Explorating Transfer Learning and Semi-Supervised Learning: Classifying cats and dogs

Silvia Arellano García, Arnau Casas Saez, Nora Dunder
KTH Royal Institute of Technology
silviaag@kth.se, arnaucs@kth.se, ndunder@kth.se

Abstract

In the field of deep learning, the availability of large, well-labeled datasets often presents a significant challenge. Using unlabeled data is seen as a potential solution, but it comes with the risk of learning incorrect patterns and suboptimal performance. This project addresses these challenges by implementing pseudo-labeling, a semi-supervised technique. These approaches, based on classifiers, predictors, and autoencoders, manage to achieve strong performance, particularly a fixed increase approach that shows significant improvements when leveraging large amounts of unlabeled data. Specifically, the fixed increase approach achieved an accuracy of 74.01% with 99% of unlabeled data, outperforming the basic and threshold approaches by margins of 20% and 10%, respectively.

1 Introduction

In the rapidly evolving field of machine learning, the primary bottleneck is often not the algorithms or computing power, but the availability of large, well-labeled datasets. These datasets are expensive and time-consuming to produce, and in some domains, nearly impossible to gather in significant quantities. Conversely, obtaining unlabeled data is usually less costly and easier, though it introduces the risk of the model learning incorrect patterns, potentially leading to subpar performance.

This project addresses these issues by leveraging semi-supervised learning techniques, particularly through pseudo-labeling. This method minimizes reliance on extensive labeled datasets by generating new labels from the model's predictions, thereby enhancing the learning process with minimal human supervision.

We have implemented four methods based on pseudo-labeling. The first three approaches are based on a classifier and a predictor, while the fourth approach utilizes an autoencoder combined with the k-nearest neighbors algorithm. The proposed methods perform very differently. The results from the methods based on the classifier and predictor are promising, achieving high accuracies even with small percentages of labeled data.

2 Related Work

Pseudo-labeling, also known as self-training, involves training a model using its own high-confidence predictions to augment the labeled dataset. One seminal work in this area is by Lee (2013), who proposed pseudo-labeling as a simple yet effective way to use unlabeled data to enhance supervised learning models. Lee's study on pseudo-labeling presents a practical semi-supervised approach that enhances learning by using both labeled and unlabeled data. This method assigns the most probable class labels to unlabeled data, treating them as accurate labels, which proves especially beneficial when labeled data is scarce. By incorporating techniques like Denoising Auto-Encoders and Dropout, Lee's work significantly boosts performance on tasks such as MNIST, establishing a strong foundation for semi-supervised learning methods.

This is a notice box

Despite its effectiveness, false pseudo-labels, which may negatively influence learning target representation, remain a major challenge in pseudo-labeling neural networks. Existing pseudo-labeling approaches often suffer from two more major drawbacks: conservatively expanding the label set and ignoring distinct contributions to the classification task, as exposed in Choi et al. (2019) and Li et al. (2022).

In recent years, there have been numerous enhancements to the basic pseudo-labeling technique. To tackle label noise, Xu et al. (2023a) proposed a neighborhood-based sample selection strategy. This approach capitalizes on the similarity of representations among samples with similar labels, leading to a 36.8% reduction in label noise while also optimizing processing time. Choi et al. (2019) introduced a similar density-based clustering algorithm that effectively addresses false pseudo-labels in unsupervised domain adaptation by leveraging high-density samples. Furthermore, Cascante-Bonilla et al. (2020) demonstrated that pseudo-labeling can rival state-of-the-art methods by integrating curriculum learning principles and implementing model parameter restarts to prevent concept drift between self-training cycles. These advancements collectively highlight the evolving landscape of pseudo-labeling techniques, offering robust solutions to address various challenges in semi-supervised learning, all of which would be interesting to explore further.

3 Data

For this project, we use *The Oxford-IIIT Pet Dataset* from Parkhi et al. (2012a), consisting of 7,439 RGB images representing 37 different breeds of cats and dogs. The dataset is well-balanced, with approximately 200 images per class.

The dataset presents images of animals captured in different backgrounds and positions, but always ensuring its recognizability. However, the images vary in size, which is inconvenient for training deep learning architectures. Therefore, the only preprocessing technique applied is resizing, guaranteeing all images are square and have the same size.

A relevant publication derived from this dataset was written by Parkhi et al. (2012b), which serves as the main base for this project. In their work, Parkhi et al. present a classifier of cat and dogs breeds, using two different classification methodologies. The first experiment entails discerning between cats and dogs, followed by the categorization into specific breeds. Conversely, the second experiment directly targets breed differentiation.

This dataset remains relevant in current research. An illustrative example of this is OmniVec, a model for learning embeddings of different natures using a common backbone network, introduced by Srivastava and Sharma (2024). By processing data from various sources such as point clouds, audio, or video, the model presents high generalization and a strong performance. In this paper, the Oxford-IIIT Pet Dataset is used for evaluating image classification tasks.

4 Methods

4.1 Binary classifier

Initially, we developed a binary classifier to differentiate between cats and dogs considering the complete labeled datasets. We achieved this by fine-tuning the ResNet-18 Network and modifying the last layer accordingly. The implementation commenced with preprocessing the data, involving cropping, resizing, and normalizing the images. This preprocessing step enhances model stability and facilitates learning. Subsequently, the dataset was split into training (comprising 80% of the samples) and testing (20% of the samples) sets. For training, we utilized batches of 100 images, an Adam optimizer, and conducted training for 3 epochs.

4.2 Breed classifier

Following the implementation of the previous section, this experiment involves fine-tuning the ResNet-18 model to classify 37 different cat and dog breeds. Initially, we pre-processed the images by resizing them to a squared size and normalizing them. However, this time, we also incorporated data augmentation to create a larger and more varied dataset. Among the data augmentation techniques employed are horizontal flips, random rotations ranging from 0º to 15º, and adjustments to brightness

and contrast within the range of 0 to 0.2. Subsequently, the dataset is divided into training, validation, and test sets.

Once the dataset is prepared, we proceed with fine-tuning the model. To accomplish this, we utilize the Adam optimizer, while introducing additional features not previously considered. Firstly, we incorporate a learning rate scheduler that reduces the learning rate by 10% after a specific number of steps, facilitating faster and more stable convergence. Furthermore, we enhance the base model by adding extra linear and batch normalization layers before the final layer, allowing the model to learn more complex distributions. Finally, we evaluate the benefits of fine-tuning the batch normalization layers of the model.

4.3 Semi-supervised learning

One of the primary objectives of this project is to explore semi-supervised learning and pseudo-labeling. Aiming to get practical experience in PyTorch, we implemented several methods approaches targeting this challenge. The implementation of these methods is available in our GitHub repository.

4.3.1 Basic approach

The initial approach attempted was a commonly used method employed in many previous works. The model used was the one that gave the best result for the breed classifier problem, the ResNet-18 with all the details in section 4.2 without decaying learning rate.

The first step involved training the model using labeled data. Subsequently, the model was used to predict the unlabeled data. Lastly, these predictions were used for further training alongside the labeled data. This approach enabled us to regularize the model, preventing overfitting and enhancing its generalization capabilities.

However, a notable drawback of this method is its reliance on a well-trained model for effectiveness. Without a sufficiently pre-trained model, incorrect predictions may occur, leading to poor learning outcomes from the predicted images.

4.3.2 Threshold approach

In an attempt to enhance the previous approach, instead of making a single prediction, we considered recomputing the predictions after each epoch during the secondary training phase. These predictions would involve assigning new labels to all the unlabeled data, regardless of the previous prediction results. However, this approach proved suboptimal for a less trained model, as it could lead to inconsistent predictions for a single image over time, confusing the model's features.

After exploring various ideas, we chose to incorporate a threshold for the probability derived from the softmax of the image. This threshold would signify the model's confidence level regarding the image classification. Only when the output from the softmax exceeded the specified threshold was the image included in the training. Consequently, we observed some improvements even with a less pre-trained model, although it still fell short of the performance achieved through supervised learning.

4.3.3 Fixed increase approach

Based on our previous findings, we identified that the primary issue lay in the indiscriminate addition of images, disregarding their predicted labels, which resulted in unbalanced training data across different breeds. To address this concern, we transitioned from using a threshold approach to a fixed addition of images, ensuring balance between breeds. This involved adding a fixed number of predicted images for each breed every epoch.

To implement a gradual addition process, we opted to use the Fibonacci sequence. This approach consisted on initially including on the labeled set the best image for each breed. After the training epoch, we recomputed the prediction for all the originally unlabeled datasets. The process then continued with the addition of two images, followed by three, five, and so forth. This was done to smooth the learning of the model with the predicted and uncertain images and to mimic the benefits of the previous threshold approach.

With this methodology, we aimed to facilitate the gradual growth of the model while mitigating the risk of overfitting on specific breeds.

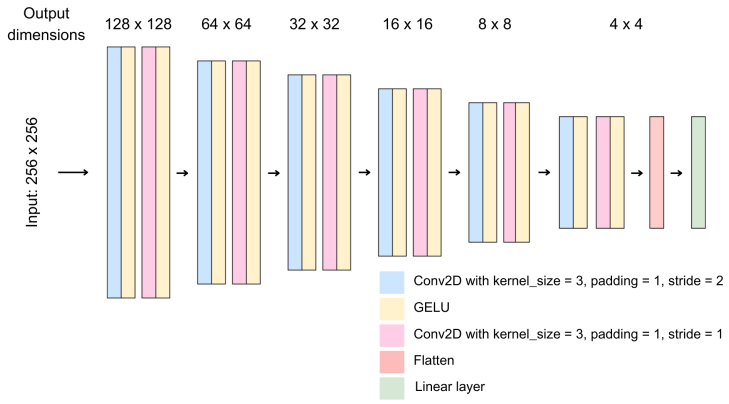

Figure 1: Encoder architecture.

4.3.4 Autoencoder with KNN

This method is inspired by the paper presented by Xu et al. (2023b), which uses the knowledge of similar samples to generate pseudo-labels. In our case, we started implementing an autoencoder architecture to obtain the latent representation of the images. The autoencoder consists of an encoder and a decoder, composed by 12 2D convolutional layers, which are followed by a GELU activation function. Finally, there is a Linear Layer. Figure 1 shows the structure of the encoder. The Decoder has the same structure but the layers are in reverse order, and the 2D-Convolutional Layers are substituted by 2D-Transposed Convoutional Layer. It is important to note that the AutoEncoder is trained using the MSE loss as a metric, which will be relevant in the result analysis.

Then, once all the images have a latent dimension, we applied the K-nearest neighbor algorithm to the unlabeled samples using the Euclidean distance and the Cosine distance. Then, the label that is repeated with more frequency amongst the k-neighbors is the one that is finally assigned to the sample.

5 Experiments

5.1 Binary and breed classifier

The binary classifier achieved a test accuracy of 99.05%, demonstrating a strong performance of the neural network.

With this result, we experimented with the breed classifier, incorporating some enhancements and different network versions. We explored various training configurations, including one, two, and three training layers, while employing enhancements such as the Adam optimizer, data augmentation with decaying learning rate (LR), and Batch Normalization (BN). The results are presented in Table 1. We observed that with only data augmentation and Adam optimizer, one layer performs adequately, but when incorporating Batch Normalization and decaying learning rate, using three layers results in a better performance.

	1 layer	**2 layers**	**3 layers**
Adam	88.70%	86.33%	86.87%
Adam with decaying LR	88.70%	87.55%	88.97%
Adam with decaying LR + BN	85.99%	88.97%	89.37%

Table 1: Accuracy of different models

5.2 Semi-supervised learning

For the semi-supervised learning part of the project, we opted to compare the results of three approaches: Basic, Threshold, and Fixed increase, using 50%, 90%, and 99% of unlabeled data. Table 2 presents the accuracies achieved by these different approaches.

