A Simple Nadaraya-Watson Head can offer Explainable and Calibrated Classification

Anonymous authors
Paper under double-blind review

Abstract

In this paper, we empirically analyze a simple, non-learnable, and nonparametric Nadaraya-Watson (NW) prediction head that can be used with any neural network architecture. In the NW head, the prediction is a weighted average of labels from a support set. The weights are computed from distances between the query feature and support features. This is in contrast to the dominant approach of using a learnable classification head (e.g., a fully-connected layer) on the features, which can be challenging to interpret and can yield poorly calibrated predictions. Our empirical results on an array of computer vision tasks demonstrate that the NW head can yield better calibration than its parametric counterpart, while having comparable accuracy and with minimal computational overhead. To further increase inference-time efficiency, we propose a simple approach that involves a clustering step run on the training set to create a relatively small distilled support set. In addition to using the weights as a means of interpreting model predictions, we further present an easy-to-compute "support influence function," which quantifies the influence of a support element on the prediction for a given query. As we demonstrate in our experiments, the influence function can allow the user to debug a trained model. We believe that the NW head is a flexible, interpretable, and highly useful building block that can be used in a range of applications.

1 Introduction

Many state-of-the-art classification models are parametric deep neural networks, which can be viewed as a combination of a deep feature extractor followed by one or more fully-connected (FC) classification layers (He et al., 2015; Huang et al., 2016; Dosovitskiy et al., 2020). Two common problems with parametric deep learning models are that they can be hard to interpret (Li et al., 2021) and often suffer from poor calibration (Guo et al., 2017).

Nonparametric (and, similarly, attention-based) modeling strategies have been explored extensively in deep learning (Wilson et al., 2015; Papernot & McDaniel, 2018; Chen et al., 2018; Kossen et al., 2021; Laenen & Bertinetto, 2020; Iscen et al., 2022; Frosst et al., 2019). In this approach, the prediction for a given test input (which we will also refer to as a query) is computed as an explicit function of training datapoints. These models can be more interpretable, since the dependence on other datapoints gives an indication about what is driving the prediction Hanawa et al. (2021). Yet, they can suffer from significant computational overhead, can take a hit in performance versus parametric approaches, and/or require complex approximate inference schemes (Bui et al., 2016; Salimbeni & Deisenroth, 2017).

In this work, we present an analysis of a Nadaraya-Watson (NW) head, a simple, non-learnable, and nonparametric prediction layer based on the Nadaraya-Watson model (Nadaraya, 1964; Watson, 1964; Bishop, 2006). For a given query, a pre-defined similarity or distance is computed between query features and the features in a "support set", which can be made up of samples from the training data. These scores are in turn used to compute a weighted average of the labels in the support set.

As we discuss in this paper, the choice of the support set for the NW head affords the user an additional degree of freedom in implementation. We demonstrate several ways of constructing the support set, including

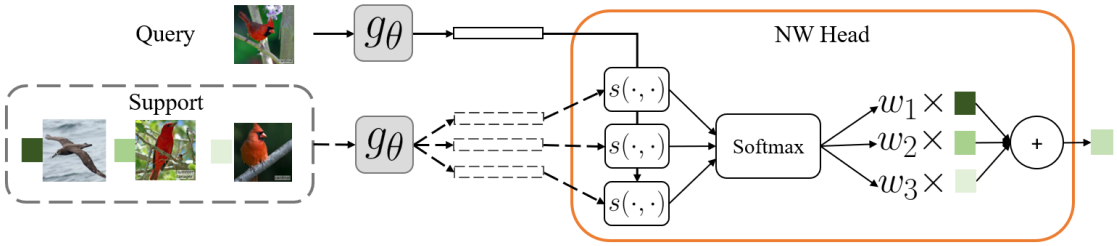


Figure 1: The NW head. Query image and support images are fed through a feature extractor g_θ. Pairwise similarities $s(\cdot, \cdot)$ (e.g. negative Euclidean distance) are computed between the query and each support feature. These similarities are normalized and used as weights for computing the weighted sum of corresponding support labels.

a simple support set distillation approach based on clustering, and discuss the relative advantages and trade-offs. In our experiments, we show that the NW head can achieve comparable (and sometimes better) performance than the FC head on a variety of computer vision datasets, with little sacrifice to computation time. In particular, we find that the NW head exhibits better calibrated predictions.

In addition, we explore two means of interpretability/explainability which fall naturally from the NW head. The first is interpreting the support set weights, which directly correspond to the degree of contribution a specific training datapoint has on the prediction. The second is our novel concept of "support influence", which highlights helpful and harmful support examples and can be used as a diagnostic tool to understand and explain model behavior. In our experiments, we demonstrate the utility of these methods for purposes of interpreting, explaining, and potentially intervening in model predictions.

2 Related Work

2.1 Nonparametric Deep Learning and Attention

Nonparametric models in deep learning have received much attention in previous work. Deep Gaussian Processes (Damianou & Lawrence, 2013), Deep Kernel Learning (Wilson et al., 2015), and Neural Processes (Kossen et al., 2021) build upon Gaussian Processes and extend them to representation learning. Other works have generalized k-nearest neighbors (Papernot & McDaniel, 2018), decision trees (Zhang et al., 2018), density estimation (Fakoor et al., 2020), and more general kernel-based methods (Nguyen et al., 2021; Zhang et al., 2021; Ghosh & Nag, 2001) to deep networks and have explored the interpretability that these frameworks provide.

Closely-related but orthogonal to nonparametric models are attention-based models, most notably self-attention mechanisms popularized in Transformer-based architectures in natural language processing (Vaswani et al., 2017) and, more recently, computer vision (Dosovitskiy et al., 2020; Jaegle et al., 2021; Parmar et al., 2018). Retrieval models in NLP learn to look up relevant training instances for prediction (Sachan et al., 2021; Borgeaud et al., 2021) in a nonparametric or semiparametric manner. Many works have recognized the inherent interpretable nature of attention and have leveraged it in vision (Chen et al., 2018; Guan et al., 2018; Wang et al., 2017; Xu et al., 2015). More recently, nonparametric transformers (NPT) (Kossen et al., 2021) applied attention in a nonparametric setting. NPT leverages self- and cross-attention between both attributes and datapoints, which are stacked up successively in an alternating fashion to yield a deep, non-linear architecture.

Building upon and in contrast to these prior works, we present a comparatively simple nonparametric head which is computationally-efficient and performant, and which additionally lends itself naturally to being interpretable.

2.2 Metric Learning

Metric learning seeks to quantify the similarity between datapoints for use in downstream tasks (e.g., unsupervised pretraining (Wu et al., 2018) or retrieval (Chen et al., 2021)). Of particular relevance to our method is metric learning in the context of low-shot classification (Snell et al., 2017; Vinyals et al., 2016; Sung et al., 2017; Santoro et al., 2016). In particular, Matching Networks (Vinyals et al., 2016) are most similar to our model in the sense that it uses an NW-style model to compute predictions, where both the query image and support set are composed of examples from *unseen* classes. In contrast, we analyze the NW head from a nonparametric perspective in the typical, many-shot image classification setting.

2.3 Influence Functions

Following classical work by Cook & Weisberg (1982), Koh & Liang (2017) propose the influence function as a means of explaining black-box neural network predictions. Let h denote a function parameterized by θ. $\mathcal{D} = \{z_i : (x_i, y_i)\}_{i=1}^n$ is a set of training examples and $L(h_\theta(z))$ indicates the loss for a training example z. θ^* is the set of optimal parameters which minimizes the empirical risk. Similarly, the parameter set associated with up-weighting a training example z by ϵ is found by solving:

$$\arg\min_\theta \frac{1}{n} \sum_{i=1}^{n} L(h_\theta(z_i)) + \epsilon L(h_\theta(z)). \tag{1}$$

Koh & Liang (2017) define influence as the change in the loss value for a particular test point z_t when a training point z is up-weighted. Using a first-order Taylor expansion and the chain rule, influence can be approximated by a closed-form expression:

$$\mathcal{I}(z_t, z) = -\nabla L(h_{\theta^*}(z_t))^T H_{\theta^*}^{-1} \nabla L(h_{\theta^*}(z)), \tag{2}$$

where H_{θ^*} represents the Hessian with respect to θ^*. $\mathcal{I}(z_t, z)/n$ is approximately the change in the loss for the test-sample z_t when a training sample z is removed from the training set. However, this result is based on the assumption that the underlying loss function is strictly convex in θ and that the Hessian matrix is positive-definite. Follow-up work has questioned the efficacy of influence functions in practical scenarios where these assumptions are violated (Basu et al., 2020).

In this work, we define the support influence function which follows naturally from the NW head; it computes the change in the loss directly on the prediction vector itself (and not indirectly through the model parameters θ). Furthermore, it is easy-to-compute, requires no approximations, and does not require any assumptions on convexity or differentiability.

3 Methods

3.1 Nadaraya-Watson Head

Consider a "support set" of N_s examples and their associated labels, $\mathcal{S} = \{z_i : (x_i, y_i)\}_{i=1}^{N_s}$. This support set can be a subset, or the whole, of the training dataset. In this paper, we will focus on the image classification setting, where x's are images and y's are integer class labels from a label set $\mathcal{C} = \{1, ..., C\}$. We will use $\vec{y}$ to denote the one-hot encoded version of y. We note that the NW head can be used with any input data type, not just images, and can be readily extended to the regression setting.

For a query image x, the Nadaraya-Watson prediction is computed as the weighted sum of support labels $\vec{y}_i$, where the weights quantify the similarity of the query image with each support image x_i:

$$f(x, \mathcal{S}) = \sum_{i=1}^{N_s} w(x, x_i)\vec{y}_i. \tag{3}$$

Classically, the weights are defined as a function of handcrafted kernel functions κ:

$$w(x, x_i) = \frac{\kappa(x, x_i)}{\sum_{j=1}^{N_s} \kappa(x, x_j)}, \tag{4}$$

where $\kappa(\cdot,\cdot)$ quantifies the non-negative similarity between its two arguments (Bishop, 2006). Higher similarity values correspond to more similar pairs of datapoints.

In the NW head, we define the weights as:

$$w_\theta(x, x_i) = \frac{\exp\{-\|g_\theta(x) - g_\theta(x_i)\|_2/\tau\}}{\sum_{j=1}^{N_s} \exp\{-\|(g_\theta(x) - g_\theta(x_j))\|_2/\tau\}}. \tag{5}$$

Here, g_θ is a feature extractor with parameters θ, which we implement as a neural network. τ is a temperature hyperparameter. $\|\cdot\|_2$ denotes Euclidean distance. Since $\sum_{i=1}^{N_s} w_\theta(x, x_i) = 1$ and $\vec{y}_i$'s are one-hot encoded vectors, the output f of the estimator can be interpreted as a conditional probability for the class label given the image and support set.

3.2 Loss Function and Training

Let $\mathcal{D}$ be a training set of image and label pairs. Our objective to minimize is:

$$\arg\min_\theta \sum_{(x,y)\in\mathcal{D}} \mathbb{E}_{\mathcal{S}\sim\mathcal{D}}\ L(f_\theta(x, \mathcal{S}), y), \tag{6}$$

where L is the cross-entropy loss. This indirectly encourages similar-looking images to have higher similarity score, since more similar images will tend to have the same labels.

We optimize this objective using stochastic gradient descent (SGD) by sampling a mini-batch size N_b of query-support pairs and computing gradients with respect to the mini-batch. Note that our model assumes no dependency between support set samples; indeed, we are free to sample any arbitrary $\mathcal{S}$ and query x, as long as the class of x is within the support classes. Therefore, the support set size $N_s = |\mathcal{S}|$ is a hyperparameter analogous and orthogonal to the batch size N_b.

3.3 Inference

The ability to select and manipulate the support set at inference time unlocks a degree of flexibility not possible with parametric classification models. In particular, the label set $\mathcal{C}$ of the support set directly determines the label set of the resulting prediction. For example, if the user has prior knowledge of which classes a particular query is most likely to be, the support set can be restricted to only contain those classes.

Note that we assume the model has been trained with random support sets drawn from the training data for each mini-batch. To characterize the effect of the support set on inference, in our experiments we implement the following "inference modes":

1. **Full.** Use the entire training set: $\mathcal{S} = \mathcal{D}$.

2. **Random.** Sample uniformly at random over the dataset, such that each class is represented k times: $\mathcal{S} \sim \mathcal{D}$ and $|\mathcal{S}| = k|\mathcal{C}|$.

3. **Cluster.** Given the trained model, we first compute the features for all the training datapoints. Then, we perform k-means clustering on the features of the training datapoints for each class. These k cluster centroids are then used as the support features for each class. Note that in cluster mode, the support set does not correspond to observed datapoints.

4. **Closest cluster (CC).** Same as Cluster, except the support features correspond to real training datapoints that are closest to the cluster centroids in terms of Euclidean distance.

Each mode except Full assumes that each class is represented k times in $\mathcal{S}$. One can, obviously, modify this assumption to reflect any class imbalance that might exist in the training data.

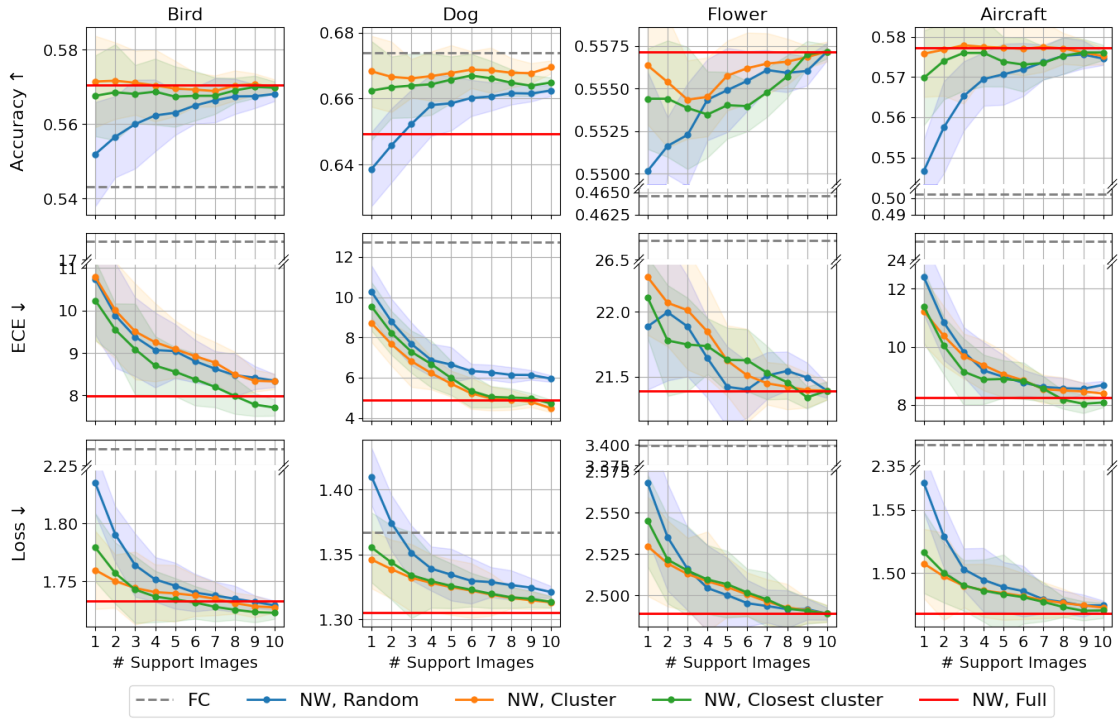


Figure 2: Model performance across different support set sizes for DenseNet-121. Full mode and parametric baseline are constant across support set size and are depicted as horizontal lines.

3.4 Support Influence

Koh & Liang (2017) define influence as the change in the loss as a result of upweighting a particular training point. With an NW head, quantifying the influence of a support image to the given query is straightforward and can be computed in closed-form. Indeed, the proposed support influence computes the change in the loss directly on the prediction vector itself (and not indirectly through the model parameters θ), and furthermore does not require approximations, nor assumptions on convexity or differentiability.

Denote by $\mathcal{S}_{-z_s} = \mathcal{S} \setminus z_s$, i.e. the support set with an element removed. For a fixed query x:

$$f(x, \mathcal{S}_{-z_s}) = \frac{f(x, \mathcal{S}) - w(x, x_s)\vec{y}_s}{1 - w(x, x_s)}. \tag{7}$$

We define the support influence of z_s on z as the change in the cross-entropy loss incurred from removing a support element z_s:

$$\mathcal{I}(z, z_s) = L(f(x, \mathcal{S}_{-z_s}), y) - L(f(x, \mathcal{S}), y) = \log\left(\frac{f^y - f^y w(x, x_s)}{f^y - w(x, x_s)\mathbb{1}_{\{y=y_s\}}}\right), \tag{8}$$

where superscript denotes indexing into a vector and $\mathbb{1}$ is the indicator function. The derivation is provided in the Appendix.

4 Experiments

4.1 Experimental Setup

Datasets. We experiment with CUB-200-2011 (Bird) (Wah et al., 2011), Stanford Dogs (Dog) (Khosla et al., 2011), Oxford Flowers (Flower) (Nilsback & Zisserman, 2008), and FGVC-Aircraft (Aircraft) (Maji et al., 2013).

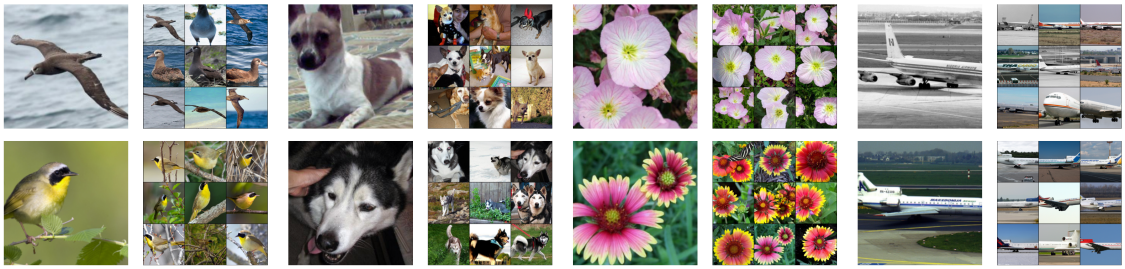


Figure 3: Corresponding images of embeddings closest to cluster centroids, for $k = 1$ and $k = 9$.

Table 1: Accuracy ↑ / ECE ↓ (%). **Bold** is best and underline is second best. $k = 1$ for NW models.

	Model	*Bird*	*Dog*	*Flower*	*Aircraft*
ResNet-18	FC	50.23 / 20.23	**62.58** / 16.02	53.00 / 22.26	50.59 / 24.55
	NW, Random	48.55 / 7.762	54.67 / 5.583	56.14 / 19.59	58.24 / 12.16
	NW, Cluster	**53.40** / 6.051	61.90 / 4.025	**57.13** / 19.83	60.06 / 12.12
	NW, CC	52.24 / 6.224	61.00 / 4.757	56.30 / 20.09	60.01 / 11.66
	NW, Full	53.02 / **3.327**	61.93 / **4.256**	57.03 / **19.10**	**60.07** / **9.235**
DenseNet-121	FC	54.32 / 17.67	**67.38** / 12.74	46.46 / 26.79	50.26 / 25.03
	NW, Random	55.41 / 10.74	61.53 / 10.26	55.02 / 21.89	54.67 / 12.40
	NW, Cluster	**56.77** / 10.78	66.58 / 8.709	55.64 / 22.27	57.58 / 11.23
	NW, CC	56.32 / 10.23	66.04 / 9.535	55.44 / 22.11	56.98 / 11.39
	NW, Full	56.67 / **7.997**	66.93 / **4.892**	**55.72** / **21.39**	**57.73** / **8.248**

Network Architecture and Hyperparameters. For the feature extractor g_θ, we use a convolutional neural network (CNN) followed by a linear projection to an embedding space of dimension d. In this work, we experiment with the ResNet-18 (He et al., 2015) and DenseNet-121 (Huang et al., 2016) architectures and set $d = 128$ and $\tau = 1$.

We train each model 3 times with different random initializations. During training, we set the randomly sampled support size to be $N_s = 10$ for all datasets, and the mini-batch size to be $N_b = 4$. Empirically, we find that $N_s > 5$ is necessary in order to have a sufficient number of "negative" samples to compare against. We use SGD with momentum 0.9, weight decay 1e-4, and an initial learning rate of 1e-3. The learning rate is divided by 10 after 20K and 30K gradient steps, and training stops after 40K gradient steps. We use the standard data augmentation technique of random flipping and cropping. All training and inference is done on an Nvidia A6000 GPU and all code is written in Pytorch[2].

Table 3: GPU inference times ↓ for ResNet-18.

Model	Time (10^{-3} sec)
FC	4.00
NW, $k = 1$	4.05
NW, $k = 10$	4.78
NW, Full[1]	7.76

Baseline. We compare the NW head to a standard parametric model with a fully-connected (FC) head. For the parametric model, all hyperparameters are identical to NW except we train for 200 epochs, use an initial learning rate of 0.1, and divide the learning rate by 10 after epochs 100 and 150. We found that this setting yielded better results for the baseline models.

[1]Computed on Bird dataset, where $k = 5994$.
[2]Code to be provided upon publication.

Table 2: Accuracy ↑ / ECE ↓ (%) for label smoothing (LS) and temperature scaling (TS) for ResNet-18. **Bold** is best. Cluster and $k = 1$ for NW models.

Model	*Bird*	*Dog*	*Flower*	*Aircraft*
FC-LS	54.64 / 15.83	**62.42** / 10.79	53.65 / 3.84	51.28 / 7.16
NW-LS	**54.90** / **2.097**	61.71 / **2.24**	**56.77** / **1.99**	**54.24** / **3.61**
FC-TS	50.23 / 2.711	**62.58** / 2.208	53.00 / 4.426	50.59 / **4.423**
NW-TS	**53.40** / **2.460**	61.90 / **1.911**	**57.13** / **2.693**	**60.06** / 5.191

4.2 Model Performance

We measure performance using accuracy (the fraction of correctly classified test cases) and expected calibration error (ECE) (Guo et al., 2017; Naeini et al., 2015). ECE approximates the difference in expectation between confidence and accuracy, computed by partitioning the model predictions into bins and taking a weighted average of each bins' difference between confidence and accuracy. Table 1 shows these metrics for all models and datasets, and Fig. 4 further shows reliability diagrams. We generally observe that the NW head achieves an accuracy that is comparable and at times better than its parametric counterpart, and always with improved calibration.

Table 3 shows inference times for both parametric and the NW-head models, where we assume that all support embeddings, cluster centroids, and closest cluster embeddings have been precomputed. We observe that inference times are nearly identical to the FC head, except Full.

In Fig. 2, we plot various metrics across support set size k. Generally, performance improves as support set size increases. This could be explained by the fact that more support images leads to more robust predictions, as more support examples per class can average out potential errors. At the limit, Full mode typically performs the best, at the cost of inference time.

Cluster mode often improves over Random mode and is on par with Full mode (at larger k) while having favorable inference time. However, this comes at the cost of loss of interpretability, since the cluster centroids do not correspond to actual images in the support set. Closest cluster addresses this deficiency, often with minimal sacrifice to performance compared to Cluster.

Fig. 3 visualizes the support images used in Closest cluster for two classes, with $k = 1$ and $k = 9$. These images are the prototypical examples of the class and can be interpreted as a summary or compression of the class and dataset. We observe that for $k = 1$, the support image is a prototypical example of the class. For $k = 9$, the images have a variety of poses and backgrounds, allowing the model to have a higher chance of computing similarities with a wide range of class prototypes. Further research might explore leveraging these prototypes.

Overall, our experiments indicate that the NW head has advantages in accuracy and calibration in the datasets we tested. These datasets are small-medium in size and are relatively fine-grained; it remains to be seen whether these results will hold on larger, diverse datasets, which we leave to future work.

4.2.1 Improving Calibration

In this experiment, we explore whether existing calibration techniques originally proposed for parametric models can further improve calibration. We explore two popular and effective techniques: label smoothing (LS) and temperature scaling (TS). Label smoothing (Szegedy et al., 2015) uses soft labels that are a weighted average of the one-hot labels and the uniform distribution. We choose the weight used in the original paper, 0.1. Temperature scaling (Guo et al., 2017) is a post-training technique which optimizes the temperature τ in the softmax computation. Following the original paper, τ is optimized with respect to the loss on the validation set.

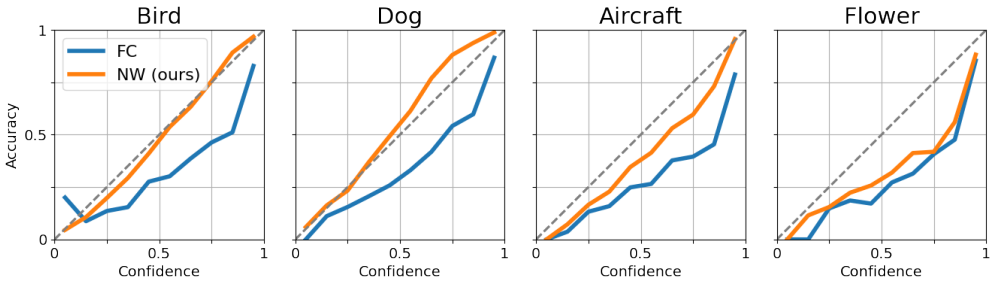


Figure 4: Reliability diagrams for ResNet-18. DenseNet-121 in Appendix.

We show the results in Table 2. In general, we find that the NW head outperforms the FC head under both LS and TS techniques. In particular, the simple, post-training technique of TS is a simple strategy that further improves our calibration performance.

4.3 Interpretability and Explainability

In the following section, we restrict our attention to Full mode and explore the interpretability and explainability aspects of the NW head.

4.3.1 Interpreting the Weights

Visualizing the support images ranked by weight w can give insight to users about how the model is making a prediction. For cases where the model is unsure, visualizing the support images along with confidence scores can uncover where the model is unsure and potentially allow for user intervention.

In Fig. 5, we plot three examples of a query image and the top support set images ranked by weight w, along with the top 3 softmax predictions. We notice significant visual similarity between the query image and the top support set images. Notably, the second query of a “cardinal" is incorrectly classified by the model. We notice that the model is confused between the top 2 classes in the support set. The model thinks the third query of “barbeton daisy" is most similar to “sunflowers", possibly due to the similarity in color and shape. A user can look at the support set and potentially intervene, either by correcting the model's prediction or removing confounding elements from the support set. An automated method for flagging conspicuous predictions could be based on confidence scores.

Additional visualizations are provided in Appendix A.3. In Appendix A.4, we compare against ranking images by Euclidean distance in the feature space for the baseline FC model, and show that the degree of semantic similarity between the query and top most similar images is lower than the NW head.

4.3.2 Explaining Model Behavior with Support Influence

Support influence can be used to debug and diagnose issues in otherwise black-box neural networks. In this section, we explore the use of support influence in helping to determine helpful and harmful training examples. Note that the support influence requires knowledge of the ground truth label for the query image and thus cannot be used during inference/test time.

By ranking and visualizing the support set by support influence, we can see the support images which are most helpful (highest $\mathcal{I}(z, z_s)$ values) and most harmful (lowest $\mathcal{I}(z, z_s)$ values) in predicting the given query z. In Fig. 6, we visualize three query images and their top 5 most helpful and top 5 most harmful support images. The top images are “helpful" in the sense that they belong to the same class as the query and look visually similar. The bottom images are “harmful" since, despite the fact that they are visually similar, they are a different class. For the first dog example, we see that “redbone" and “walker hound" breeds are harmful examples for the “beagle" query image.

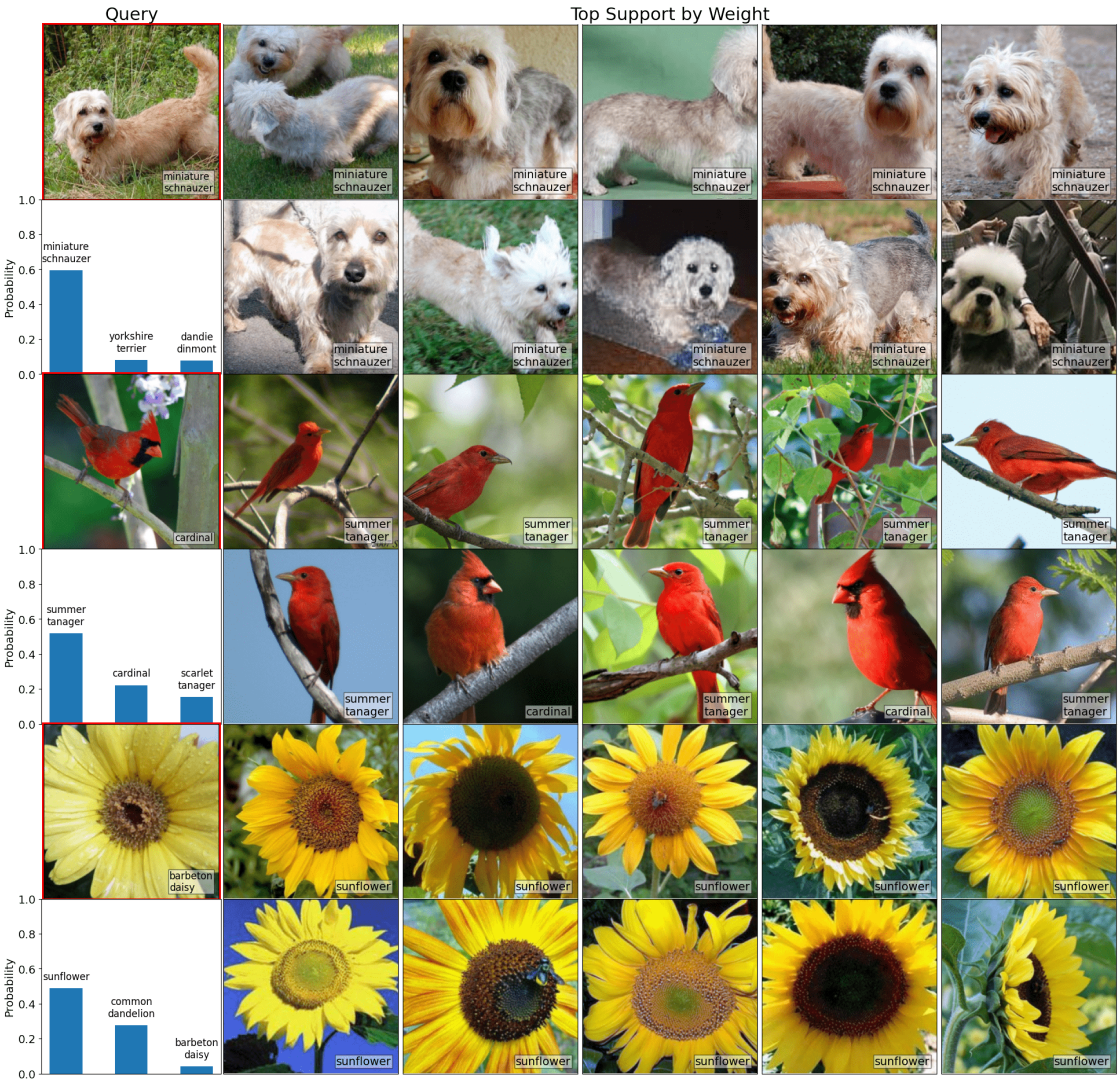


Figure 5: Examples of correctly (dog) and incorrectly (bird and flower) classified query images and their top support images ranked by w. Ordering is left to right, top to bottom.

The bird and flower queries are the same query images from Fig. 5, this time ranked by influence. For the bird, we see that helpful and harmful support images separates the two classes that had the highest weight and were confusing the model in Fig. 5. For the flower, we notice that the most helpful support images are from the correct class but have different colors, whereas the images which the model thinks are most similar to the query are in fact the most harmful. From this analysis, it is evident that the model is using the shape and color to reason about the query image, which are confounding features for this query.

5 Conclusion

We presented the Nadaraya-Watson (NW) head as a general-purpose, flexible, and interpretable building block for deep learning. The NW head predicts the label for a given query by taking the weighted sum of labels in a support set. We show that the NW head can be an efficient replacement for the fully-connected layer and offer good accuracy with excellent calibration. Additionally, the NW head is interpretable in the sense that the weights correspond to the contribution of each support image on the query image. Finally, we

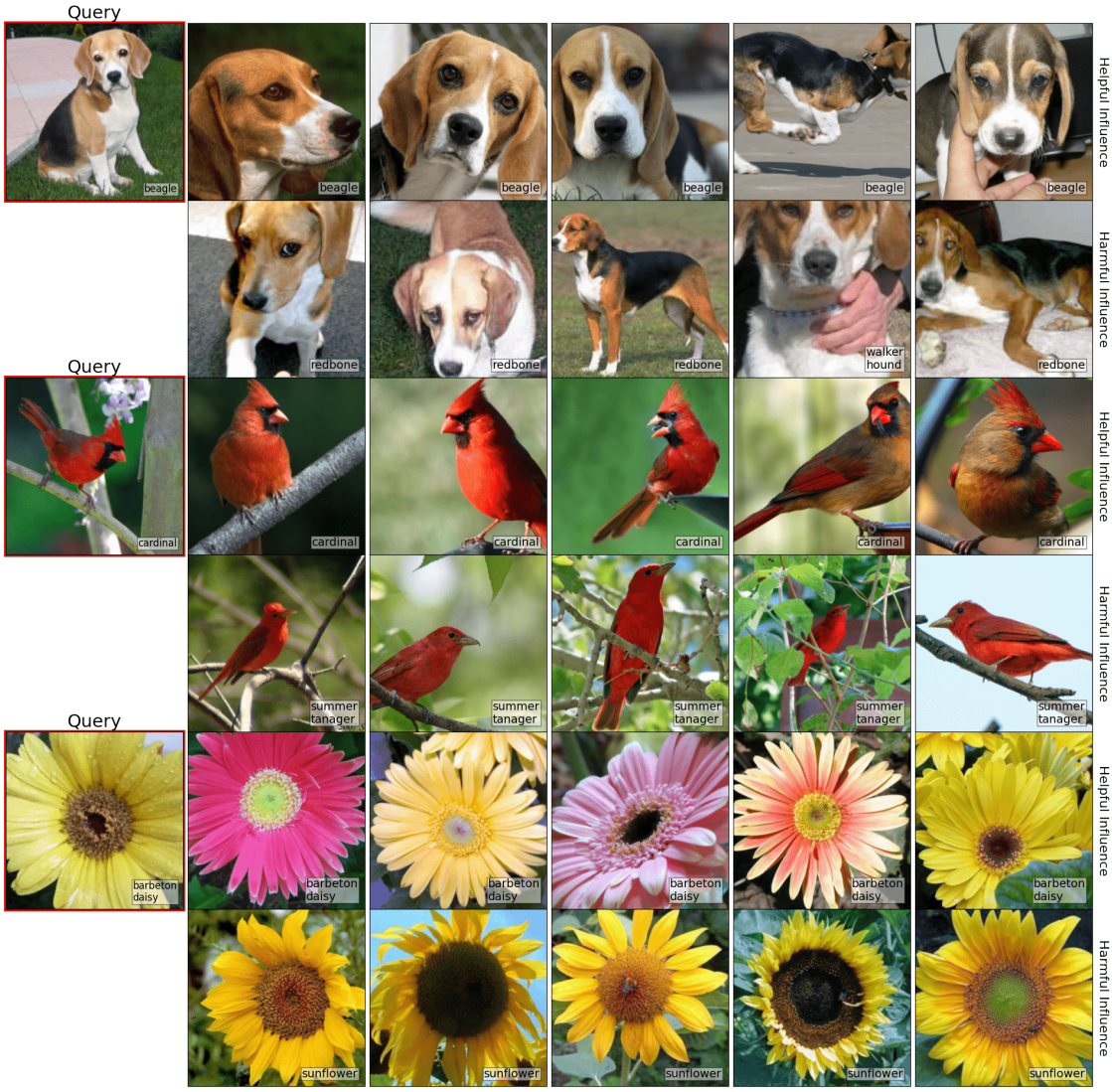


Figure 6: Query images and support images separated by most helpful influence (top row) and most harmful influence (bottom row).

define support influence, and show that this can be used as a tool to explain model behavior by highlighting helpful and harmful support images.

References

Samyadeep Basu, Philip Pope, and Soheil Feizi. Influence functions in deep learning are fragile, 2020.

Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006. ISBN 0387310738.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W.

Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens, 2021. URL https://arxiv.org/abs/2112.04426.

Thang D. Bui, Daniel Hernández-Lobato, Yingzhen Li, José Miguel Hernández-Lobato, and Richard E. Turner. Deep gaussian processes for regression using approximate expectation propagation, 2016. URL https://arxiv.org/abs/1602.04133.

Chaofan Chen, Oscar Li, Chaofan Tao, Alina Jade Barnett, Jonathan Su, and Cynthia Rudin. This looks like that: Deep learning for interpretable image recognition. 2018. doi: 10.48550/ARXIV.1806.10574. URL https://arxiv.org/abs/1806.10574.

Wei Chen, Yu Liu, Weiping Wang, Erwin Bakker, Theodoros Georgiou, Paul Fieguth, Li Liu, and Michael S. Lew. Deep learning for instance retrieval: A survey, 2021. URL https://arxiv.org/abs/2101.11282.

R. D. Cook and S Weisberg. Residuals and influence in regression, 1982.

Andreas Damianou and Neil D. Lawrence. Deep gaussian processes. In Carlos M. Carvalho and Pradeep Ravikumar (eds.), *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, volume 31 of *Proceedings of Machine Learning Research*, pp. 207–215, Scottsdale, Arizona, USA, 29 Apr–01 May 2013. PMLR. URL https://proceedings.mlr.press/v31/damianou13a.html.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2020. URL https://arxiv.org/abs/2010.11929.

Rasool Fakoor, Pratik Chaudhari, Jonas Mueller, and Alexander J. Smola. Trade: Transformers for density estimation, 2020. URL https://arxiv.org/abs/2004.02441.

Nicholas Frosst, Nicolas Papernot, and Geoffrey Hinton. Analyzing and improving representations with the soft nearest neighbor loss, 2019. URL https://arxiv.org/abs/1902.01889.

J. Ghosh and A. Nag. *An Overview of Radial Basis Function Networks*, pp. 1–36. Physica-Verlag HD, Heidelberg, 2001. ISBN 978-3-7908-1826-0. doi: 10.1007/978-3-7908-1826-0_1. URL https://doi.org/10.1007/978-3-7908-1826-0_1.

Qingji Guan, Yaping Huang, Zhun Zhong, Zhedong Zheng, Liang Zheng, and Yi Yang. Diagnose like a radiologist: Attention guided convolutional neural network for thorax disease classification, 2018. URL https://arxiv.org/abs/1801.09927.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks, 2017. URL https://arxiv.org/abs/1706.04599.

Kazuaki Hanawa, Sho Yokoi, Satoshi Hara, and Kentaro Inui. Evaluation of similarity-based explanations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=9uvhpyQwzM_.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks, 2016. URL https://arxiv.org/abs/1608.06993.

Ahmet Iscen, Jack Valmadre, Anurag Arnab, and Cordelia Schmid. Learning with neighbor consistency for noisy labels, 2022. URL https://arxiv.org/abs/2202.02200.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. Perceiver: General perception with iterative attention, 2021. URL https://arxiv.org/abs/2103.03206.

Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions, 2017.

Jannik Kossen, Neil Band, Clare Lyle, Aidan N. Gomez, Tom Rainforth, and Yarin Gal. Self-attention between datapoints: Going beyond individual input-output pairs in deep learning, 2021. URL https://arxiv.org/abs/2106.02584.

Steinar Laenen and Luca Bertinetto. On episodes, prototypical networks, and few-shot learning, 2020. URL https://arxiv.org/abs/2012.09831.

Xuhong Li, Haoyi Xiong, Xingjian Li, Xuanyu Wu, Xiao Zhang, Ji Liu, Jiang Bian, and Dejing Dou. Interpretable deep learning: Interpretation, interpretability, trustworthiness, and beyond, 2021. URL https://arxiv.org/abs/2103.10689.

S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft, 2013.

E. A. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964. doi: 10.1137/1109020. URL https://doi.org/10.1137/1109020.

Mahdi Pakdaman Naeini, Gregory F. Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pp. 2901–2907. AAAI Press, 2015. ISBN 0262511290.

Timothy Nguyen, Zhourong Chen, and Jaehoon Lee. Dataset meta-learning from kernel ridge-regression, 2021.

Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

Nicolas Papernot and Patrick McDaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning, 2018. URL https://arxiv.org/abs/1803.04765.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer, 2018. URL https://arxiv.org/abs/1802.05751.

Devendra Singh Sachan, Siva Reddy, William Hamilton, Chris Dyer, and Dani Yogatama. End-to-end training of multi-document reader and retriever for open-domain question answering, 2021. URL https://arxiv.org/abs/2106.05346.

Hugh Salimbeni and Marc Deisenroth. Doubly stochastic variational inference for deep gaussian processes, 2017. URL https://arxiv.org/abs/1705.08933.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. One-shot learning with memory-augmented neural networks, 2016. URL https://arxiv.org/abs/1605.06065.

Jake Snell, Kevin Swersky, and Richard S. Zemel. Prototypical networks for few-shot learning, 2017. URL https://arxiv.org/abs/1703.05175.

Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Philip H. S. Torr, and Timothy M. Hospedales. Learning to compare: Relation network for few-shot learning, 2017. URL https://arxiv.org/abs/1711.06025.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision, 2015. URL https://arxiv.org/abs/1512.00567.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning, 2016.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset, 2011.

Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaoou Tang. Residual attention network for image classification, 2017. URL https://arxiv.org/abs/1704.06904.

G. S. Watson. Smooth regression analysis. *Sankhya: The Indian Journal of Statistics*, Series A(26):359–372, 1964.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep kernel learning, 2015. URL https://arxiv.org/abs/1511.02222.

Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance-level discrimination, 2018. URL https://arxiv.org/abs/1805.01978.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention, 2015. URL https://arxiv.org/abs/1502.03044.

Dequan Zhang, Ning Zhang, Nan Ye, Jianguang Fang, and Xu Han. Hybrid learning algorithm of radial basis function networks for reliability analysis. *IEEE Transactions on Reliability*, 70(3):887–900, 2021. doi: 10.1109/TR.2020.3001232.

Quanshi Zhang, Yu Yang, Haotian Ma, and Ying Nian Wu. Interpreting cnns via decision trees, 2018. URL https://arxiv.org/abs/1802.00121.

