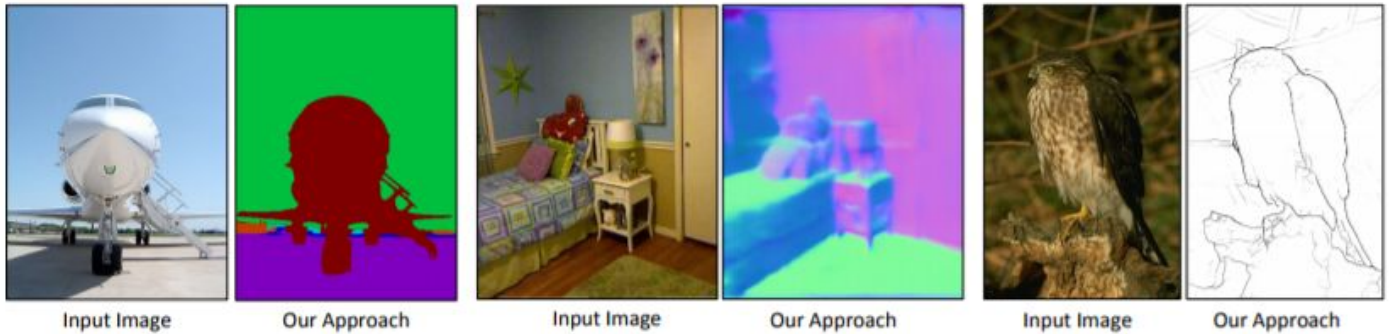
PixelNet: Representation of the pixels, by the pixels, and for the pixels

A Paper Review by Kristen Bystrom

Three Common Pixel Prediction Problems



(a) Semantic Segmentation

- High-level
- Puts pixels into k categories for each of the hypercolumn features
- Commonly benchmarked against PASCAL-context dataset

(b) Surface Normal Estimation

- Mid-level
- Predicts a real value output for the values of the hypercolumn of each pixel
- Commonly benchmarked against NYUDv2 depth dataset

(c) Edge Detection

- Low-level
- Predicts a binary output for the values of the hypercolumn of each pixel
- Commonly benchmarked against BSDS

What is PixelNet?

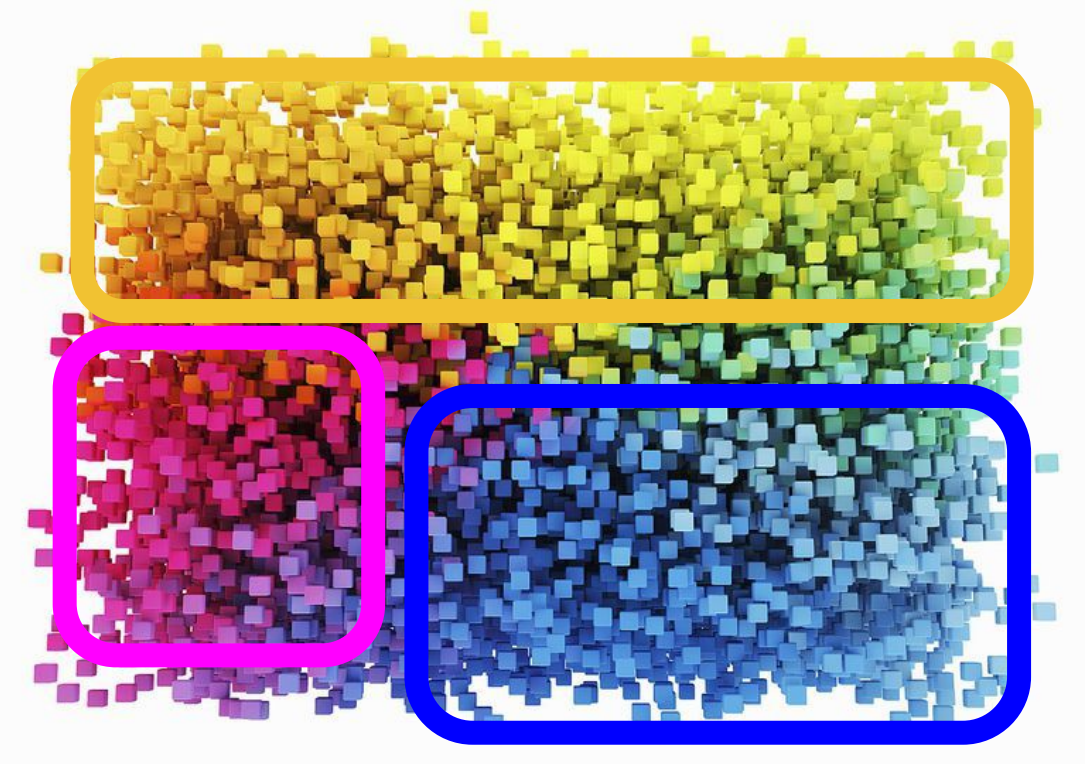
PixelNet is a single architecture that solves these pixel prediction problems VERY QUICKLY

