Followup: other examples where brain data improves ML benchmarks

1. Li et al. 2019 Use a loss function that balances classification accuracy as well as a match between the DNN and Brain similarity spaces (mouse data). Improves classification for noisy images and robustness to adversarial attacks.
2. Federer et al. 2020 Neural nets that mimic representational similarity matrices perform better on object recognition and were more robust against corruption of training labels. (monkey data). Joint loss term.

Palazzo et al. 2020

DECODING BRAIN REPRESENTATIONS BY MULTIMODAL LEARNING OF NEURAL ACTIVITY AND VISUAL FEATURES

Aims of the work

1. Achieve multimodal learning that projects brain data and image data into the same latent space
2. Using the latent space learned, they will aim to decode brain representations (“mind reading”; e.g., can we tell what image is a person seeing) (not part of the course or exam)
3. Using brain activity to guide machine learning tasks. (visual saliency modeling; our topic)

Relevant prior methods

1. Multimodal learning: learning embeddings from two or more modalities that encode a joint representation of the observation being coded.
    1. Their example: “an image and a piece of text describing the content of that image should be closer in the joint space than an image and an unrelated piece of text”
2. Most prior work in ML tries to learn a joint embedding space for images and text
3. Here: joint embedding of EEG (features) and image (features).

The principle

A Siamese network that uses a version of a triplet loss to maximize the similarity between modalities

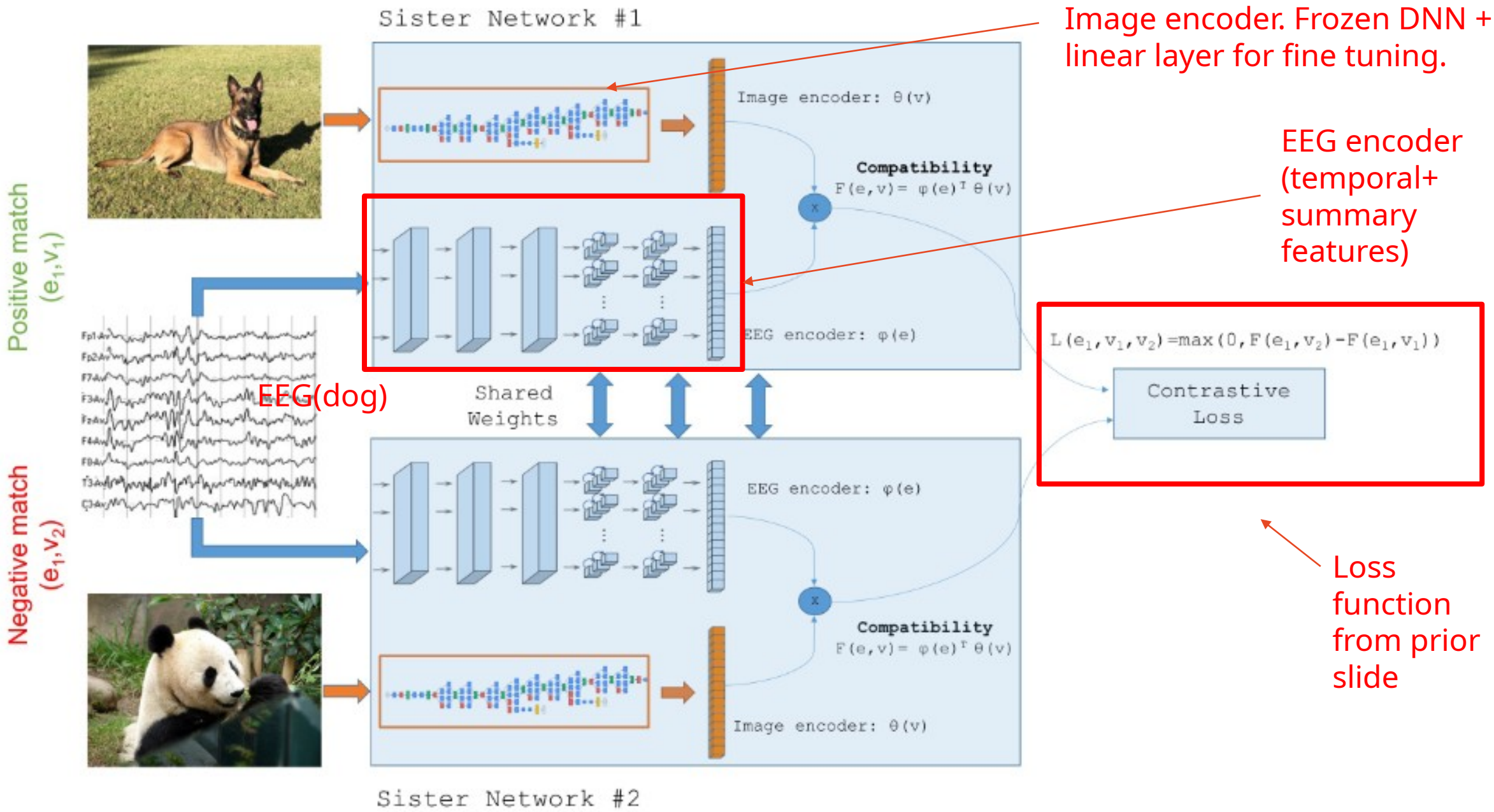
EEG (e) and visual (v) features of the same image (e1, v1) should be mapped to be nearby in image space whereas

