Reimplementation of SimpleShot

Kevin Chen, Sophia Wang, Katherine Wei, Anya Yerramilli
https://github.com/anya-yerramilli/cs4782_final_project

1. Introduction

Few-shot learning requires models to recognize new object classes using only a few labeled examples per class. Traditionally, these tasks have been solved by applying meta-learning on features extracted by a CNN and then applying a nearest neighbor classifier. However, the paper *SimpleShot: Revisiting Nearest-Neighbor Classification for Few-Shot Learning* [1] by Wang, Chao, Wienberger, and Maaten proposes a new model called SimpleShot that is not as complex as meta-learning but produces comparable results. This new model applies two feature transformations, L2-normalization and centering, on the feature space before applying a nearest-neighbor classifier.

2. Chosen Result

We aimed to reproduce the 5-way 1-shot and 5-way 5-shot accuracies of SimpleShot when classifying images in the dataset *mini*ImageNet [2]. (c.f. Figure 2 of the Appendix for the explicit results.) The CNNs used for reimplementation were a standard 4-layer convolutional network (Conv-4) [3], residual networks (ResNet-10/18) [4], and a wide residual network (WRN-28-10) [5]. We also tested three feature transformations: unnormalized (UN), L2-normalization (L2N), and centering with L2-normalization (CL2N). We chose to replicate these results to test how well SimpleShot can perform on few-shot learning. These results are important to the paper because they test the paper's claim that the feature transformations used in SimpleShot can yield comparable results to more traditional few-shot learners that use meta-learning.

3. Methodology

For reimplementing SimpleShot, we decided to focus on one of the datasets presented in the original paper, *mini*ImageNet. This dataset has 100 classes and 600 examples per class. Following the structure from the paper, we split the dataset into 64 base training classes, 16 validation classes, and 20 test classes. Images were also resized to be 84 x 84 pixels, and we used center cropping.

Feature Extraction and Transformation

SimpleShot trains a CNN given by $f_\theta(I) \in \mathbb{R}^D$ to extract features from an image I. We implemented multiple CNNs for feature extraction: Conv-4, ResNet-10/18, and WRN-28-10. We followed [3], [4], and [5] for implementing those CNNs, as the paper suggests. We then trained this CNN for few-shot classification by having a linear classifier $W \in \mathbb{R}^{D \times A}$, where A is the number of base classes in the test dataset, and minimizing the loss $\mathcal{L}(W,\theta) = \sum_{(I,y)} l(W^T f_\theta(I), y)$, where the sum goes over all images I with label y in the test dataset and l is the cross-entropy loss. During inference for few-shot classification with a support dataset, we pass the images through the CNN to get its features and then apply one of the following feature transformations:

1. **Unnormalized (UN):** Raw features without any normalization.
2. **L2-Normalization (L2N):** Normalize the feature vector x to unit length via $x \leftarrow x / ||x||_2$.
3. **Centering + L2-Normalization (CL2N):** Before applying L2-normalization, first center the feature vector $x \leftarrow x - \bar{x}$, , where $\bar{x}$ is the mean of all features vectors of the support dataset.

To classify a test image, we apply a 1-nearest neighbor classifier with Euclidean distance on its transformed features. For 1-shot classification, the image is labeled with the class of the image in the support data that it is closest to (in the feature space). For multi-shot classification, we do the same approach but use the centroids (i.e. average) of the feature vectors from each class to perform 1-nearest neighbor classification.

Training and Testing

We trained SimpleShot for 90 epochs and used SGD on the loss function $\mathcal{L}(W,\theta) = \sum_{(I,y)} l(W^T f_\theta(I), y)$ explained in the previous section. The model starts with a learning rate of 0.1, then is reduced by a factor of 10 at epochs 45 and 66. We performed early stopping according to the 1-shot 5-way accuracy of SimpleShot with CL2N on the validation dataset. For testing, the metrics that we used to evaluate SimpleShot were given by the percent accuracy for labeling images in the test dataset. We recorded the accuracy of SimpleShot across the three feature transformations UN, L2N, and CL2N.

Modifications

The only modifications we made were that we only replicated the results for a subset of the CNNs used by the paper and only tested SimpleShot on one of the three datasets used in the paper. We did so because of the lack of time to train and implement (especially for the deeper and more complex CNNs). Especially considering how the paper does not fully explain the specifics of each CNN but defer to other full-length papers for reimplementation, we did not have time to read those papers and implement those CNNs. Moreover, we felt that the CNNs we did choose were enough to show the original paper's main goal of how well SimpleShot performs compared to other state-of-the-art few-shot learners.

