Are important features more interpretable?

Mikhail Terekhov
Final project for the AGISF course

1 Introduction

The idea for this project was born out of Stephen Casper's "Engineer's Interpretability Sequence". In part VI [2], he critisizes current mechanistic interpretability approaches along several directions. One of his arguments is that feature visualizations in the papers are often cherry-picked, while features "in the wild" are more often non-interpretable. This claim can be considered in conjunction with the end goal of mechanistic interpretability, which is to extract an algorithm that underlies the neural network's predictive power. A prominent approach to constructing such algorithms is through *circuits* [7]. A circuit is a subgraph of the initial neural net that connects individual features in different layers in meaningful ways, composing complicated abstractions in later layers out of simpler ones in earlier layers. Authors of [7] use an example of a "car detector" being composed of a "window detector", "car body detector", and a "wheel detector". Crucially, the components of this circuit are all somewhat recognizable in the individual feature visualizations. Features in convolutional neural nets (CNNs) can be visualized using approaches from [8]. These approaches include maximizing the activation over the input image with some regularization and looking for images from the dataset that activate the feature the most. Strong negative activations can be considered as well. If the feature visualizations do not make sense to a human observer, then the feature cannot be manually put into a circuit, and the critique of [2] applies. Thus, if most features are not interpretable, circuits can only explain a minority of the network's inner workings.

In this project, I consider an alternative hypothesis:

Are the most important features for network's performance more interpretable?

If this hypothesis holds, mechanistic interpretability research is more promising: even though the majority of features are not interpretable, they can be discarded without significant performance degradation. Otherwise, we have less hope: interpretable features are a minor artifact and the network is still mostly opaque.

Some evidence for the possibility of this hypothesis is provided by the success of *sparsification* of neural networks. As it turns out, the majority of features can indeed be pruned from a CNN without a performance drop [1]. This still leaves open the question of whether the remaining features are the interpretable ones. Sparsity of the network can be further enforced through regularization. In [6], authors train toy neural networks with methods inspired by neuroscience: regularization cost for a weight is set proportionally to the distance between assigned locations of the neurons this weight connects. Apart from sparsity, this approach encourages *modularity*. That is, the sets of remaining neuron connections tend to neatly decompose into subgraphs with clear individual functions. Here, the few remaining neurons

are also more interpretable. For example, in a regression task with a known formula, some neurons were shown to correlate to the values of subexpressions in the target formula.

The scope of this project limited me to considering the simplest notion of "interpretability". I describe the method in detail below, but the main idea is to rate the visualizations as recognizable or not by hand. One issue with this approach is that it precludes me from studying large samples of features. Another comes with the *superposition phenomenon* [4], also known as *entanglement* or *polysemanticity*. If many real-world concepts get superposed in the low-dimensional feature space, visualizations would not be as interpretable, and dataset examples would appear incoherent.

2 Methodology

I chose InceptionV1 [9] as my model network. While its features are comparatively interpretable, it still has a significant proportion of unclear visualizations, espectially at the later layers. I focused on the layer `mixed5a`. The authors of [8] describe this layer in the appendix as "Visualizations become harder to interpret here, but the semantic concepts they target are often still quite specific." This is a good match for my interpretability measure, which combines coherency of examples and the way they are represented on visualizations.

To evaluate the network and visualize its features, I used the Lucid library[1]. It was written for TensorFlow 1, but I found a working fork that migrated it to TensorFlow 2[2]. Lucid comes with several pre-trained networks, including the desired InceptionV1. My computational resources were limited to my laptop, so to evaluate the max-activating samples and neuron importance I had to limit myself to a small subset of ImageNet [3]. The official ImageNet website[3] provides a validation set of 64×64 images for classification. It contains 1000 classes, with 50 images per class. InceptionV1 has a receptive field of 224×224, so to evaluate it on the images I bilinearly upscaled the images to this resolution. Note that the original network was applied to patches of bigger images on multiple scales (144 patches were used per image!), and results from the patches were aggregated into the final prediction about the image. These issues provided siginficant challenges for the classifier in my limited setting, but it generalized rather well in spite of them. Compared to the original 93.4% of correct top-5 predictions, my evaluation got 65.6% of the images correctly. When top-10 classes were considered, the accuracy rose to 74.2%.