A Appendix

A.1 Derivation of Support Influence

We derive the expression in Eq. (8). For brevity, let $f = f(x, \mathcal{S})$ and $f_- = f(x, \mathcal{S}_{-z_s})$.

$$\begin{aligned}\mathcal{I}(z, z_s) &= L(f_-, y) - L(f, y) = \sum_{c=1}^{C} \bar{y}^c \log f^c - \sum_{c=1}^{C} \bar{y}^c \log f_-^c \\ &= \log f^y - \log f_-^y.\end{aligned}$$

If $y = y_s$, then $w(x, x_s)$ must be subtracted from f^y and re-normalized to a valid probability. Otherwise, f^y is simply re-normalized:

$$f_-^y = \begin{cases} \frac{f^y - w(x,x_s)}{1-w(x,x_s)}, & \text{if } y = y_s \\ \frac{f^y}{1-w(x,x_s)}, & \text{else.} \end{cases}$$

Thus,

$$\begin{aligned}\mathcal{I}(z, z_s) &= \log f^y - \log \left(\frac{f^y - w(x, x_s)\mathbb{1}_{\{y=y_s\}}}{1 - w(x, x_s)} \right) \\ &= \log \left(\frac{f^y - f^y w(x, x_s)}{f^y - w(x, x_s)\mathbb{1}_{\{y=y_s\}}} \right).\end{aligned}$$

A.2 Connection between Support Influence and Weights

We prove that ranking by helpful/harmful examples via the support influence is equivalent to ranking by the weights for correct and incorrect classes. Let us start from the definition of support influence in Eq. (8):

$$\mathcal{I}(z, z_s) = L(f(x, \mathcal{S}_{-z_s}), y) - L(f(x, \mathcal{S}), y) = \log \left(\frac{f^y - f^y w(x, x_s)}{f^y - w(x, x_s)\mathbb{1}_{\{y=y_s\}}} \right).$$

where $0 \leq w \leq f^y \leq 1$. Note that $w \leq f^y$ because they are related by Eq. (3).

Separate into two cases:

$$\mathcal{I}(z, z_s) = \begin{cases} \log \left(\frac{1-w(x,x_s)}{1-w(x,x_s)/f^y} \right), & \text{if } y = y_s \\ \log(1 - w(x, x_s)), & \text{else.} \end{cases}$$

In the first case where $y = y_s$, $\mathcal{I}(z, z_s)$ increases as w increases, with the constraints on w and f^y.

In the second case where $y \neq y_s$, $\mathcal{I}(z, z_s)$ decreases as w increases.

In words, the support influence increases as the weight increases for the correct class, and decreases as the weight increases for the incorrect class. Thus, the support influence can be interpreted as simply ranking by the weights for correct and incorrect classes.

A.3 More NW Visualizations

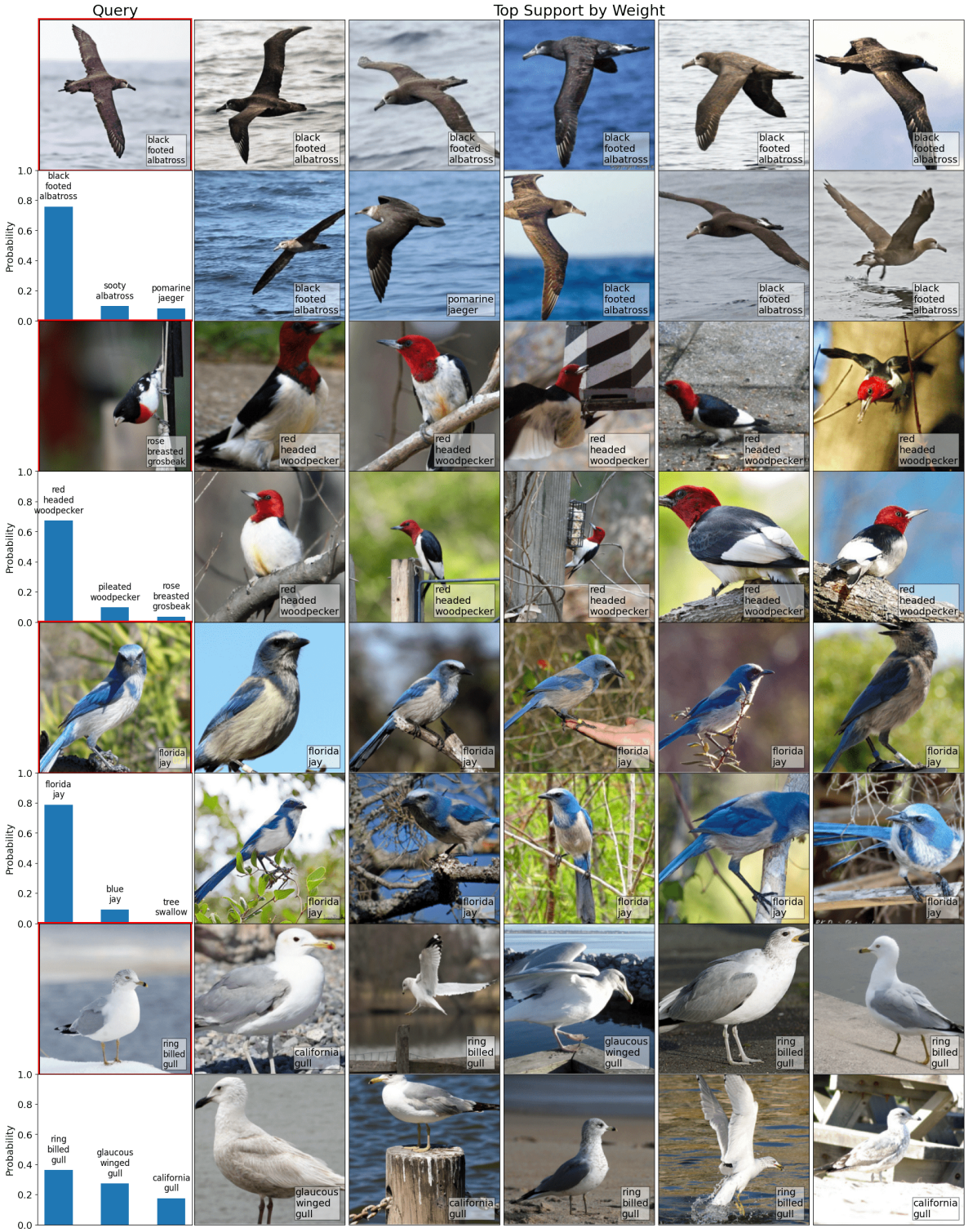


Figure 7: Query image, top 3 predicted probabilities, and top support ranked by weight for Bird dataset.

Figure 8: Query image, top helpful support images, and top harmful support images ranked via support influence for Bird dataset.

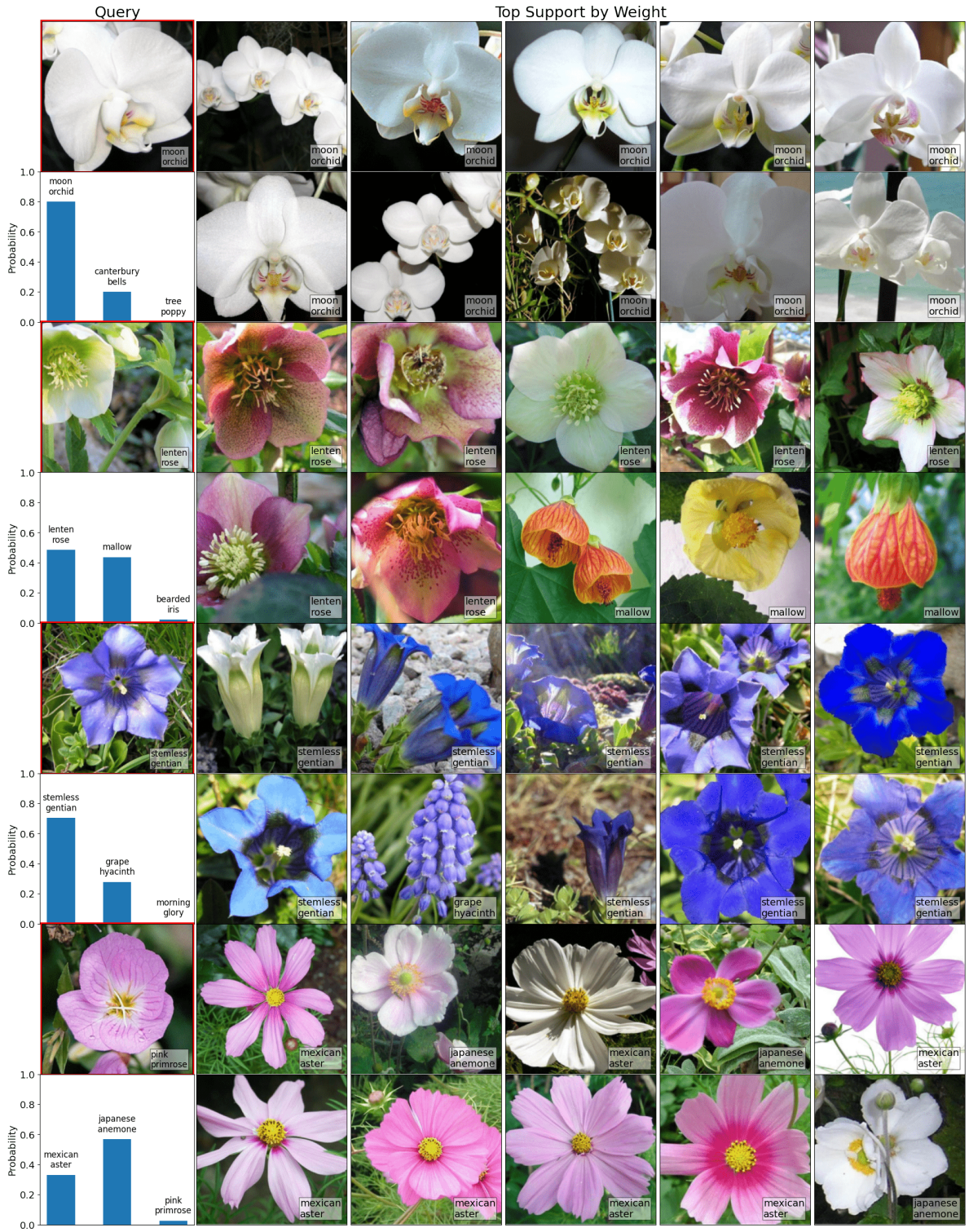


Figure 9: Query image, top 3 predicted probabilities, and top support ranked by weight for Flower dataset.

