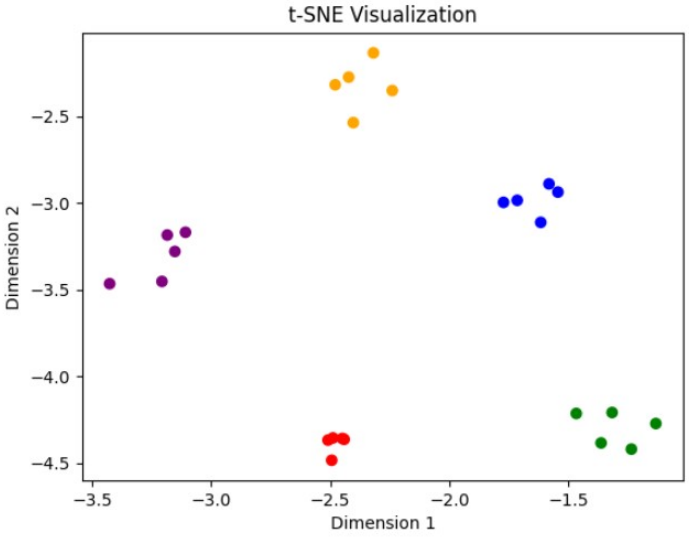

Figure 6: t-SNE visualization of our student model on support set of Pokémon images.

accuracy
0.538

Table 4: student model final result on query set of Pokémon images.

5 Conclusion

We use contrastive learning and metric learning for few-shot learning. We employ knowledge distillation to compress the model, resulting in a 94 percentage reduction in the parameters of ResNet18. However, the final results were not very satisfactory, and we believe the following factors contributed to this: there is too much noise in the Pokémon images crawled from the internet, which prevents the accuracy from improving.

To improve in the future, we suggest the following methods:

1. More powerful models: Consider using deeper or more complex models such as ResNet50, VIT, etc., to enhance the model's learning capability and expressive power.

2. Data cleaning: Perform data cleaning on the crawled images to remove noise or unnecessary pictures. This can be achieved through techniques like image processing, filtering algorithms, or manual verification.

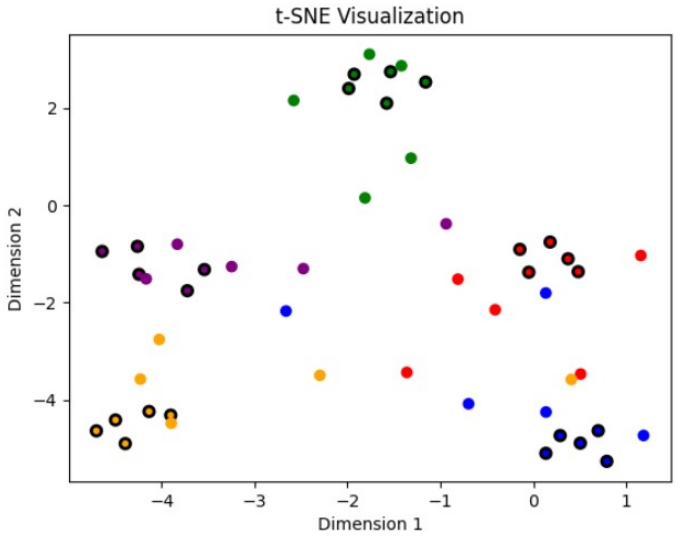

Figure 7: t-SNE visualization of our student model on support and query set of Pokémon images. The black frame dots in the figure represent the support set, others are query set.

References

[1] Prannay Khosla et al. "Supervised Contrastive Learning". In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 18661–18673. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf.