Checkpoint 3

Machine learning models often rely on spurious correlations, where irrelevant features of training data become strongly correlated with the target labels, leading to both incorrect decisions and poor generalization to new data. Our goal is to determine whether concepts can be used to effectively identify and debug these spurious correlations, ultimately improving both model robustness and interpretability. A concept, in this context, is an interpretable feature or pattern that the model learns, such as a specific texture, color, or shape, which contributes to its decision-making process. Specifically, we are examining how concept-based methods can help distinguish between *relevant* and *irrelevant* features that influence model predictions. This will involve designing and generating concepts, extracting these concepts, and quantitatively assessing their impact.

In this checkpoint, our work is structured into several sections. First, we focus on designing and generating concepts, which involves defining interpretable features that the model could use during classification. In **Section II**, we discuss the extraction of these concepts from a trained model. In **Section III**, we develop a framework for creating both baseline and buggy models that can be used in experiments to systematically identify and analyze spurious correlations. In **Section IV**, we propose a framework to quantitatively evaluate the model's sensitivity to such concepts, using the framework to assess the influence of each concept and determine whether the model is relying on spurious features.

I. Concept Definition and Generation

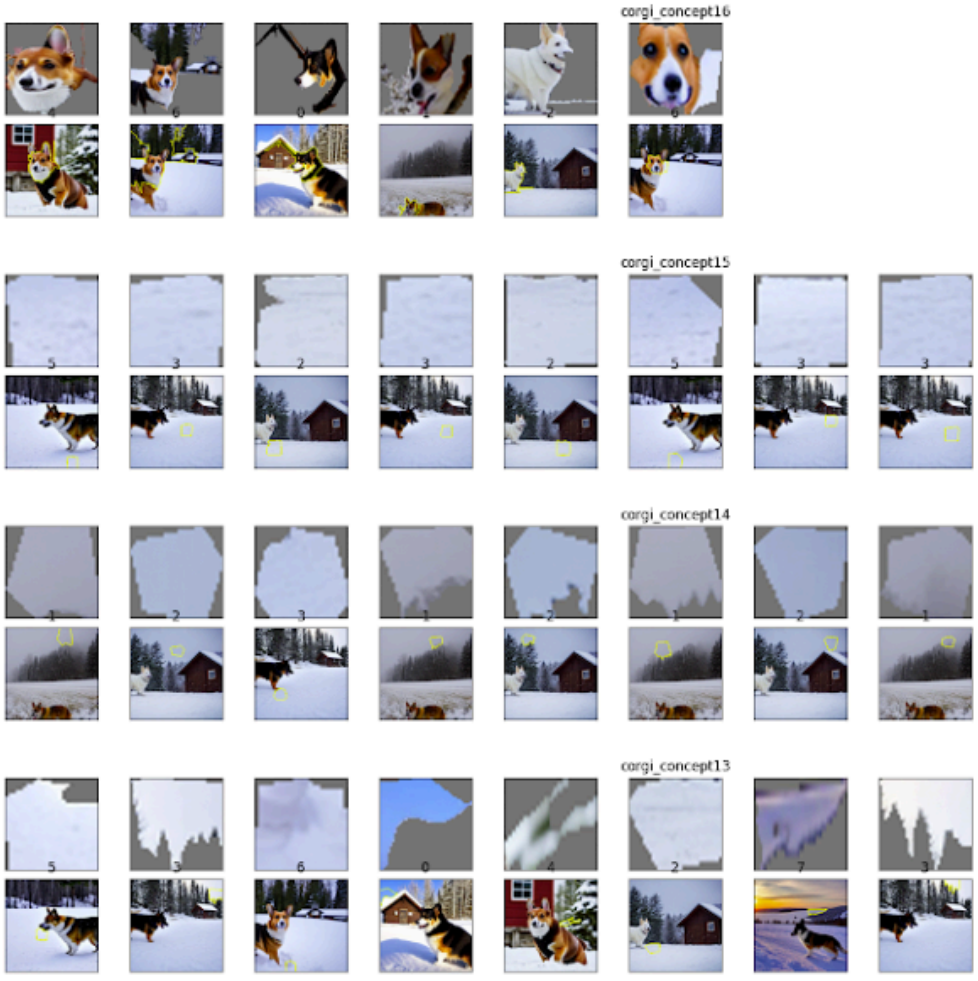
In our previous checkpoint, we outlined several methods for defining and generating concepts within machine learning models. As a reminder, concepts are *interpretable* features or patterns that the model has learned, which can be extracted to understand how they influence predictions. The process of concept generation involves breaking down the model into a concept extractor (the first n-1 layers) and a classifier (the final layer). By creating embeddings from image classes and applying dimensionality reduction techniques, we define concepts that represent shared features among images.

II. Extracting Concepts

Per the last milestone, we had a working implementation of CRAFT (Fel et al.). Similar to feature attribution methods, however, there are multiple ways to extract concepts. At the start of this project, we understood that our results should be reproducible using multiple concept-generation methods. **In this milestone, we implemented additional automatic concept extraction methods.**

Many automatic concept methods work similarly: they divide the model into a concept extractor (the first *n-1* layers) and a classifier (the final layer). Both new methods use the first *n-1* layers to (A) create embeddings of a class of images, then (B) apply some dimensionality reduction method to define concepts shared between the images. The importance of those concepts is deduced using the classifier weights.

1. Automatic Concept-based Explanations (ACE) (Ghorbani et al 2019)
    a. This method behaves differently, as it first creates differently-sized patches of the image using image segmentation techniques. These patches are passed through the concept extractor of the model and are grouped into concepts via k-means clustering. The centroid of the cluster is the representative of the cluster. Importance scores are derived from the final layer weights of the model.
    b. This method has been **fully implemented, and is ready for testing.** A small-scale example of a few concepts is shown below:



2. Invertible Concept-based Explanations (ICE) (Zhang et al 2021)
    a. This paper explores using PCA or Non-negative Matrix Factorization (NMF) to obtain concepts. We will test out both in the time before our final submission.

III. Framework for Evaluating Buggy Models

In this section, we describe the development of a framework that allows us to evaluate the impact of spurious concepts on a model's performance. The primary goal is to establish a systematic method for assessing which concepts are contributing to flawed decision-making and how to effectively debug these models. This evaluation phase allows us to ensure that we have a truly "buggy" model and a robust baseline, both of which are required to rigorously test our concept-based debugging methods in the next steps. This will require developing models in the following parts:

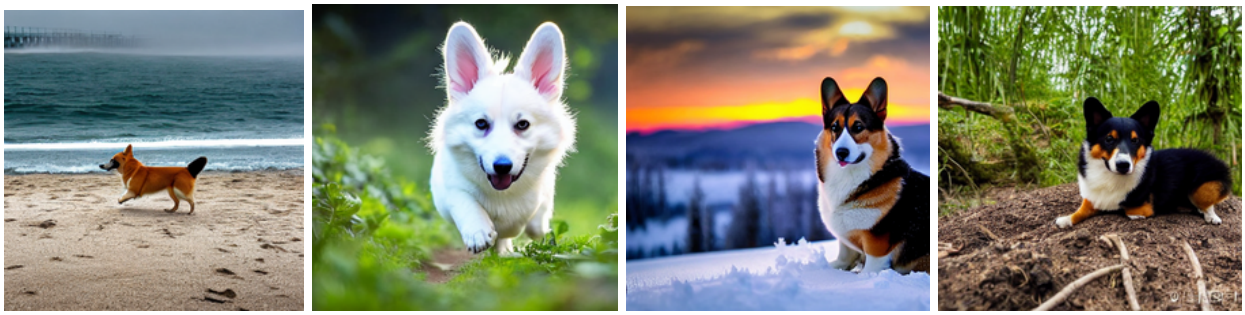
1. **Baseline Model:** The baseline model helps establish a reference for how the model *should* behave without spurious correlations, showing the ideal sensitivity to **relevant** concepts rather than irrelevant concepts. This will require working with a dataset with a balanced set of heterogeneous backgrounds.

2. **Buggy Model:** This model will measure the influence of spurious correlations on the model by comparing its concept sensitivity to the baseline model. A higher sensitivity to irrelevant concepts (like background) in the model indicates that it has learned spurious correlations. This will require working with a dataset with homogenous backgrounds, i.e. the backgrounds do not change across images.

3. **Validate Baseline and Buggy Model.** In order to evaluate the baseline and buggy model, we feed a BACKGROUND-ONLY dataset into both models and evaluate their accuracy. For the baseline model, we would expect to see low accuracy, as ideally it has not learned to associate backgrounds with specific classes. The buggy model may achieve a higher accuracy, as it has learned to associate backgrounds with classes, showing a reliance on spurious correlations.

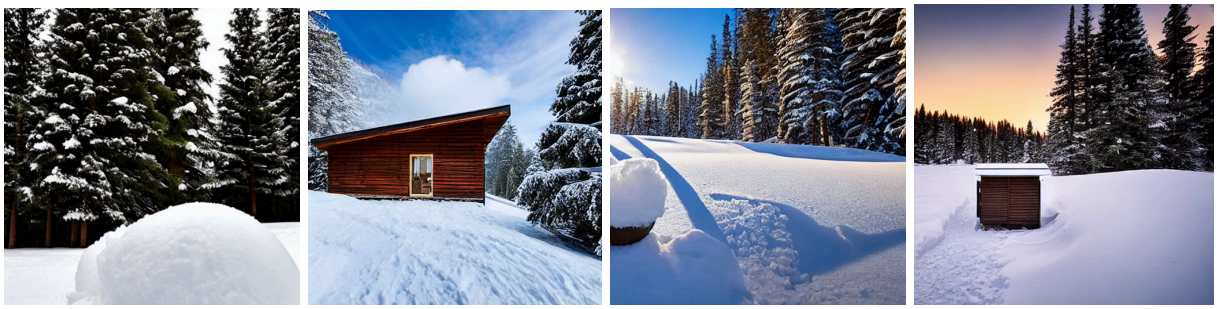
To implement this, we gathered the following datasets (1) corgis on a range of backgrounds, (2) corgis on snow backgrounds *only*, (3) snow background-only scenes (no foreground subject). Examples are shown below:

Corgi on Range of Backgrounds:



Corgi in Snow:

Snow Only:



The results of the model when feeding it the background-only dataset is as follows:

	Baseline Model	Buggy Model
Test Accuracy (%)	38.72%	73.21%

The tests for both the baseline and buggy models demonstrate promising but imperfect results. The baseline model, trained on the corgi-on-a-range-of-backgrounds dataset, shows lower accuracy, indicating

a level of generalization with fewer spurious correlations, but it could be further improved by introducing even more background diversity to lower its accuracy on the snow-only dataset and reduce overfitting. Thus, for the next iteration, we will focus on creating a more enriched dataset to improve robustness for the baseline model. Meanwhile, the buggy model, trained on corgis only in snow backgrounds, demonstrates higher accuracy, suggesting it may be leveraging spurious correlations by relying heavily on background features rather than focusing on the corgi itself.

However, the mid-range accuracy values for both the baseline and buggy models highlights the need to refine the training process to achieve a truly buggy model and a more generalized baseline. In the next iteration, we plan to expand the baseline training dataset with more diverse and balanced backgrounds and optimize the training process. To create a truly buggy model, we plan to focus on training with more limited, repetitive backgrounds that closely resemble the snow backgrounds with the corgi's present, which will further encourage the model to rely on spurious features. Thereafter, these models can then be used to assess the true contribution of learned concepts (outlined in concept generation, Part I) to debug the models quantitatively, as described in Part II.

IV. Quantify Concept Sensitivity

In this section, we use the framework developed in Section III to quantitatively evaluate a model's sensitivity to the identified concepts. The goal is to determine how much specific concepts influence the model's output and whether these concepts are spurious or genuinely important to the classification task. To achieve this, we use two datasets: Database A, which contains images of animals along with their respective backgrounds (e.g., corgis in the snow), and Database B, which contains only the backgrounds (e.g., snow without any animals).

1. Database A has both animals and their respective backgrounds, e.g. corgis in the snow. Database B has just the backgrounds, e.g. snow. Run the model first on Database A, then on Database B. Use NMF to determine concepts detected by the model for both A and B.
2. Within a given class, create two sets of concepts: *C* and *C'*. *C* is the set of concepts detected from Database A, and *C'* is the set of concepts detected from Database B. Consider the 'corgis in the snow' example:
    a. Essentially, *C* - *C'* becomes a set of concepts that are particular to corgis; that is, the set of concepts we would like to be attributed to a corgi classification (since the image background should not alter a corgi classification).
    b. *C'* becomes a set of concepts that could be spuriously attributed to corgis.

More Complicated SS(*k*) Calculation Method:

- For each concept $k \in C$, calculate its average importance score $\text{avg}(I_C(k))$. For each concept $k \in C'$, calculate its average importance score $\text{avg}(I_{C'}(k))$. For the next step, $\max(I_{C'}(k))$ is defined as the maximum importance score $I_{C'}(k)$ that concept *k* achieved in Database B.
- We define a spurious score for a concept *k* (SS(*k*)) as the following: $[\max(I_{C'}(k))]^2 \div \text{avg}(I_C(k))$. Consider the following implications of this score:
    - A concept *k* can either occur in both databases (pertains to the snowy background) or just Database A (pertains to the corgi).
    - If we have found a concept *k* that is detected in Database A and not Database B (belonging to the set $C - C'$), then the concept is very likely particular to corgis. $\max(I_{C'}(k))$ will equal 0 because this concept is not present in C'. Therefore, the concept's spurious score will be 0, correctly correlating with its extremely strong likelihood to be a corgi-specific (and non-spurious) concept.
    - If we have found a concept *k* that is detected in Database A and Database B (belonging to the set C'), $\max(I_{C'}(k))$ can either be 0 or nonzero. If $\max(I_{C'}(k))$ is 0, *k* cannot be a spuriously correlated concept (and will therefore have a SS of 0) because, even though we are detecting a concept that is unrelated to corgis, it has no importance to the model's output. If $\max(I_{C'}(k))$ is nonzero, *k* could be a spuriously correlated concept, but its spurious nature is dependent both on its difference in importance across Databases A and B and the general scale of its importance.
        - Let us rewrite SS(*k*) as $[\max(I_{C'}(k)) \div \text{avg}(I_C(k))] \bullet \max(I_{C'}(k))$ to represent that we are both accounting for *k*'s relative importance in Database B compared to Database A ($\max(I_{C'}(k)) \div \text{avg}(I_C(k))$) and *k*'s general importance ($\max(I_{C'}(k))$). Say that our first term is 1: the concept is equally important across two databases, but we are further ranking likeliness for spuriousness by multiplying by the extra term $\max(I_{C'}(k))$.

Less Complicated SS(*k*) Calculation Method:

- We know that *C'* houses all potentially spurious concepts. For each concept $k \in C'$, calculate its average importance score avg($I_{C'}(k)$). We will call this score SS(k).

3. Rank concepts in order of highest to lowest SS in list L. Input your desired classification accuracy x. For L_j, remove concept $L_j(k)$ from the model's consideration and test the image classification accuracy x' on Database A; if $x' > x$, then append k to a set of spurious concepts S. Repeat for every nonzero item in list L until the first item that is not appended to S is reached. Or, we could just arbitrarily determine some value τ for which k is a spurious concept if SS(k) > τ.)

References:

1. Adebayo, Julius, et al. "Debugging tests for model explanations." arXiv preprint arXiv:2011.05429 (2020).
2. Ghorbani et al. "Towards Automatic Concept-based Explanations." *NeurIPS,* 2019.
3. Fel, Thomas, et al. "CRAFT: Concept Recursive Activation FacTorization for Explainability." arXiv preprint arXiv:2211.10154 (2022).
4. Kim, Been, et al. "Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV)." *International Conference on Machine Learning (ICML)*, 2018, https://doi.org/10.48550/arXiv.1711.11279.
5. Lynch, Aengus, et al. "Spawrious: A Benchmark for Fine Control of Spurious Correlation Biases." arXiv preprint arXiv:2303.05470 (2023).
6. Zhang et al. "Invertible Concept-based Explanations for CNN Models with Non-negative Concept Activation Vectors." 2021, https://doi.org/10.48550/arXiv.2006.15417.