What shapes feature representations? Exploring datasets, architectures, and training

arXiv: https://arxiv.org/pdf/2006.12433.pdf

With many latent features in the training data — such as the colors, textures,and shapes of objects in visual datasets — how does the model separate task-relevant features from irrelevant ones?

Does a model gain sensitivity to diagnostic features by enhancing its representation of them, or by suppressing its representations of other features?

How similar are feature representations across models as a function of training task?

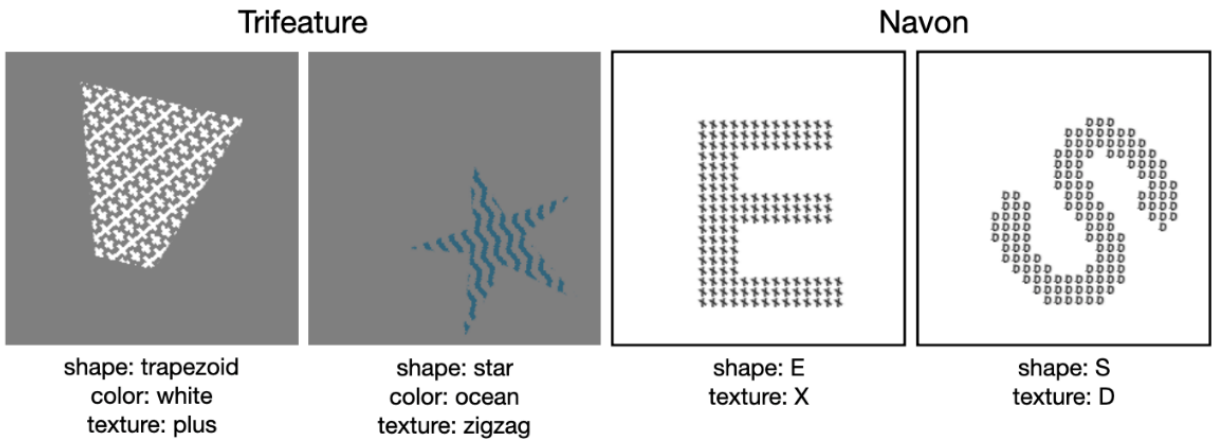
Main takeaways:

- Many input features are already partially represented in the higher layers of untrained models
- Training on a task enhances task-relevant features (increases their decodability relative to an untrained model), and suppresses both task-irrelevant features and some task-relevant features
- When a pair of features redundantly predicts the label, models prefer one of the features over the other, and the preference structure tracks untrained decodability
- In some cases where the models are “lazy”, they suppress a more predictive feature in favor of an easier-to-extract, but less predictive, feature
- Easy features induce representations that are more consistent across model runs than do hard features
- A multi-task model trained to report both easy and hard features produces representations that are very similar to those of a model trained only on the easy feature

Does feature selection happen by enhancement or suppression?

Over training, as a model becomes sensitive to a target feature, does it build this sensitivity by enhancing the target feature, or by suppressing non-target features? How much enhancement and/or suppression occurs?

To measure this, consider two synthetic datasets (shown below) where the model is train to classify images according to their shape, texture, or color. Then we test to see to what extent target and non target features are decodable.



Findings:

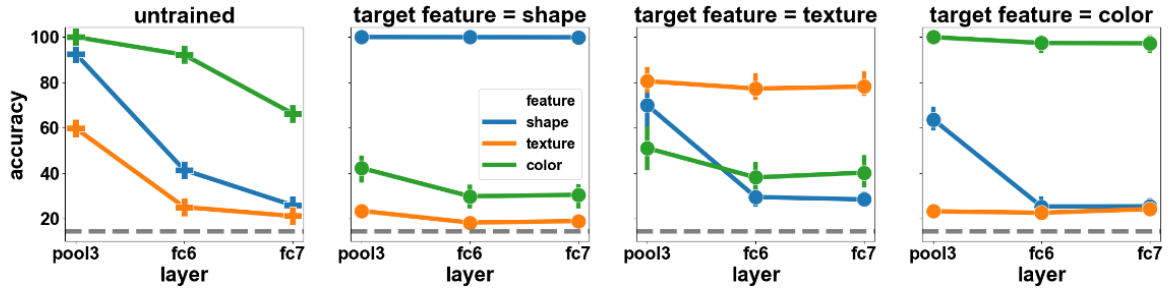


Figure 2: **Target features are enhanced, and non-target features are suppressed, relative to an untrained model in models with an AlexNet architecture trained on Trifeature tasks.** Accuracy decoding features (shape, texture, color) from layers of an untrained model (far left) versus from models trained to classify shape, texture, or color (mean decoding accuracy across models trained on each of 5 cv-splits of the data; error bars indicate 95% CIs). Chance = $\frac{1}{7}$ = 14.3% (dashed gray line). Decoding accuracy is generally higher for target features in the trained than in the untrained model (enhanced) and lower for non-target features (suppressed).