EEG features of image1 and visual features of image2 (e1, v2) should be pushed apart.

As compatibility between features, they just take the dot product of the embeddings.

Similarity of mismatching EEG/Vision should be lower than the matching case.

L(e1, v1, v2) = max{0, F(e1, v2) - F(e1, v1)}
F is a similarity measure. The triplet loss is positive if the similarity of the eeg data (e1) with v2 (negative item) is greater than its similarity with the positive item (v1)


Sister Network #1
Image encoder. Frozen DNN + 1 linear layer for fine tuning.
Image encoder: θ(v)
Compatibility
F(e,v)= φ(e)ᵀ θ(v)
EEG encoder (temporal+ summary features)
EEG encoder: φ(e)
Positive match
(e1,v1)
EEG(dog)
Shared Weights
L(e1,v1,v2)=max(0,F(e1,v2)-F(e1,v1))
Contrastive Loss
EEG encoder: φ(e)
Negative match
(e1,v2)
Compatibility
F(e,v)= φ(e)ᵀ θ(v)
Loss function from prior slide
Image encoder: θ(v)
Sister Network #2

Details of encoders

The EEG data are pushed through four 1Dconv layers (learning filters at different temporal scales), and these are fed into recurrent layer. The hidden state of the recurrent layer is fed to fully connected layer from which embeddings are extracted for compatibility function.

For image encoding: they train different architectures, whose representations are fine tuned during the Siamese training.

Dataset

1. EEG data: 6 people watch 40 categories from ImageNet; 50 images in each class.
2. HighRes EEG: 128 channels; 1KHz recording; 0.5sec each image.
3. EEG dataset split into 80% train; 10% validation; 10% test

Classification challenge (I)

After training, they use the trained EEG and image encoders as feature extractors in the joint embedding space,

- Followed by Softmax layer for image classification, where they classify into 40 classes

They test different configurations of Image and EEG encoding in joint embedding

- They then quantify each encoder separately as a feature extractor for classification. Results :

Image encoder	EEG	EEG Accuracy	Image Accuracy	Average Accuracy
Inception-v3	LSTM	90.1%	93.6%	91.9
Inception-v3	**GRU**	**90.4%**	**94.7%**	**93.0**
ResNet-101	LSTM	90.7%	91.2%	91.0
ResNet-101	GRU	92.3%	91.5%	91.9
DenseNet-161	LSTM	92.4%	92.3%	92.4
DenseNet-161	GRU	93.7%	91.8%	92.8
AlexNet	LSTM	85.6%	70.1%	77.8
AlexNet	GRU	77.2%	69.9%	73.6

Model	Visual Feature Learning	Joint Learning with EEG
AlexNet	65.5 %	70.1 %
Inception-v3	93.1 %	94.7 %
ResNet-101	90.3 %	91.5 %
DenseNet-161	91.4 %	92.3 %

Classification challenge (II)

Joint embedding increases performance.

Saliency detection

What is saliency detection? A set of algorithms that can process an image to identify which parts of the image are particularly important.

They are evaluated by comparing the saliency map generated by the algorithm to the ground truth

- As simple as a correlation between the two maps;
- Normalized Scanpath Saliency: computes the mean normalized saliency value at fixated locations (more is better)

SALICON (Huang et al., 2015): A DNN trained to produce saliency maps. It was created by training an off-the-shelf DNN to predict which parts of image space were salient. Specifically, a layer is added on top of the last CONV layer. Contains a single feature, which learns which combinations of feature maps (at that depth) predict pixel saliency (Kernel = 1x1). The network is trained with objective functions that maximize fit to a human saliency map. SALicon can predicts saliency at many different image scales by combining information from the original image and downsampled versions of that the original.

Using the joint embedding for saliency detection

After training for joint embedding, they employ a masking process

1. A mask is applied to a part of the image
2. The compatibility between the brain vector (to original image) and the image vector (of the masked image) is computed
3. The decrease in match vs. the original compatibility score is computed.

"The saliency value [snip] at pixel (x, y) [snip] is obtained by removing the σ × σ image region around (x, y) and computing the difference between the original compatibility score and the one after suppressing that patch"

- Note in the paper they mask around each pixel at different scale; irrelevant here see their Fig. 3.

Validation and comparison of saliency maps from joint embedding-perturbation

1. Participants freely observe the same 2000 images for which EEG data were collected.
2. Competitive saliency-detection algorithms: Salicon; SALNET
3. Additional baseline: effect of masking on simple visual classification.

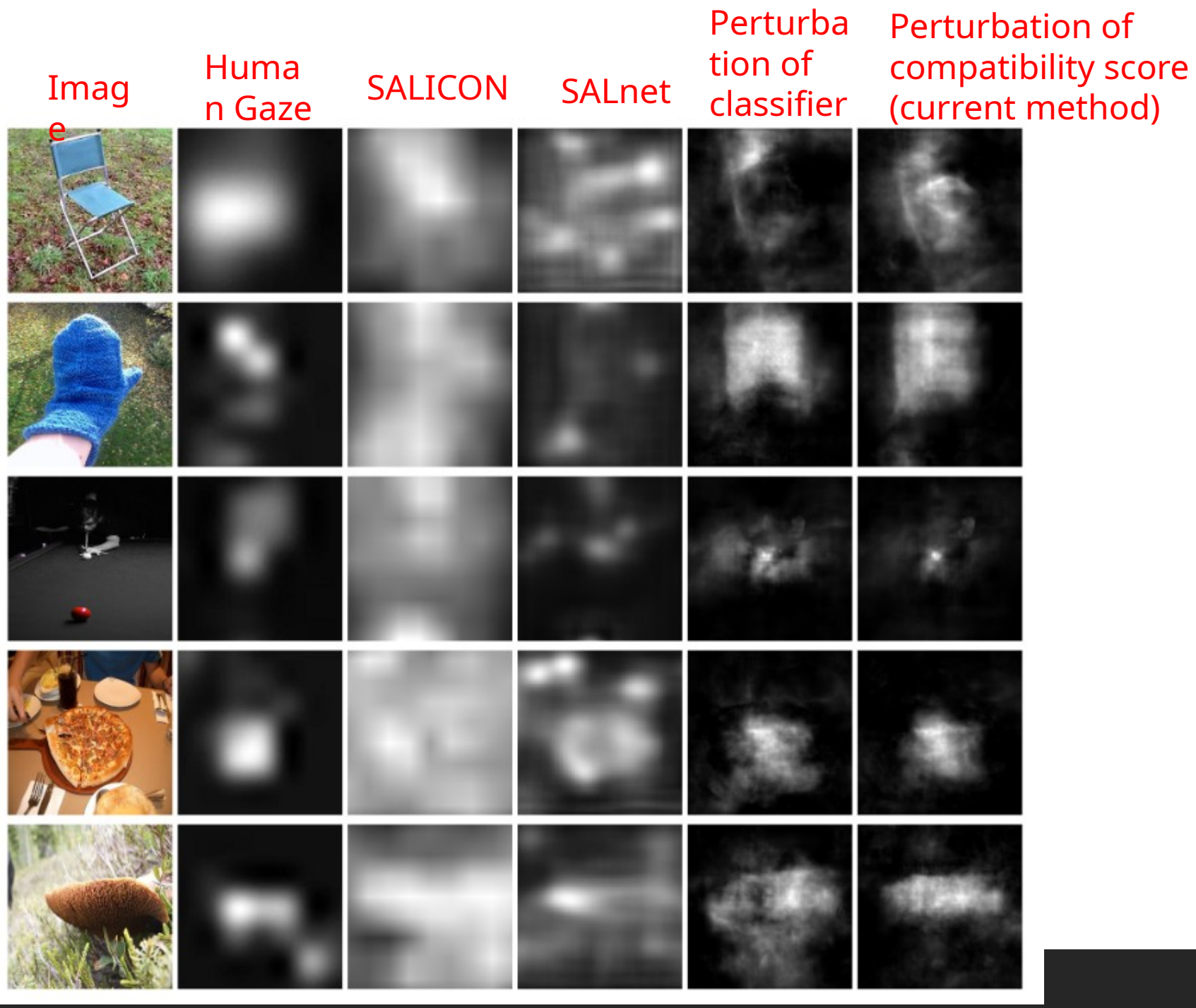


Image
Human Gaze
SALICON
SALnet
Perturbation of classifier
Perturbation of compatibility score (current method)