The three methods perform similarly when using 50% and 90% of unlabeled data. Nevertheless, the difference in performance is more notable when using 99% of unlabeled data, where the fixed increase approach presents a stronger performance. This method surpasses both the basic and threshold methods by a margin of 20% and 10%, respectively.

It is important to reflect on the results obtained with the Fixed increase approach, which particularly stands out for its high accuracy with a very small number of labeled data samples. This efficiency stems from the equal increase of each breed over the epochs, enhancing the model's learning effectiveness and preventing overfitting on specific breed features.

	50%	**90%**	**99%**
Basic	85.51%	82.73%	55.65%
Threshold	85.91%	83.48%	64.18%
Fixed increase	87.27%	83.95%	74.01%

Table 2: Performance at different thresholds

Examining the confusion matrix of these models, presented in Appendix A, reveals instances where closely similar breeds were misclassified, such as the Russian Blue cat with the British Shorthair, and the Wheaten Terrier dog with the Havanese dog. This means that the model could not converge due to the limited amount of data available for those breeds, resulting in misclassifications over the epochs.

5.2.1 Results of the autoencoder with KNN

The results of the experiment in Section 4.3.4 are not included in the comparative results tables, as this approach differs significantly from the others and demonstrates poor performance. In this section, we will analyze the reasons for this failure and discuss potential ways to improve this method.

The test accuracy obtained with this method and 50% of unlabeled data is 5.95%. This may be due to an inaccurate representation in the latent space. By decoding the latent representations, we can see that the images are blurry, showing at most the contour of the animal, as illustrated in Figure 2.

Figure 2: Decoded images from the latent space.

Some potential solutions for improvement include using more latent dimensions and extending the training period. Additionally, the autoencoder was originally trained using the MSE as the metric. However, it is not the most suitable metric, as it does not preserve the patterns, which are crucial for differentiating the different breeds.

6 Conclusion

After delving deeper into transfer learning and semi-supervised learning, we have realized the importance of having a proper dataset and how much the quality of the results can differ depending on the pseudo-labeling method used.

Some potential future extensions include improving the autoencoder method introduced in section 4.3.4 by implementing a more complex autoencoder or conducting an ablation study by tuning its parameters. Other possible extensions could involve comparing the performance of pseudo-labeling with other methods such as consistency regularization.

Overall, we believe this project has helped us improve our skills in PyTorch and coding deep learning architectures. It has complemented the theory learned in lectures with more hands-on experience.

Moreover, reading papers has helped us become familiar with diverse deep-learning architectures and methods used in today's technology.

References

D.-H. Lee, "Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks," *ICML 2013 Workshop : Challenges in Representation Learning (WREPL)*, 07 2013.

J. Choi, M. Jeong, T. Kim, and C. Kim, "Pseudo-labeling curriculum for unsupervised domain adaptation," *ArXiv*, vol. abs/1908.00262, 2019.

Y. Li, J. Yin, and L. Chen, "Informative pseudo-labeling for graph neural networks with few labels," *Data Mining and Knowledge Discovery*, vol. 37, pp. 228–254, 2022.

R. Xu, Y. Yu, H. Cui, X. Kan, Y. Zhu, J. Ho, C. Zhang, and C. Yang, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, pp. 10 611–10 619, Jun. 2023. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/26260

P. Cascante-Bonilla, F. Tan, Y. Qi, and V. Ordonez, "Curriculum labeling: Revisiting pseudo-labeling for semi-supervised learning," pp. 6912–6920, 2020.

O. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "The Oxford-IIIT PET Dataset," http://www.robots.ox.ac.uk/~vgg/data/pets/index.html, 2012.

O. M. Parkhi, A. Vedaldi, A. Zisserman, and C. V. Jawahar, "Cats and dogs," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

S. Srivastava and G. Sharma, "Omnivec: Learning robust representations with cross modal sharing," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2024, pp. 1236–1248.

R. Xu, Y. Yu, H. Cui, X. Kan, Y. Zhu, J. Ho, C. Zhang, and C. Yang, "Neighborhood-regularized self-training for learning with few labels," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 9, 2023, pp. 10 611–10 619.

Appendix A Confusion matrices

Basic

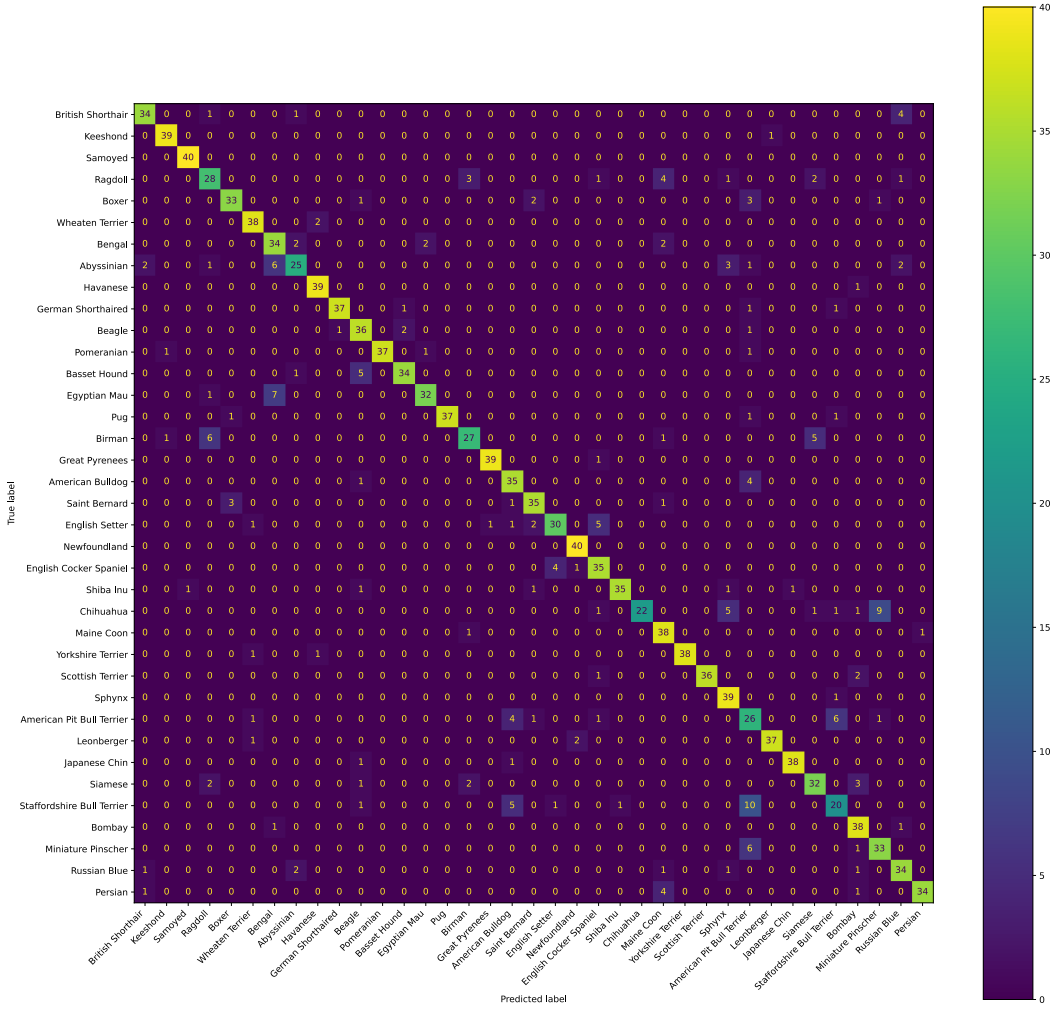

Figure 3: Basic 50%: Confusion matrix of accuracy of test data

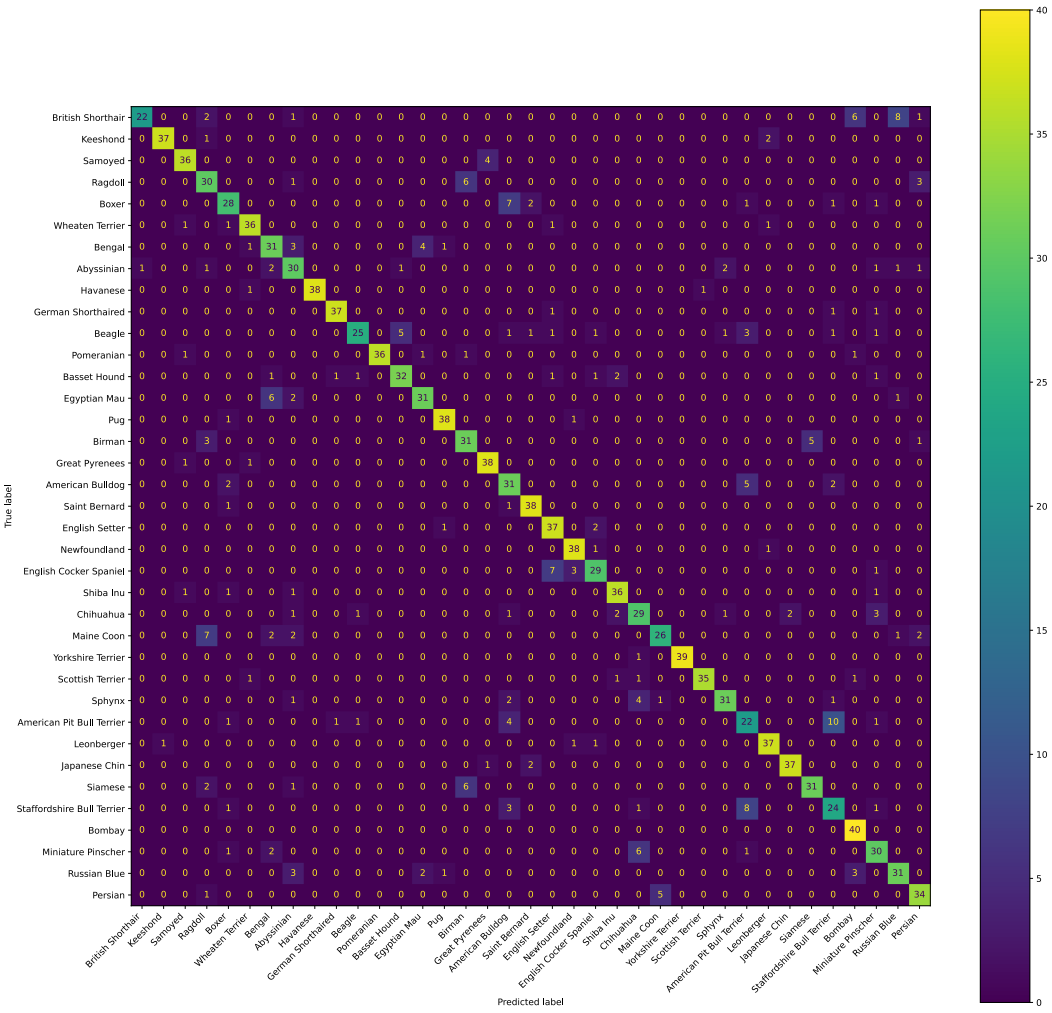

Figure 4: Basic 90%: Confusion matrix of accuracy of test data

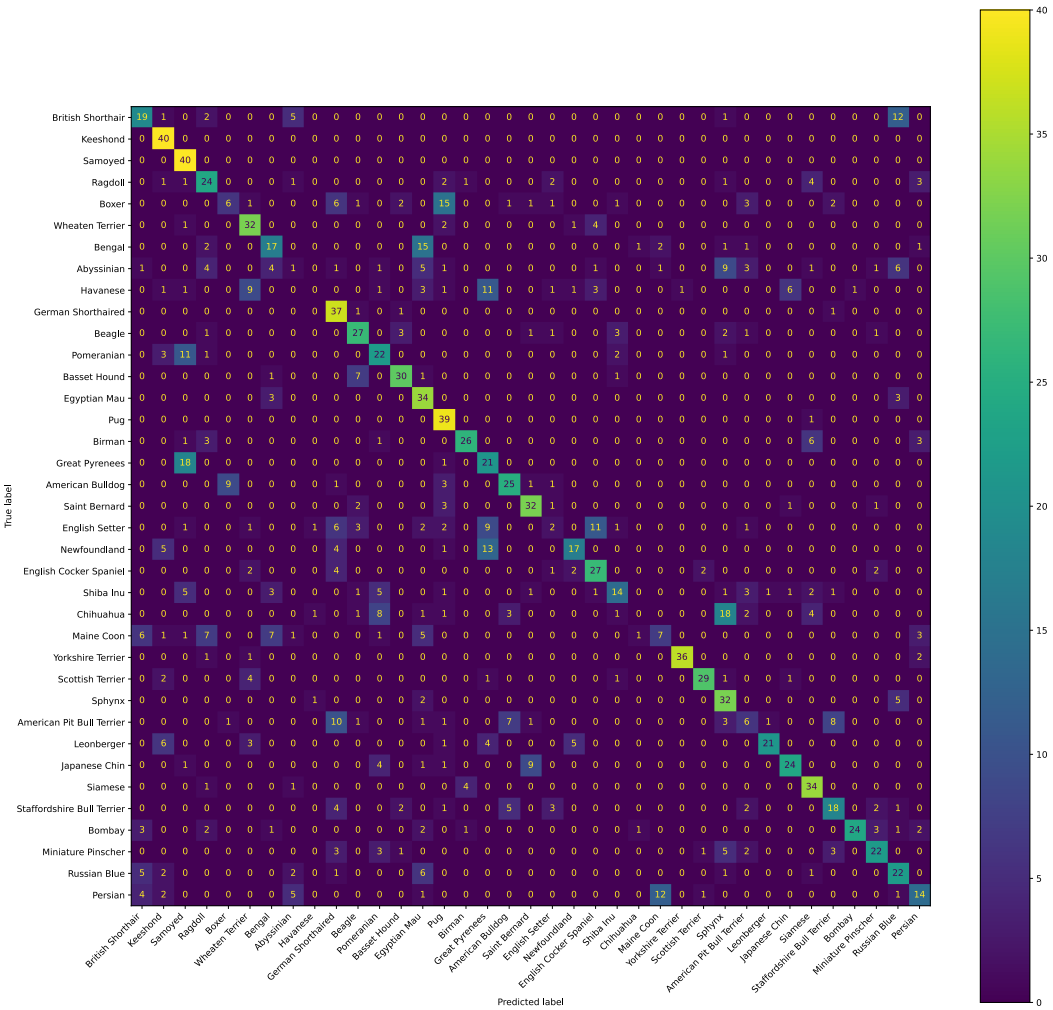

Figure 5: Basic 99%: Confusion matrix of accuracy of test data

Threshold

For this, we have a new classification called "No threshold" as they are images that do not pass the 45% threshold.

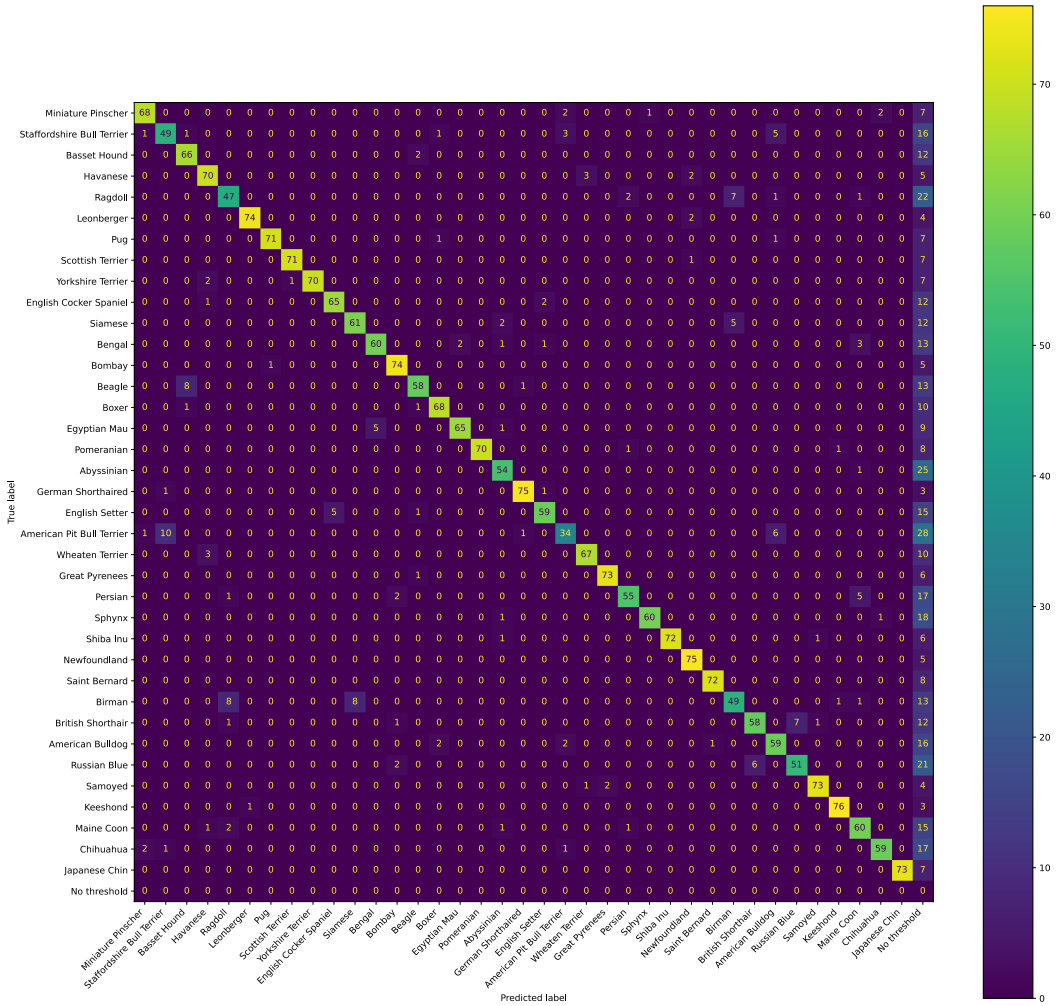

Figure 6: Threshold 50%: Confusion matrix of predictions of unlabeled data at epoch 6