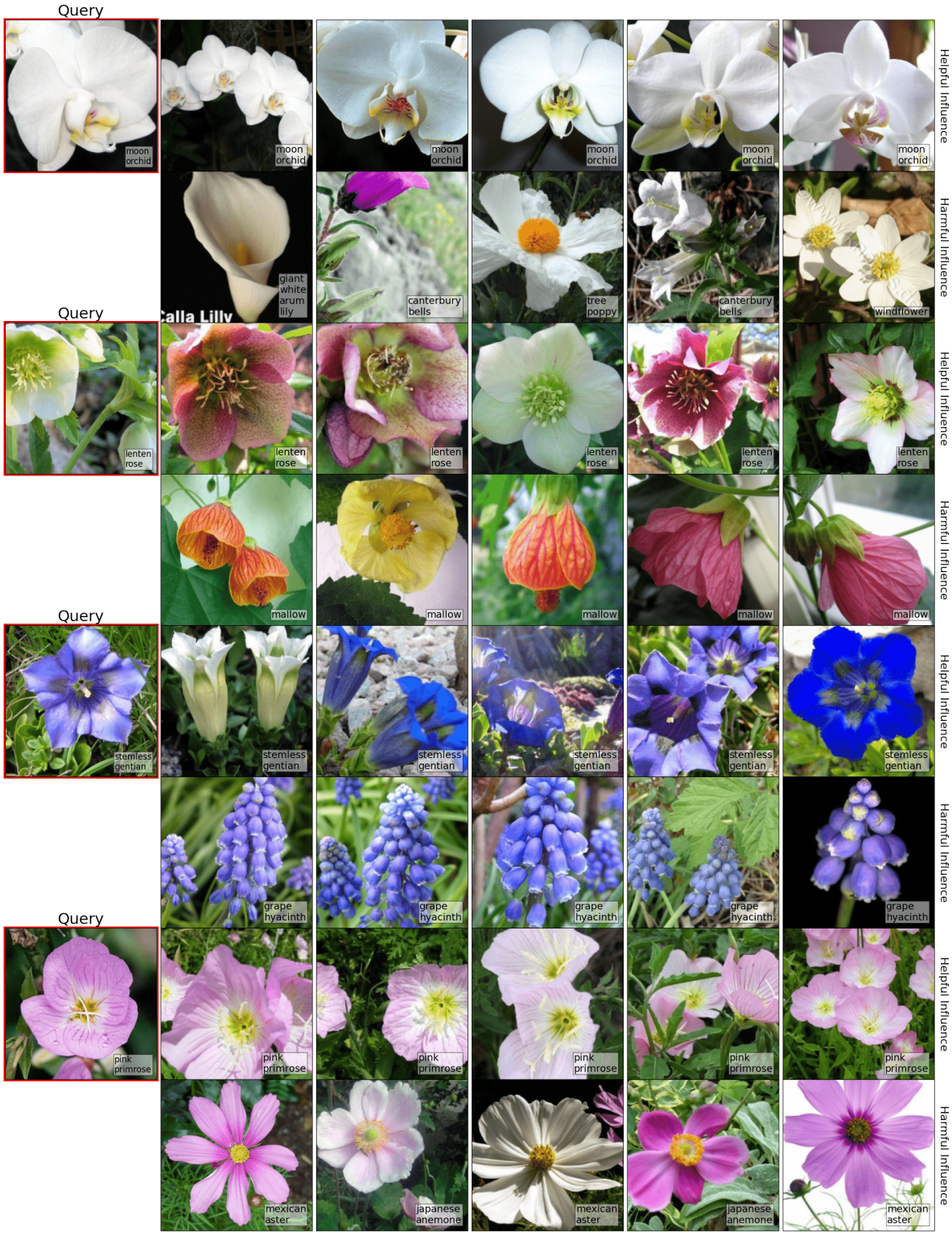


Figure 10: Query image, top helpful support images, and top harmful support images ranked via support influence for Flower dataset.

Figure 11: Query image, top 3 predicted probabilities, and top support ranked by weight for Dog dataset.

Figure 12: Query image, top helpful support images, and top harmful support images ranked via support influence for Dog dataset.

A.4 Ranking FC Images By Euclidean Distance in Feature Space

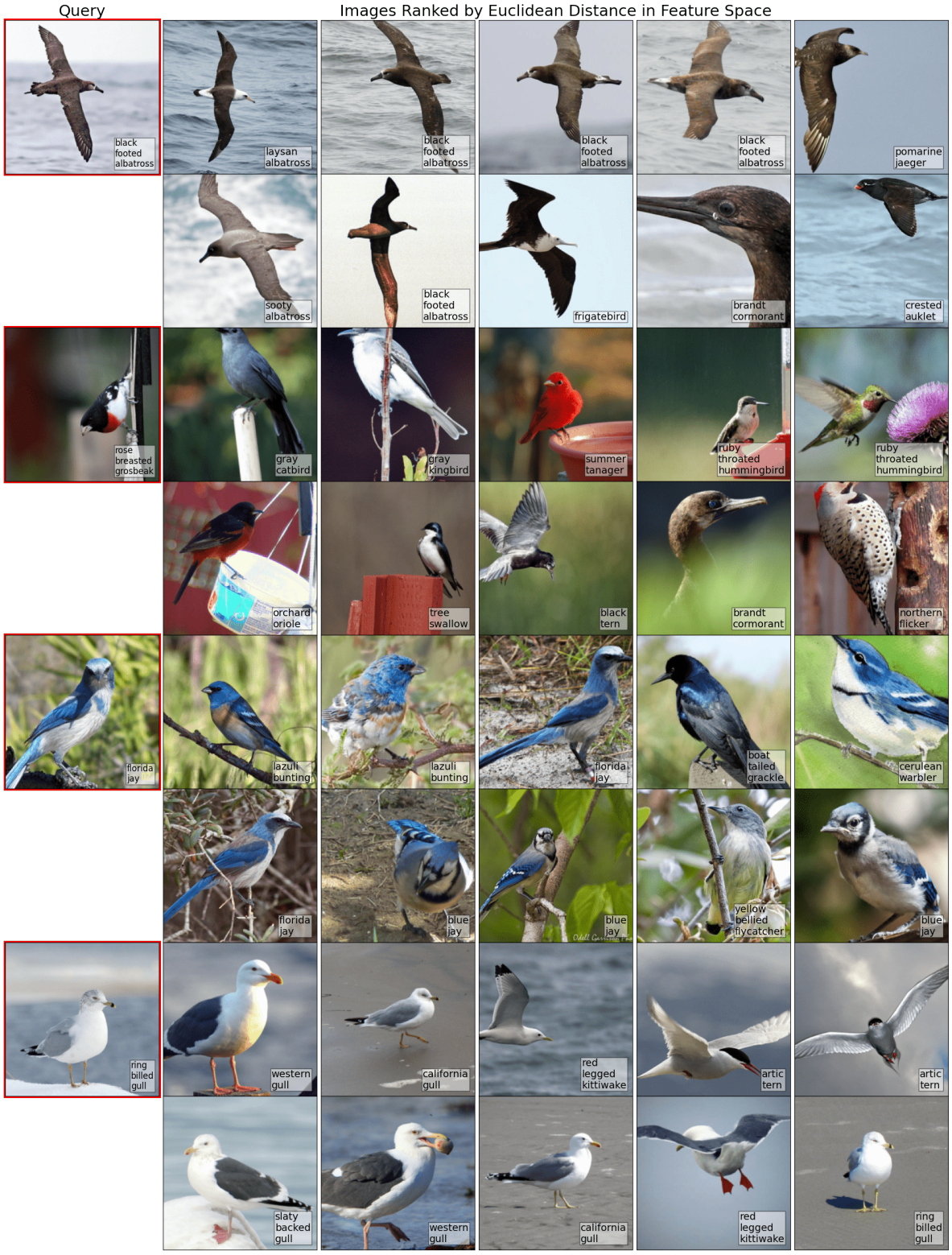


Figure 13: Query image and top images ranked by Euclidean distance in feature space of FC model for Bird dataset. The degree of semantic similarity between the query and the top images is lower than the NW model (see differences in class labels).