Method	s-AUC	NSS	CC
SalNet	0.637	0.618	0.271
SALICON	0.678	0.728	0.348
Visual classifier–driven detector	0.580	0.505	0.201
Our neural-driven detector	**0.692**	**1.061**	**0.378**
Human Baseline	0.939	3.042	1

Results: match to human saliency maps

Accommodating human uncertainty

TO MAKE CLASSIFICATION MORE ROBUST.
PETERSON, J. C., BATTLEDAY, R. M., GRIFFITHS, T. L., & RUSSAKOVSKY, O. (2019). HUMAN UNCERTAINTY MAKES CLASSIFICATION MORE ROBUST. *ARXIV:1908.07086 [CS]*. HTTP://ARXIV.ORG/ABS/1908.07086

Main idea

1. Introduce a soft-labelling scheme that is informed by human uncertainty
2. Evaluate whether soft labels makes categorization more generalizable to out of sample data

What are soft labels?

In a classification-learning context, soft labels are ‘ground truth’ labels for an observation where it is not the case that the entire mass of the observation is located at a single correct category.

Many soft labeling schemes have been described:

1. split mass uniformly among non-target item

2. split mass as a function of nearness of an observation to a classification boundary (prevents overfitting)

3. split mass in relation to types of objects potentially recognized in the scene (categorization and object detection)

Current insight

Human knowledge is inconsistent with the notion of ‘hard labels’ because in some cases humans are unconfident in category assignment whereas CNNs are

In some cases humans may be unconfident, and so will the CNN, but in different ways.

They come up with a method to estimate the soft-label probability distribution from human judgments.

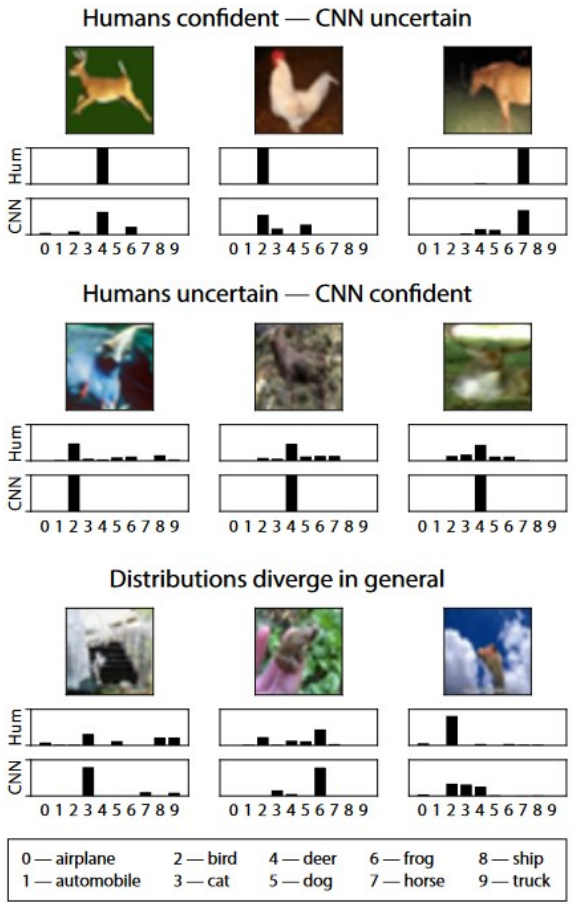


Figure 1: CIFAR10 images for which humans and our best traditionally-trained CNN (Shake-Shake [11]) agree in their top guess, but systematically differ over other choices.

Core method

"If we expect the human image label distribution $p_{hum}(y|x)$ to better reflect the natural distribution over categories given an image, we can use it as an improved estimator for $p(y|x)$."

- Where y is the distribution of activity assigned to all labels for a given image x

They then simply use the usual cross-entropy loss to minimize the divergence between the human distribution and post-softmax activity distribution

Dataset

They produce a curated dataset, CIFAR10H: a behavioral dataset consists of ~500K human categorization decisions over the 10,000-image testing subset of CIFAR10; approx. 50 judgments per image.

One of the reason for using: CIFAR10 contains observations that are close to the category boundaries

They use the Mechanical Turk: on each trial, a person categorizes each image by "clicking one of the 10 labels surrounding it as quickly and accurately as possible (but with no time limit)"

Test

They train multiple architectures using the human labeling data

- 9K images for training, 1K images for testing. This is simply to show accuracy for the homogenous dataset.
- They then apply the learned Model (Weights) to several other CIFAR10 variants.