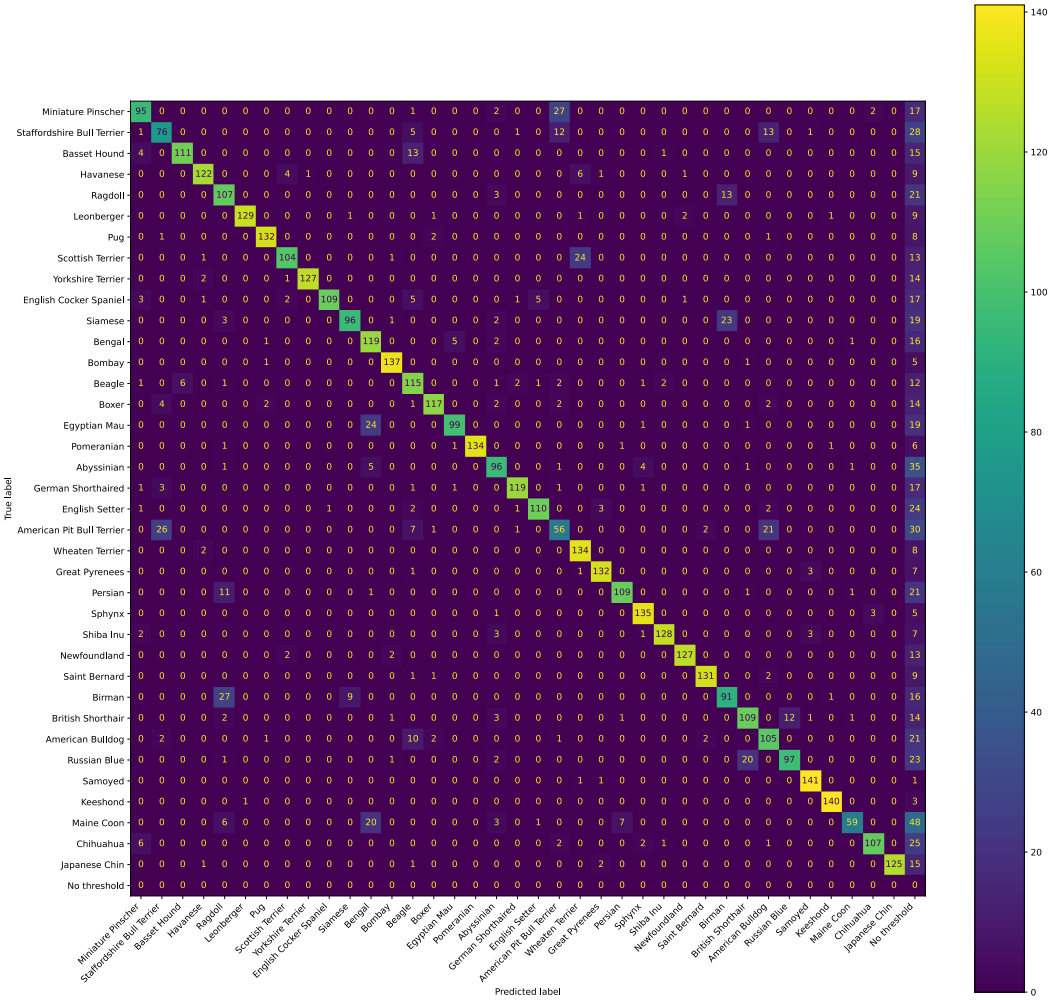

Figure 7: Threshold 90%: Confusion matrix of predictions of unlabeled data at epoch 12