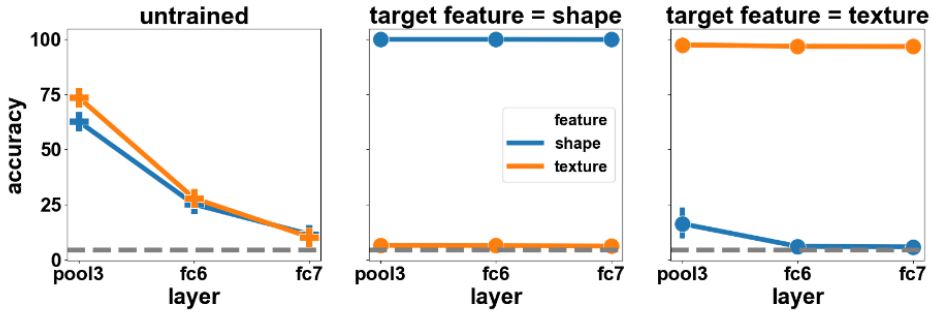


Figure 3: **Feature decodability in models with an AlexNet architecture trained on the Navon dataset.** Accuracy decoding features (shape, texture) from an untrained model (left) versus from shape- (center) and texture-trained models (right). Results corresponding to trained models are mean across models trained on 5 cv splits. Chance = $\frac{1}{23}$ = 4.3% (dashed gray line). As for the Trifeature models (Figure 2), decoding accuracy in the untrained model decreases across layers, and the target features are enhanced, whereas the non-target features are suppressed.

1. **What's already decodable from an untrained model?** Visual features are decodable significantly above chance from the upper layers of untrained models.

   → The decodability of features from an untrained model reflects the model's inductive biases, and might predict the extent to which a feature would be preserved after training the model on a different task

2. **What's decodable after training?** The models are trained to classify a target feature in the presence of one or more non-target, task-irrelevant (uncorrelated with the label) features.

   → Target features were enhanced, resulting in high accuracies across layers.

   → Non-target features were partially suppressed: decoding accuracies were lower than in the untrained model

3. **What if multiple features are predictive?** In real-world objects, features like shape,texture, and color are often correlated, especially for natural objects.

   → When a pair of features perfectly predict the label (color & shape, texture & shape), models preferentially learn one feature. Intriguingly, the less decodable, but still perfectly predictive, feature was sometimes even suppressed relative to an untrained model

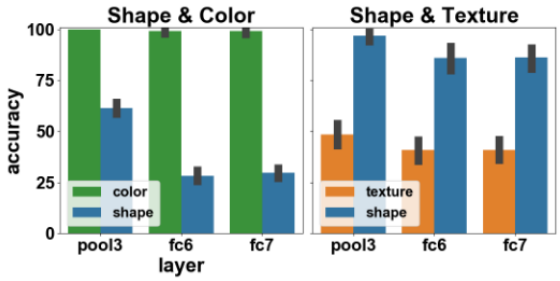


Figure 4: **When two features redundantly predict the label, models preferentially learn one feature.** Color is more decodable than shape, and shape is more decodable than texture, when AlexNet is trained on perfectly predictive pairs.

4. **Do models prefer reliable but difficult features, or easy but less reliable ones?** Does the model prefers a feature that is easier to learn (linearly decodable from the input) but only moderately predictive of the label, or a feature that is more difficult(XOR/parity, not linearly correlated with the input) but more predictive of the label. Will a model trade off predictivity for learnability?

   → The more reliable feature can be suppressed by a less reliable, but easier, one.

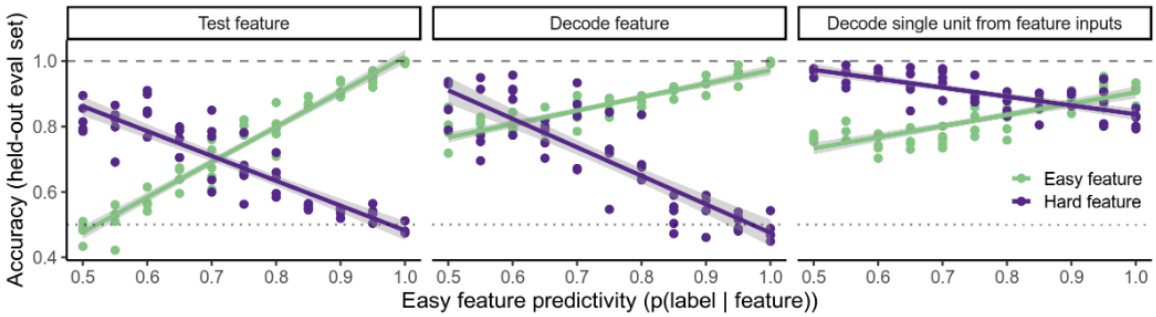


Figure 6: **More reliable, but difficult, features can be suppressed by less reliable, but easier, features.** The difficult feature (purple) had fixed predictivity of 0.9, while the easy feature (green) had varied predictivity (x-axis). The left panel evaluates the model's use of each feature (using a test set with the other feature made unpredictive). The middle panel shows decodability of each feature from the penultimate layer. The right panel shows decodability of a single input unit's value (another linear feature) from the inputs associated with each feature. (5 runs per condition; lines are linear fits.)