It does this by implementing stratified sampling of the pixels which helps to:

1. add diversity during batch updates, speeding up learning;
1. explore complex nonlinear predictors, improving accuracy;
2. efficiently train state-of-the-art models *tabula rasa* (i.e., *from scratch*) for diverse pixel-labeling tasks.

Github here: https://github.com/aayushbansal/PixelNet

Stratified sampling for pixels



Let’s say we want to sample 4% of the pixels in this image.

We could take a random 4% sample and risk getting a sample of ONLY YELLOW pixels.

Or we could check the color of each pixel in advance, and intentionally sample:

10 Yellow Pixels
5 Blue Pixels
2 Pink Pixels

Which would ensure diversity

Method	IoU (V)	Mean	Median	RMSE	11.25°	22.5°	30°
All Pixels	44.4	25.6	19.9	32.5	29.1	54.9	66.8
Random 4% Pixels	44.6	25.7	20.1	32.5	28.9	54.7	66.7

Table 1. **Sampling:** We demonstrate that sampling few pixels for each mini-batch yields similar accuracy as using all pixels. The results are computed on models trained for 10 epochs and 10,000 iterations for semantic segmentation and surface normal estimation, respectively.

Method	IoU_1 (V)	IoU_2 (V)	Mean	Median	RMSE	11.25°	22.5°	30°
$1 \times 40,000$	7.9	15.5	24.8	19.5	31.6	29.7	56.1	68.5
$5 \times 2,000$	**38.4**	**47.9**	**23.4**	**17.2**	**30.5**	**33.9**	**60.6**	**71.8**

Table 4. **Statistical Diversity Matters:** For a given computational budget, using diverse set of pixels from more images shows better performance over more pixels from a few images. IoU_1 and IoU_2 show performance for 10K and 20K iterations of SGD for semantic segmentation. For surface normal estimation, we show performance for 10K iterations of SGD. This suggest that sampling leads to faster convergence.

Implementation with sparse predictions

1. Perform a forward pass to compute dense convolutional responses at all layers
2. For each sampled pixel, where p is an element of P, compute it's hypercolumn feature on demand as follows:
    a. For each pixel layer i, compute the 4 discrete locations in the feature map closest to the p
    b. Compute the values of the hypercolumn for that pixel via bilinear interpolation
3. Rearrange the sparse of hypercolumn features into a matrix for downstream processing

Discussion Questions

1. What are your thoughts on the statistical accuracy VS computational efficiency tradeoff?
2. How does stratified sampling help to reduce the sample size? Can you think of other methods that might also address this problem?
3. What are the differences between the three pixel prediction problems addressed in the paper (edge detection, surface normal estimation, semantic segmentation)? Has anyone used any of these before?
4. This paper uses some common datasets that are popularly used to benchmark models for edge detection, surface normal estimation, and semantic segmentation. Why is benchmarking new models on common data sets important? Are there drawbacks to always using the same datasets to test new models.