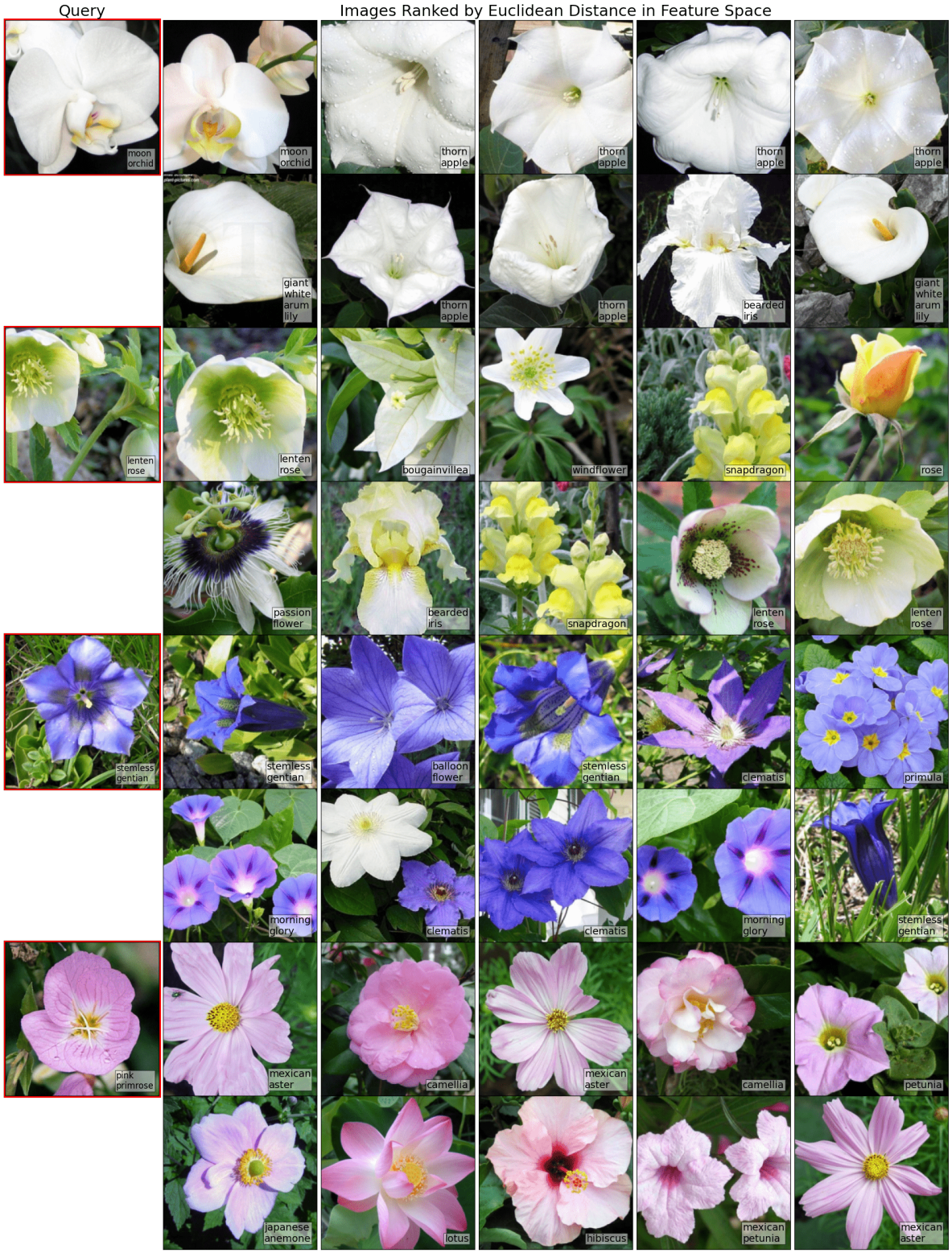
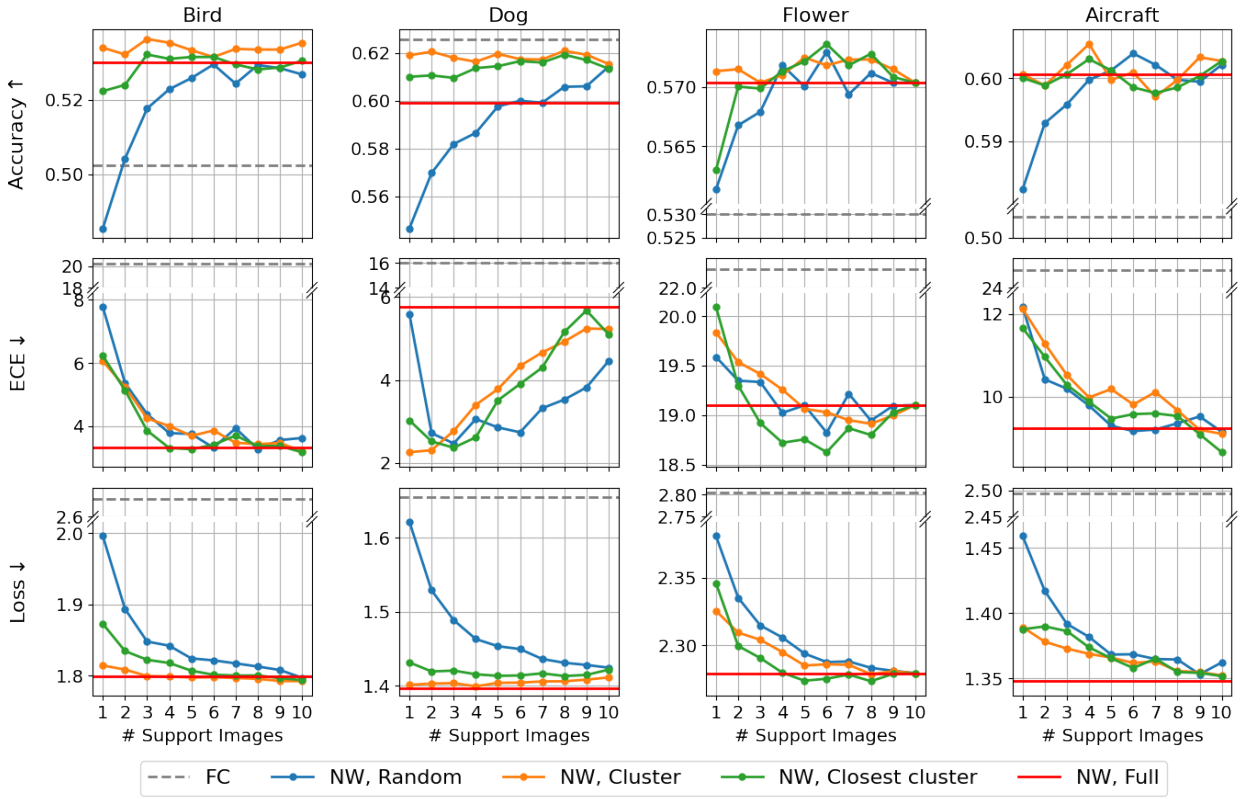
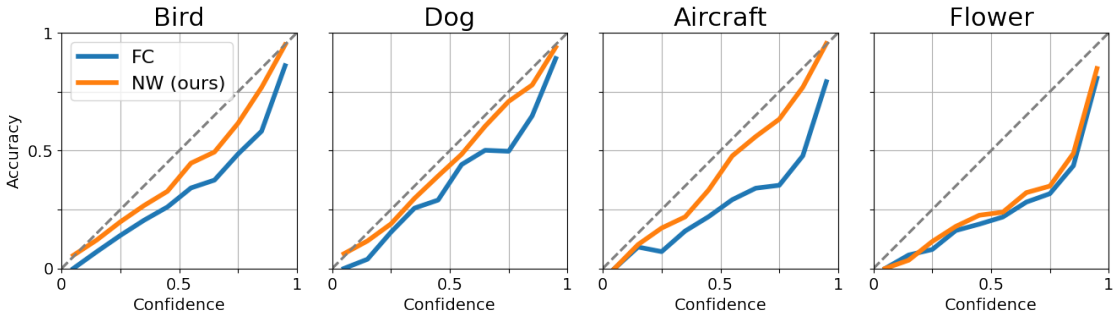


Figure 14: Query image and top images ranked by Euclidean distance in feature space of FC model for Flower dataset. The degree of semantic similarity between the query and the top images is lower than the NW model (see differences in class labels).

Figure 15: Query image and top images ranked by Euclidean distance in feature space of FC model for Dog dataset. The degree of semantic similarity between the query and the top images is lower than the NW model (see differences in class labels).

A.5 Performance vs. support set size on ResNet-18



A.6 Reliability diagram on DenseNet-121