2.1 Importance

I measured importance of a feature through mean ablations [10]. The idea here is that if a feature is important, "removing" it from the computation will significantly impact the performance of the network. However, we can't literally remove the values of the feature due to dimensionality constraints. Instead, we replace its value with a constant. An intuitive idea could be to simply substitute it with a zero, but, as authors of [10] argue, this also

[1] `https://github.com/tensorflow/lucid`

[2] `https://github.com/ercaronte/lucid`

[3] `https://image-net.org/`

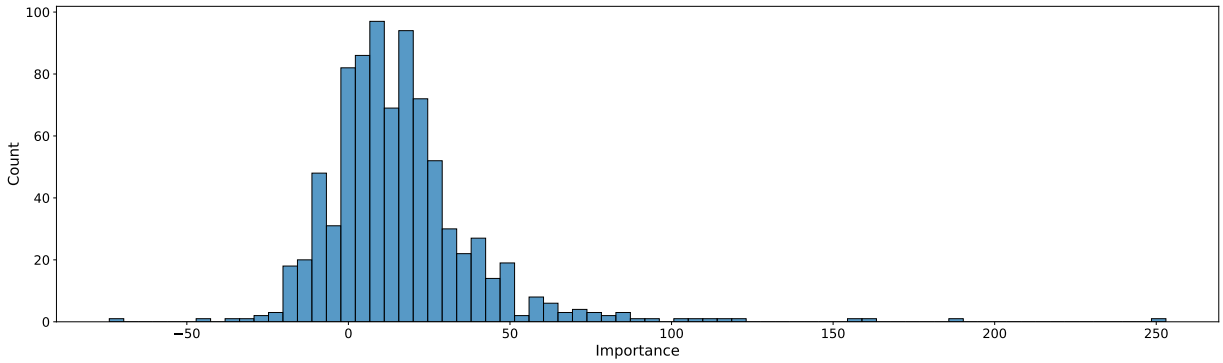


Figure 1: A histogram of importances of 832 features from `mixed5a`.

affects the follwing layers, which might expect values within a certain range. To alleviate this problem, I used their approach and instead substituted the feature values with the average over all values that this feature takes on the images from the used dataset. Thus, the variance in feature values is removed, while the range is preserved. The *importance* of the feature is defined as the difference between the number of images correctly classified (with the true class among the top-10 predictions) by the original network and the number of images correctly classified by the network with the ablated feature. This measure is very noisy, and sometimes even takes negative values.

On the implementation side, the fork of Lucid, although migrated to TF2, still used the abstractions of graphs and sessions from TF1, which has proven extremely helpful to me. As it turns out, the `Session` objects have a `partial_run` method, which can be configured to run a part of the computation graph, imputing the necessary variables. To test that imputation worked correctly, I checked that if I impute the values that the feature originally had (instead of the mean like in regular imputation), then the output of the network does not change.

I computed the importances of all 832 features from the `mixed5a` layer. The distribution of measured importance is presented in Figure 1. Since I studied one of the later layers of the network, each individual feature has a limited impact on the overall performance. Compared to 50000 images in the dataset, the mean feature importance is 16.0. It is small but positive, as one would expect.

All features were sorted according to their importance. I selected 2 groups with 40 features each to compare their interpretability. The fist one was taken from the middle of the list, resulting in the mean importance of 13.0. The second one contained the most important features according to the measure, with mean importance of 86.8.

2.2 Interpretability

I was originally planning to use OpenAI's Microscope[4], but at the time visualizations were not accessible there. So, I decided to replicate the visualizations using Lucid. As I realized

[4] `https://openai.com/research/microscope`

later, the appendix of [8] also comes with visualizations for all features of InceptionV1, not only the ones shown in the paper. However, reimplementation is still useful as it can be easily adapted to other networks supported by Lucid.

I chose to measure interpretability using a visual inspection of 4 feature visualziations with a diversity term and 8 max-activation dataset examples. These visualizations were pre-computed for all features of the `mixed5a` layer. I could check that the visualizations are reasonable by comparing to visualizations from [8]. Some examples are shown in Figure 4. We can see that the visualizations usually match diversity visualizations from the original paper (minor discrepancies are likely due to weighting the diversity versus feature maximization terms and randomness). At the same time, dataset examples do not always show matching concepts. For example, `mixed5a:4` feature is responding to flames in the original examples, but my selection fails to show this. I suspect that such occasional failures are due to the limited selection of the validation sample that I chose. I didn't use negative visualization and examples for simplicity.