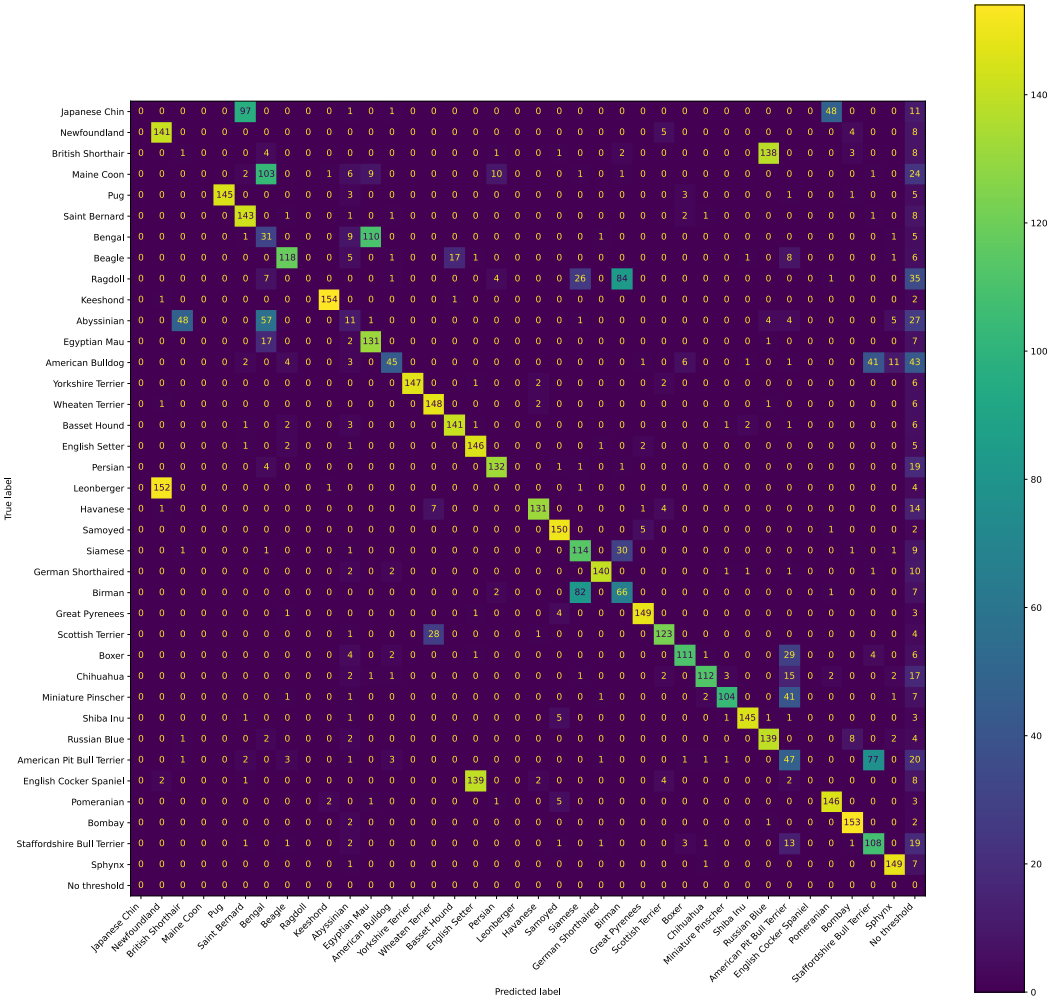

Figure 8: Threshold 99%: Confusion matrix of predictions of unlabeled data at epoch 27