4. Results and Analysis

We created this graph for a run of the model using the Conv-4 CNN for feature extraction and using CL2N for feature transformation. Training accuracy rose to ~80% by epoch 45, and jumped to > 95% after the LR drop. Validation accuracy didn't jump as significantly, hovering at around 40%, and peaking at 43.17% after the LR drop. These results make sense because our model is really good at learning the classes it sees in training, but not as good at learning novel classes.

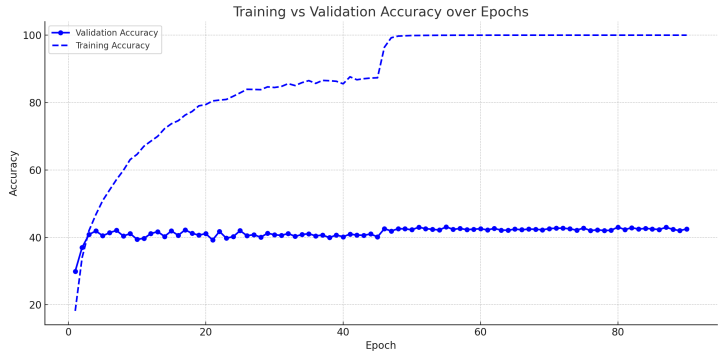

Figure 1: Training and validation accuracy over epochs under Conv-4

Network	One Shot	Five Shots	Network	One Shot	Five Shots
Conv-4 (UN)	30.76% ± 0.15%	58.41 ± 0.17%	ResNet-10 (UN)	50.70% ± 0.20%	70.09% ± 0.16%
Conv-4 (L2N)	43.53% ± 0.18%	61.09% ± 0.17%	ResNet-10 (L2N)	52.41% ± 0.20%	70.96% ± 0.16%
Conv-4 (CL2N)	44.65% ± 0.18%	61.19% ± 0.17%	ResNet-10 (CL2N)	54.84% ± 0.20%	70.92% ± 0.16%
WRN (UN)	47.49% ± 0.20%	72.37% ± 0.15%	ResNet-18 (UN)	48.58% ± 0.20%	69.91% ± 0.16%
WRN (L2N)	55.08% ± 0.15%	73.31% ± 0.15%	ResNet-18 (L2N)	53.08% ± 0.20%	71.10% ± 0.16%
WRN (CL2N)	56.32% ± 0.19%	73.56% ± 0.15%	ResNet-18 (CL2N)	55.71% ± 0.20%	71.46% ± 0.16%

Our results are shown in the above table. Comparing them to the original paper's results shown in Figure 2 of the Appendix, we see that we achieved slightly lower results. This discrepancy could be due to how

the descriptions on how to implement the CNNs were vague and deferred to other papers, which made it more challenging to exactly recreate each CNN. But even with inferior performance, our reimplemented results still reinforce the findings of the original paper [1] that centering and L2-normalization significantly improve few-shot classification performance, as we see how accuracy from the table increases as we go from UN to L2N to CL2N. Also, our results are still on par with some produced by the other few-shot learners that use meta-learning as shown in Figure 2. This again supports the paper's findings of how SimpleShot can be just as good as these more traditional few-shot learning algorithms. In particular, these comparable performances are critical to the broader research area of few-shot learning since our results show that we can use a much simpler piece of architecture in SimpleShot than few-shot learners using meta-learning but still produce similar performances.

5. Reflections

Our results support the hypothesis that the SimpleShot, when paired with proper feature normalization, such as CL2N, can achieve comparable performance to meta-learning approaches in few-shot learning tasks. The SimpleShot methods provide significant advantages in simplicity compared to meta-learning. Future work in few-shot classification can explore integrating more advanced feature embeddings, investigating the impact of different CNNs, and combining SimpleShot with lightweight fine-tuning or adaptation techniques to enhance performance beyond meta-learning techniques.

References:

[1] Wang, Y., Chao, W.-L., Weinberger, K. Q., & van der Maaten, L. (2019). SimpleShot: Revisiting nearest-neighbor classification for few-shot learning. arXiv. https://arxiv.org/abs/1911.04623
[2] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra, et al. Matching networks for one shot learning. arXiv. https://arxiv.org/abs/1606.04080
[3] J. Snell, K. Swersky, and R. Zemel. Prototypical networks for few-shot learning. arXiv. https://arxiv.org/abs/1703.05175
[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. arXiv. https://arxiv.org/abs/1512.03385
[5] Zagoruyko, S., & Komodakis, N. (2016). Wide Residual Networks. arXiv. https://arxiv.org/abs/1605.07146

A. Appendix: Other figures

Figure 2: Results of SimpleShot compared to other state-of-the-art few-shot classifiers. The table reports the accuracy of each model on 1-shot and 5-shot classification on miniImageNet along with its 95% confidence interval. SimpleShot results are shown in blue and are the ones we aimed to replicate.