The measurement metrics, detailed survey guidelines, and expected results were pre-recorded in a Google doc[5]. My metric for "interpretability" is derived from two equally weighted components. First, how predictable are the dataset examples given only the visualizations. Second, how coherent are the examples between each other. I also tried to measure some notion of "polysemanticity" in the features but in practice found that my visualizations do not allow for a clear distinction of polysemantic neurons against non-interpretable ones. To be specific, I implemented a form that shows the generated images and asks a participant several questions about what they see. First, only the four visualizations are shown. The following two questions are asked:

1. *On a scale from 1 to 5, how clear are the objects on the 4 feature visualization images? If you can't recognise any objects on the images, respond as 1:*
2. *Which words or phrases are best describing objects on the feature visualization images. List 0-4 phrases through comma:*

After the participant fills out these questions and submits, the dataset examples are revealed and the participant has the opportunity to compare the examples against the phrases that they recorded in Q.2. The following questions become available:

3. *On a scale from 1 to 5, how closely do the dataset examples resemble the phrases you put in the previous question? Respond 1 if you didn't write any phrases:*
4. *On a scale from 1 to 5, how coherent are the examples (i.e. 1=they have nothing in common, 5=they all show the same concept)?*
5. *On a scale from 1 to 5, do you agree that feature visualizations and dataset examples show 2 distinct concepts? 1 = "There clearly aren't two objects"; 5="There clearly are two objects"*

[5] `https://docs.google.com/document/d/1_xDxej78VRy_xRjoJ7SecTBaEWsnTHy4s7bSXkBqRNo/edit?usp=sharing`

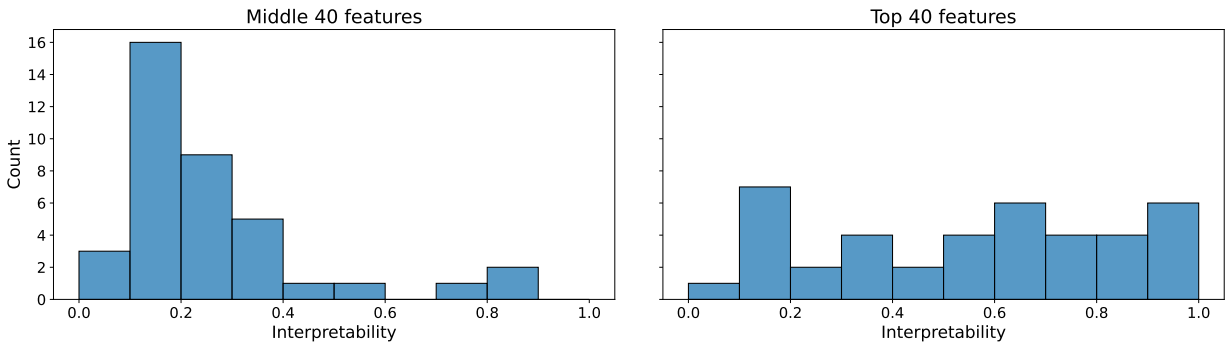


Figure 2: Histograms of interpretability for the two groups of features.

6. *On a scale from 1 to 5, do you agree that feature visualizations and dataset examples show 3 distinct concepts? 1 = "There clearly aren't three objects"; 5="There clearly are three objects"*

The interface with an example set of visualizations is presented in Figure 3. Text response to Q.2 is not used in evaluations, but it provides an accountability mechanism for the respondent to evaluate the correctness of their intuitions about the visualizations.

Let $Q_i \in \{1, \cdots, 5\}$ be the answer provided to the i-th question by the respondent. We denote $q_i = (Q_i - 1)/4$ as its normalized value. Then, my measure of interpretability is given by

$$I = \frac{q_1 q_3 + q_4}{2} \in [0, 1]. \tag{1}$$

Here, the intuition for the first term is as follows. The value is high if the respondent put high confidence into their prediction about the objects on the visualization, and then correctly predicted them. If one of these things did not happen, visualizations should not be considered informative. The other term is more straightforward, simply tracking the coherency of the dataset examples.

My idea was to get a few students from AGISF to take the survey and use their answers, but no one responded to my call, so, going against the practices, I filled out the survey myself. I shuffled the features in the two groups so that I would not know if a given visualization comes from a high- or median-importance feature. I thought to use Q.5 and Q.6 to measure polysemanticity, but when I started evaluating the features, I recognized that there were almost no situations where I could name several objects on the visualizations, and then all of them would appear in the dataset examples. At the same time, dataset examples alone are insufficient to distinguish polysemantic features from those without clearly encoded objects. These issues led me to abandon the attempt at measuring polysemanticity.

3 Results

Finally, I compared the interpretability between the two groups I selected earlier based on importance. The results are presented in Figure 2. The two groups display significant

differences in values of I. The first one has an average of 0.24, while the second one's average is 0.56. In this layer of the network, the preliminary answer to the titular question of the project is *yes*. We can see from the diagrams that most of the low-importance features have $I < 0.5$, indicating not only that the visualizations were hard to read, but also that the dataset examples for them were incoherent, leading to low answers to Q.4. The other group, on the other hand, shows a larger variance, but all of the highest values of I ended up there.