As generalization measures they evaluate

- Accuracy.
- Cross-entropy loss

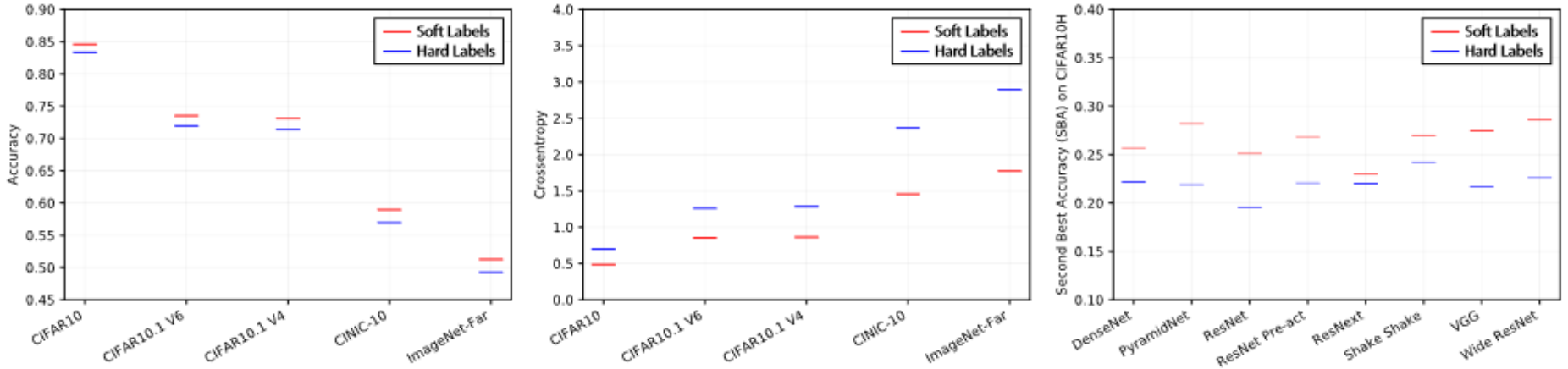


Figure 2: Generalization results. Left: accuracy against ground-truth labels, for increasingly out-of-training-sample distributions, averaged across CNNs. Accuracy was higher using human labels for every individual CNN and dataset. Center: crossentropy against ground-truth labels, averaged across CNNs. Loss was lower using human labels for every individual CNN and dataset. Right: Second best accuracy (SBA) for all models using `CIFAR10H` held out set, averaged across folds.

Generalization improves with human soft labels

Alternatives to training on image level soft labels

1. They approximate category-level confusions by averaging ratings across images within each category. They apply then the same soft label for all images in a category “category soft targets” in Table 1
2. As an alternative way for simulating confusion, they take the post-softmax profile of each image from 8 different classification models (“knowledge distillation”)
3. They use mixup that is a virtual training mechanism that generates merged images (“virtual training examples”) and assigns them merged labels.
4. They also use human uncertainty in different way: Based on the human confusion, assign all mass to a single category, but during training ‘swap’ the labels: for single image train on more than single 1-hot model. “sampled hard targets” in Table 1

Results of evaluation against soft-label alternatives

Probabilistic soft labels outperform their alternative methods for soft-label construction and also generalize better to new test sets.

Protection against adversarial attacks

The adversarial attack aims to produce a wrong label for a minimally changed image.

Accurate behavior here is maintenance of the original label.

Human training produces a network that is much more robust against Fast Gradient Sign Method (FGSM) attacks; one of the easiest ways to create an adversarial image.

Also, when exposed to an Adv. Example, the Cross Entropy between the network and model is lower after being trained with Human-trained soft labels. This means that A.E is not shifting the decision as much as it does on a non-soft-label architecture.

Potential weaknesses

There are multiple ways of creating soft labels, or preventing overfitting for stimuli that don't match a specific annotation

1. The cross entropy loss term for training instance X might be down-weighted if the label for X is determined as not trustworthy or ambiguous. This does not require setting up a soft label for training; simply determining which observations are less important than others.
2. Li et al. 2020. To prevent overfitting, down-weight images that are near decision boundary of two classes
3. In the area of NLP there is a substantial literature on how to incorporate annotator disagreement into models. Review in Fornaciari et al 2021. They find that incorporating soft-label information based on annotator disagreement improves performance on NLP categorization tasks. They add prediction of soft labels as an auxiliary task in Multi-Task Learning.