Fixed increase

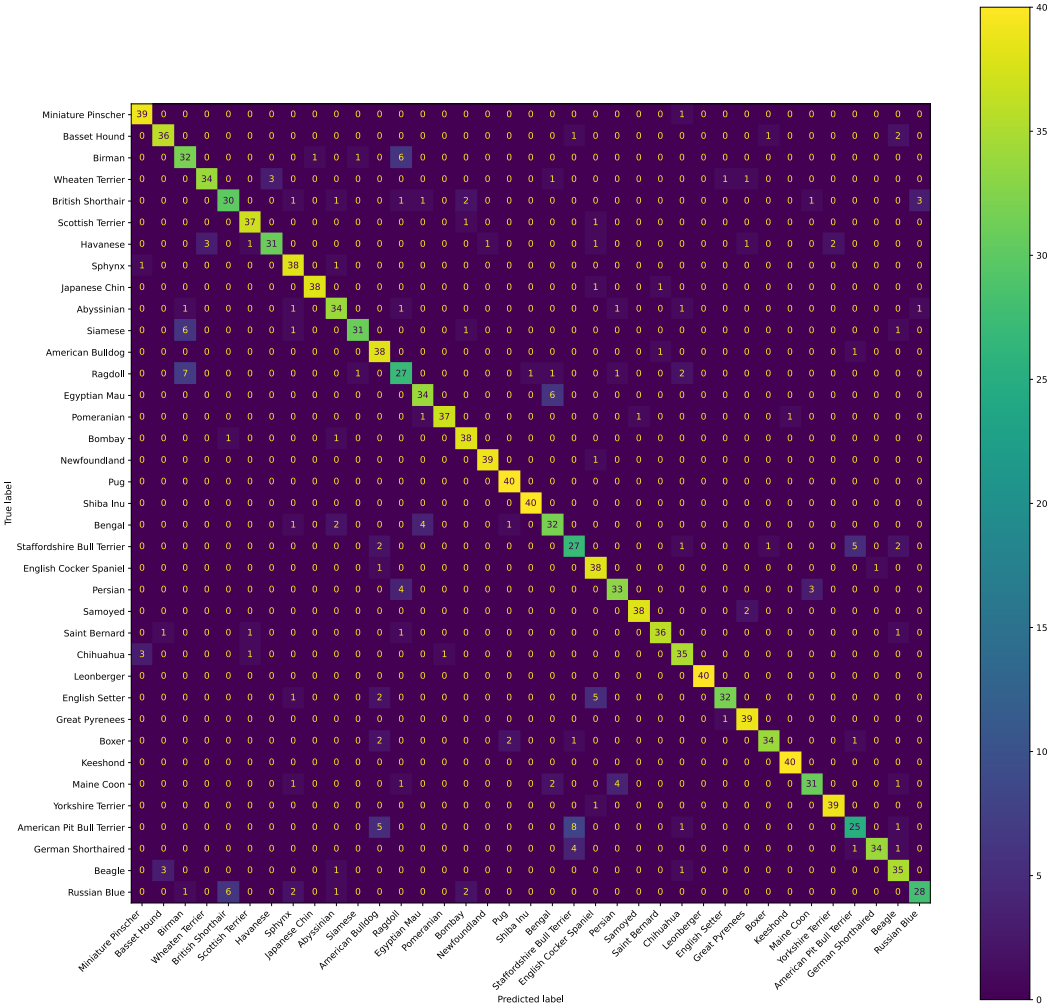

Figure 9: Fixed increase 50%: Confusion matrix of accuracy of test data

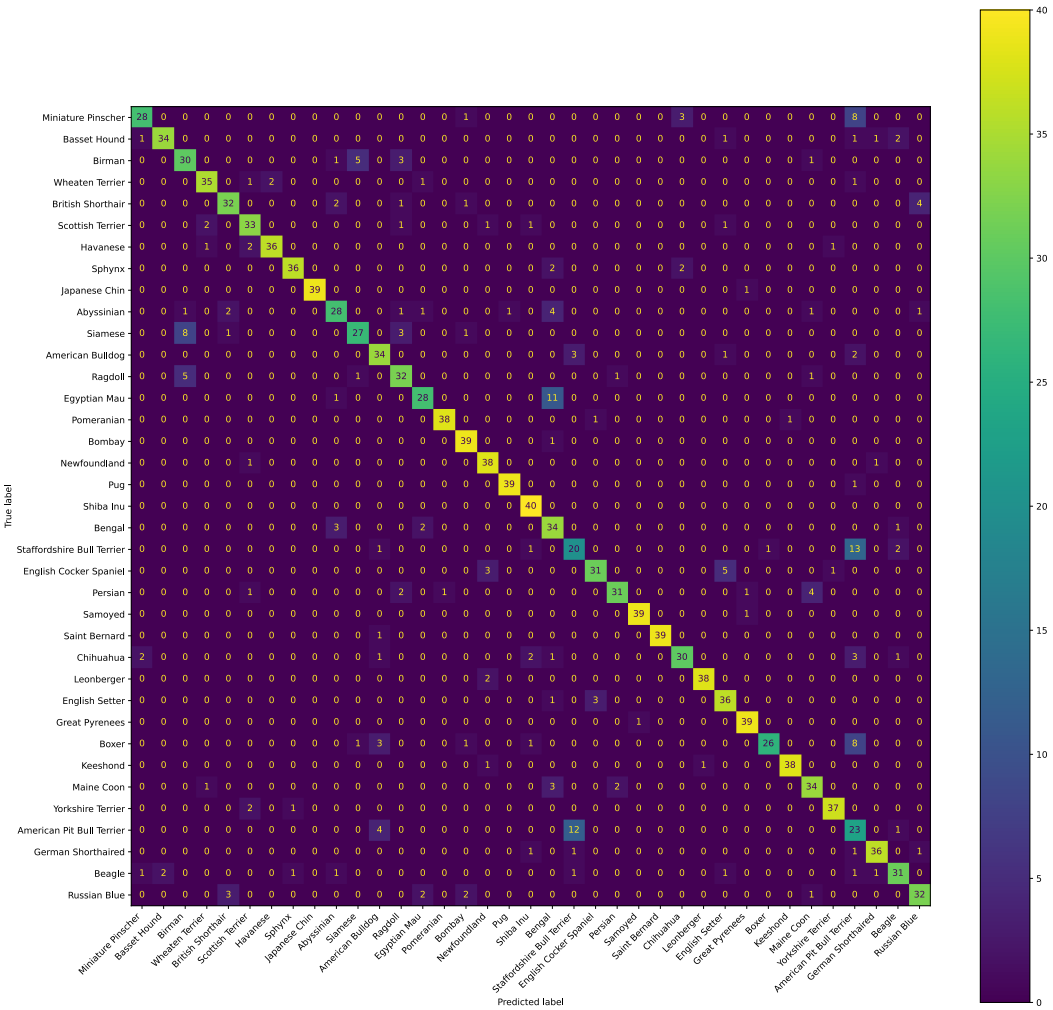

Figure 10: Fixed increase 90%: Confusion matrix of accuracy of test data

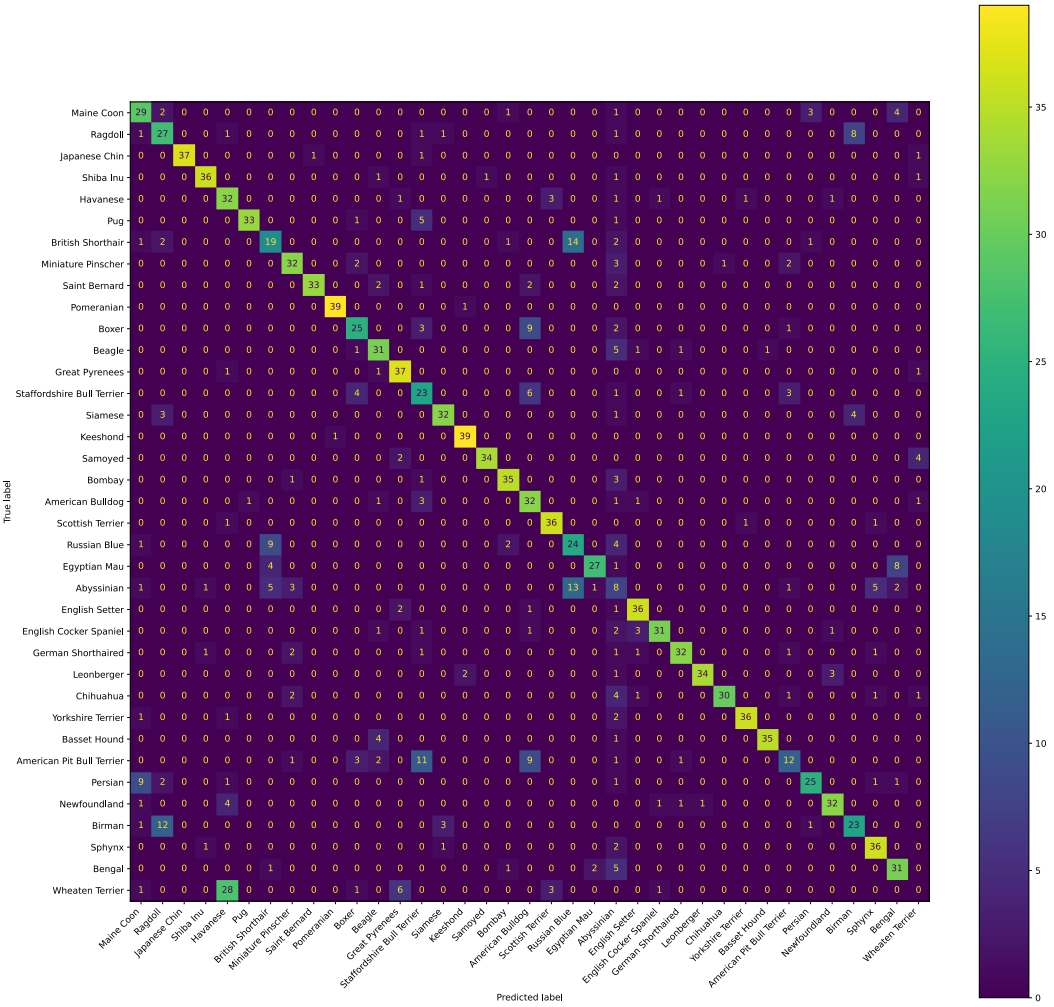

Figure 11: Fixed increase 99%: Confusion matrix of accuracy of test data