Of course, this analysis is preliminary and it is currently unclear whether it would generalize to larger models and different data modalities (e.g. text). I suspect that features reacting to single clear concepts, such as "brown and white dog muzzle" emerge in our situation because the number of features in the layer (832) is not far from the number of classes that the CNN needs to distinguish (1000). This allows the network to avoid polysemantic features. Another reason for the abundance of these "clean" interpretations of features might lie in the intermediate outputs of InceptionV1. The network was considered deep at the time of its creation, and to help with training, two extra outputs were attached to intermediate layers of the network. This might have enforced meaningful concepts to be represented early on in the layers. In modern deep networks, skip connections are used to solve the convergence issues, but they do not lead to correspondingly higher interpretability.

Analysis on a larger scale would require some automation in the measure of "interpretability". For example, the similarity of the dataset examples and visualizations could be measured using the Frechet Inception Distance [5]. Compute was also clearly a limiting factor in the project, and a larger dataset would make for better examples and importance evaluations.

This project touched on two common approaches to interpretability research. One consists of visualization of the components of a network, the other — of causal interventions to reveal the function of the component. The fact that these two methods are connected in a way described above might indicate that the network's reasoning is more "human-like", and give a little extra hope to the grand project of mechanistic interpretability.

References

[1] S. Anwar, K. Hwang, and W. Sung. Structured pruning of deep convolutional neural networks. *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, 13(3):1–18, 2017.

[2] S. Casper. The Engineer's Interpretability Sequence VI: Critiques of Mechanistic Interpretability Work in AI Safety. "https://www.lesswrong.com/posts/wt7HXaCWzuKQipqz3/eis-vi-critiques-of-mechanistic-interpretability-work-in-ai", 2023.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[4] N. Elhage, T. Hume, C. Olsson, N. Schiefer, T. Henighan, S. Kravec, Z. Hatfield-Dodds, R. Lasenby, D. Drain, C. Chen, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

[5] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

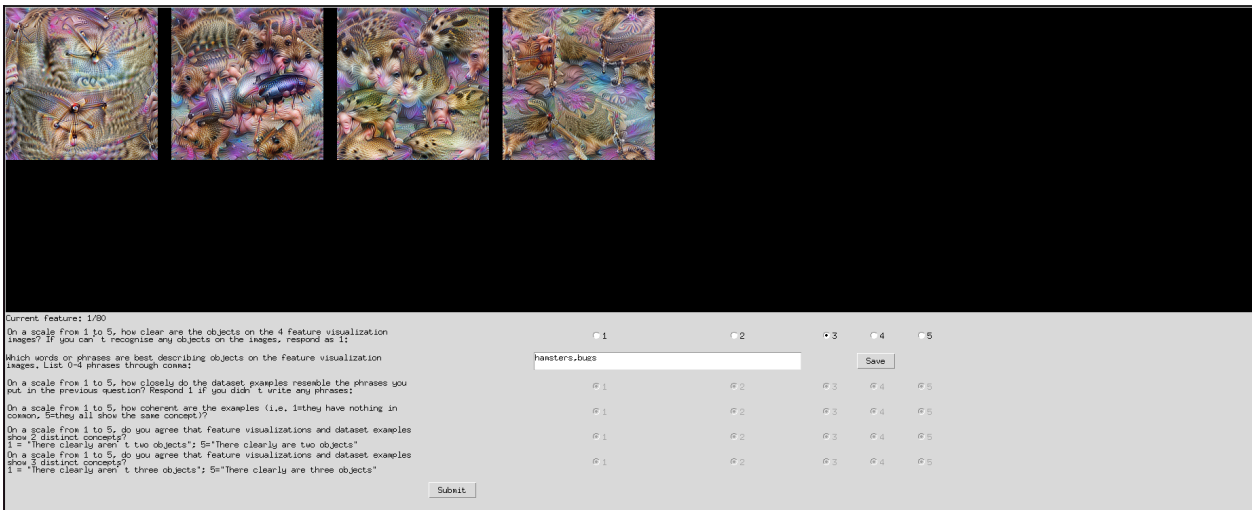
[6] Z. Liu, E. Gan, and M. Tegmark. Seeing is believing: Brain-inspired modular training for mechanistic interpretability. *arXiv preprint arXiv:2305.08746*, 2023.