Approach	Network	One shot	Five shots
Meta LSTM [26]	Conv-4	43.44 ± 0.77	60.60 ± 0.71
MatchingNet [34]	Conv-4	43.56 ± 0.84	55.31 ± 0.73
MAML [4]	Conv-4	48.70 ± 1.84	63.11 ± 0.92
LLAMA [10]	Conv-4	49.40 ± 1.83	–
ProtoNet [30]	Conv-4	49.42 ± 0.78	68.20 ± 0.66
Reptile [23]	Conv-4	49.97 ± 0.32	65.99 ± 0.58
PLATIPUS [5]	Conv-4	50.13 ± 1.86	–
mAP-SSVM [32]	Conv-4	50.32 ± 0.80	63.94 ± 0.72
GNN [6]	Conv-4	50.33 ± 0.36	66.41 ± 0.63
RelationNet [31]	Conv-4	50.44 ± 0.82	65.32 ± 0.70
Meta SGD [18]	Conv-4	50.47 ± 1.87	64.03 ± 0.94
MTNet [17]	Conv-4	51.70 ± 1.84	–
Qiao *et al.* [25]	Conv-4	54.53 ± 0.40	67.87 ± 0.20
FEAT [36]	Conv-4	**55.15 ± 0.20**	**71.61 ± 0.16**
SimpleShot (UN)	Conv-4	33.17 ± 0.17	63.25 ± 0.17
SimpleShot (L2N)	Conv-4	48.08 ± 0.18	66.49 ± 0.17
SimpleShot (CL2N)	Conv-4	49.69 ± 0.19	66.92 ± 0.17
MAML [4]†	ResNet-18	49.61 ± 0.92	65.72 ± 0.77
Chen *et al.* [2]	ResNet-18	51.87 ± 0.77	75.68 ± 0.63
RelationNet [31]†	ResNet-18	52.48 ± 0.86	69.83 ± 0.68
MatchingNet [34]†	ResNet-18	52.91 ± 0.88	68.88 ± 0.69
ProtoNet [30]†	ResNet-18	54.16 ± 0.82	73.68 ± 0.65
Gidaris *et al.* [8]	ResNet-15	55.45 ± 0.89	70.13 ± 0.68
SNAIL [21]	ResNet-15	55.71 ± 0.99	68.88 ± 0.92
Bauer *et al.* [1]	ResNet-34	56.30 ± 0.40	73.90 ± 0.30
adaCNN [22]	ResNet-15	56.88 ± 0.62	71.94 ± 0.57
TADAM [24]	ResNet-15	58.50 ± 0.30	76.70 ± 0.30
CAML [15]	ResNet-12	59.23 ± 0.99	72.35 ± 0.71
SimpleShot (UN)	ResNet-10	54.45 ± 0.21	76.98 ± 0.15
SimpleShot (L2N)	ResNet-10	57.85 ± 0.20	78.73 ± 0.15
SimpleShot (CL2N)	ResNet-10	60.85 ± 0.20	78.40 ± 0.15
SimpleShot (UN)	ResNet-18	56.06 ± 0.20	78.63 ± 0.15
SimpleShot (L2N)	ResNet-18	60.16 ± 0.20	79.94 ± 0.14
SimpleShot (CL2N)	ResNet-18	**62.85 ± 0.20**	**80.02 ± 0.14**
Qiao *et al.* [25]	WRN	59.60 ± 0.41	73.74 ± 0.19
MatchingNet [34]♯	WRN	64.03 ± 0.20	76.32 ± 0.16
ProtoNet [30]♯	WRN	62.60 ± 0.20	79.97 ± 0.14
LEO [29]	WRN	61.76 ± 0.08	77.59 ± 0.12
FEAT [36]	WRN	**65.10 ± 0.20**	**81.11 ± 0.14**
SimpleShot (UN)	WRN	57.26 ± 0.21	78.99 ± 0.14
SimpleShot (L2N)	WRN	61.22 ± 0.21	81.00 ± 0.14
SimpleShot (CL2N)	WRN	63.50 ± 0.20	80.33 ± 0.14