[7] C. Olah, N. Cammarata, L. Schubert, G. Goh, M. Petrov, and S. Carter. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

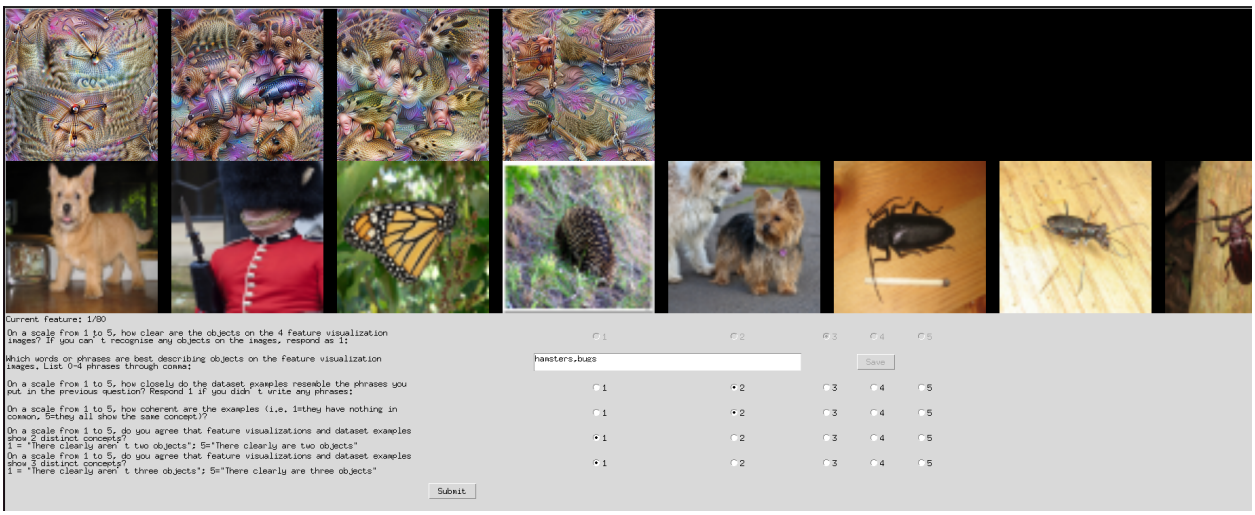
[8] C. Olah, A. Mordvintsev, and L. Schubert. Feature visualization. *Distill*, 2(11), 2017.

[9] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[10] K. Wang, A. Variengien, A. Conmy, B. Shlegeris, and J. Steinhardt. Interpretability in the wild: a circuit for indirect object identification in gpt-2 small. *arXiv preprint arXiv:2211.00593*, 2022.

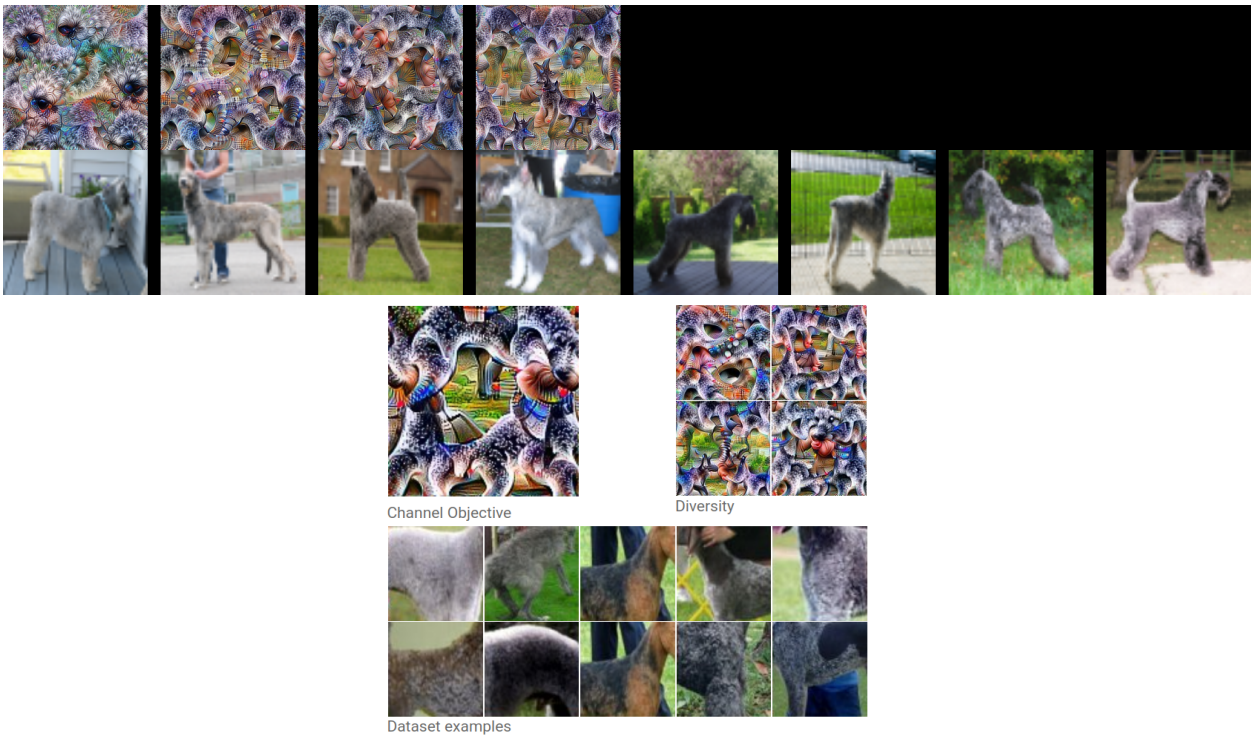


(a) Before submitting the guess about visualizations

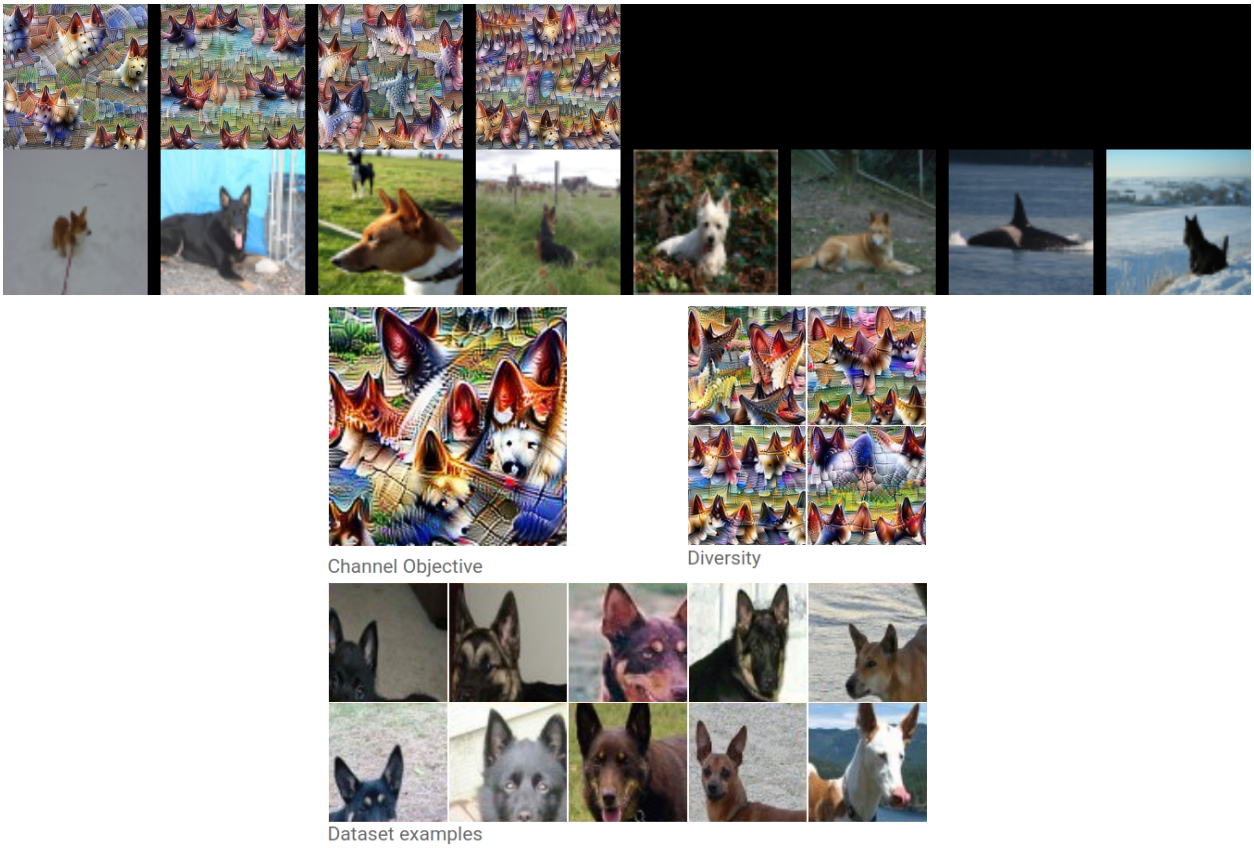


(b) After submitting the guess about visualizations

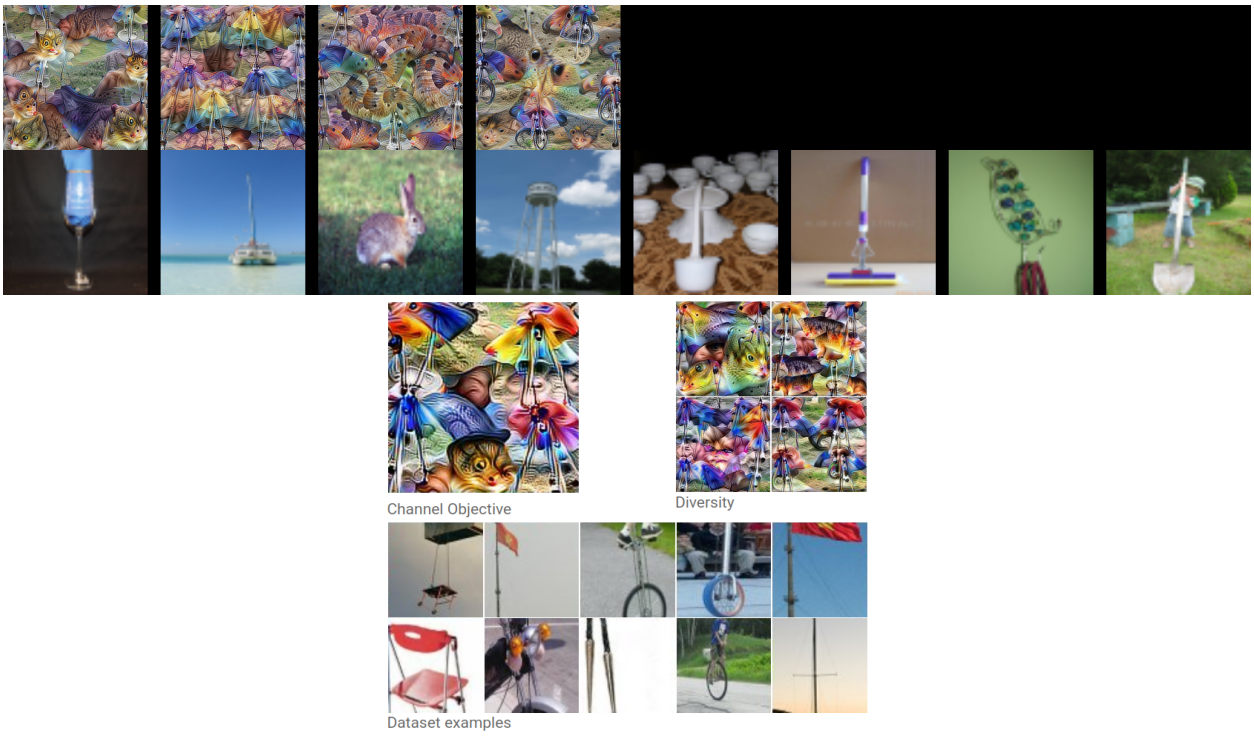
Figure 3: Screenshots of the interface. Questions are unintelligible here, but they correspond to the ones I provide in the text.



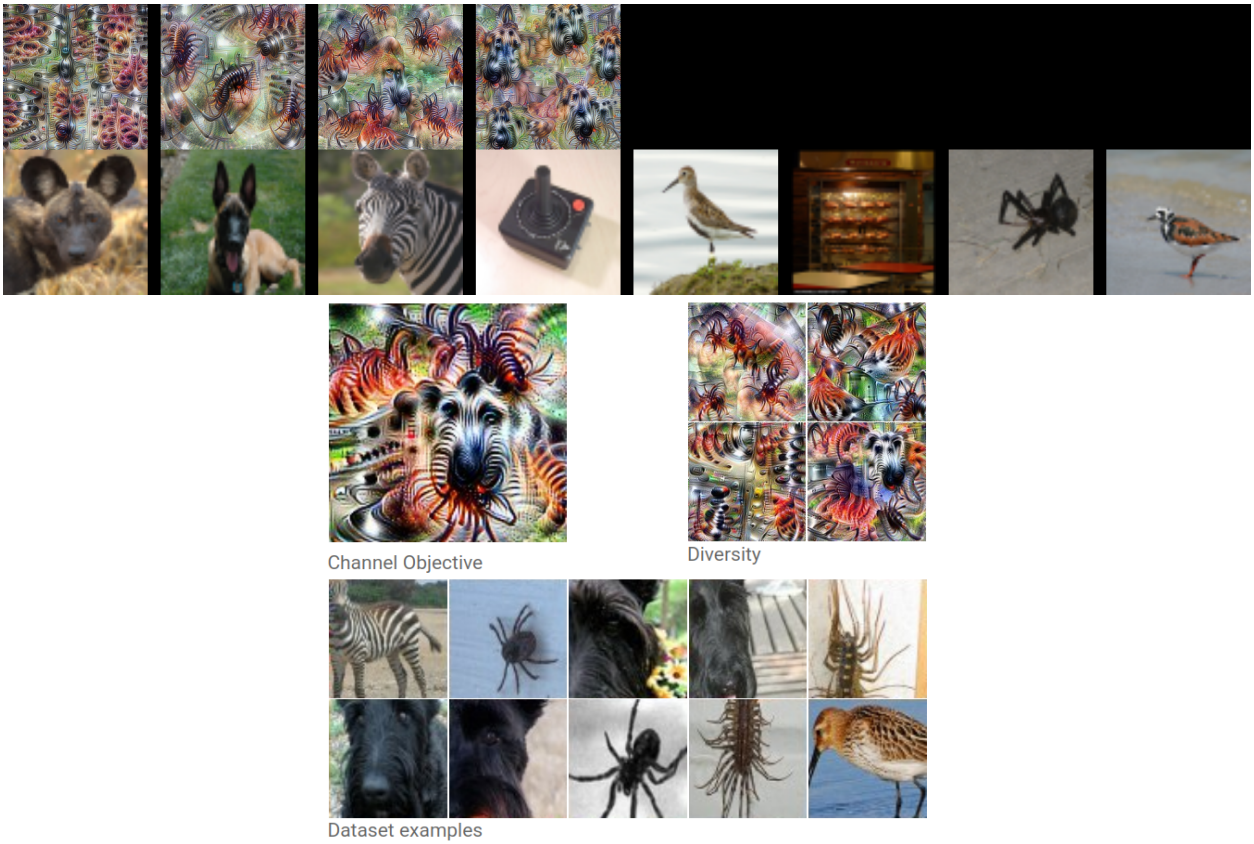
(a) `mixed5a:0`



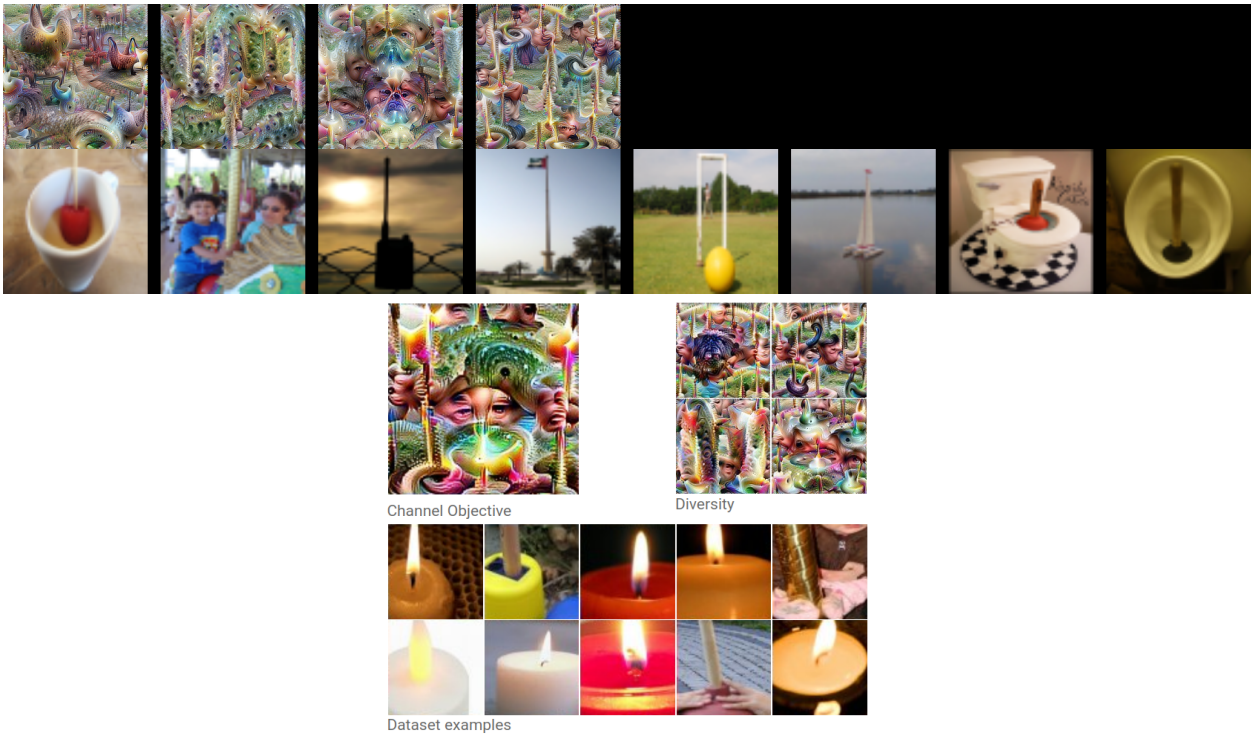
(b) `mixed5a:1`



(c) `mixed5a:2`



(d) `mixed5a:3`



(e) `mixed5a:4`

Figure 4: 5 pairs of visualizations of the first features in `mixed5a`. In each pair, the top set of images are the ones that I generated, and the bottom ones are from [8].