LCNN: Lookup-based Convolutional Neural Network
–Supplementary Material–

1. Layer-wise speedup

In this section we compare the layer-wise speedup of LCNN with the baselines. In AlexNet most of the computation is done in the early layers, where the input size is still large. Table 1 shows the percentage of the computation in each layer of AlexNet and the speedup gain of each model on each layer. XNOR-Net [1] gets 32× speedup on 32-bit machines, and it can be higher for 64-bit or 128-bit machines. However, since they don't binarize the first layer, where 9.29% of computation is done, their speedup is bounded by $\frac{1}{9.29\%} = 10.8\times$. This is still much lower than LCNN-fast speedup, which gets about the same accuracy. Wen *et al.* [2] gets good speedup on conv2-5, yet their speedup is much lower on the first layer. We think this is because they're sparsifying the convolution tensors. The convolution tensor in the first layer cannot become very sparse as they are performing on the input itself, which has only 3 channels. LCNN-accurate, however, is speeding up the first layer by representing the convolution tensor by a sparse combination of a set of vectors. This allows a more compact representation, and therefore larger speedup in that layer.

2. Few-example trials

We do the few-example experiment under two settings: 1) Try 5 random samplings of 10 random categories for few-example training and report the average over all. 2) Set aside all cats (7 categories), bicycles (2 categories) and sofa (1 category). For the latter setting, the following categories are excluded:

1-	`n02123045`	tabby.n.01
2-	`n02123159`	tiger_cat.n.02
3-	`n02123394`	persian_cat.n.01
4-	`n02123597`	siamese_cat.n.01
5-	`n02124075`	egyptian_cat.n.01
6-	`n02125311`	cougar.n.01
7-	`n02127052`	lynx.n.02
8-	`n02835271`	bicycle-built-for-two.n.01
9-	`n03792782`	mountain_bike.n.01
10-	`n04344873`	studio_couch.n.01

The first 7 categories are cats, categories 8 and 9 are bicycles, and category 10 is a sofa.

In each of the trials, we repeat the random sampling of the few examples (1, 2 or 4 examples) 20 times. We evaluate the performance of LCNN and CNN on each random sampling and get the average over all. Figure 1 shows the categories that have been excluded and the performance of LCNN and the CNN baseline in each trial. Notably, LCNN is consistently getting higher accuracy in all trials.

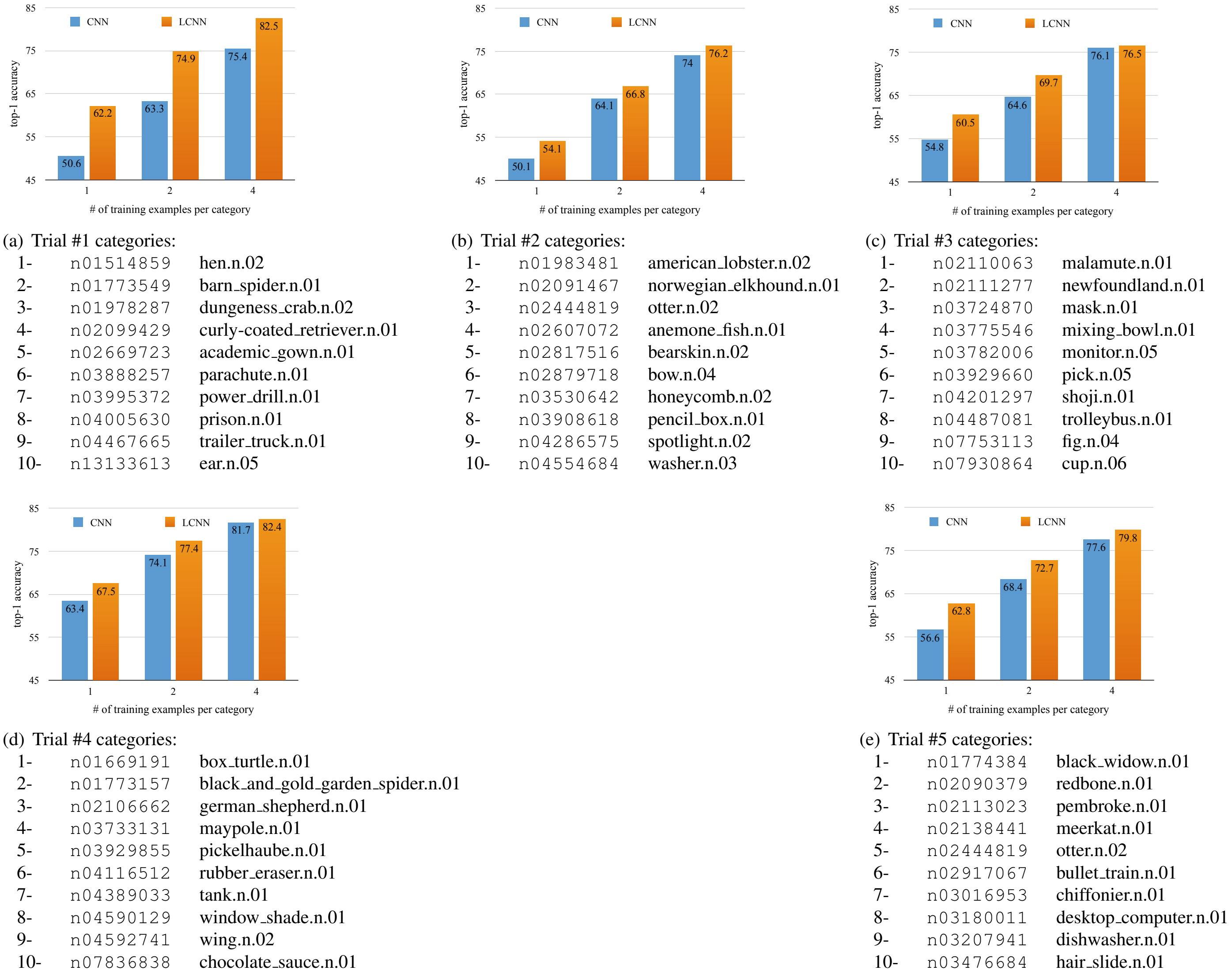
References

[1] M. Rastegari, V. Ordonez, J. Redmon, and A. Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, 2016. 1

[2] W. Wen, C. Wu, Y. Wang, Y. Chen, and H. Li. Learning structured sparsity in deep neural networks. In *NIPS*, 2016. 1

AlexNet	conv1	conv2	conv3	conv4	conv5	fc6	fc7	fc8	overall
computation %	9.29%	39.45%	13.17%	19.76%	13.17%	3.33%	1.48%	0.36%	100%
Wen *et al.* [2]	1.05×	3.37×	6.27×	9.73×	4.93×	1×	1×	1×	3.1×
XNOR-Net [1]	1×	32×	32×	32×	32×	32×	32×	1×	8.0×
LCNN-fast	16.66×	80.24×	83.23×	75.47×	61.99×	7.73×	7.91×	1×	37.6×
LCNN-accurate	6.97×	2.57×	3.51×	3.75×	3.21×	3.14×	3.83×	1×	3.2×

Table 1. Comparing the layer-wise speedup of each model on AlexNet. The accuracy of each model is reported in the paper.



(a) Trial #1 categories:

1-	`n01514859`	hen.n.02
2-	`n01773549`	barn_spider.n.01
3-	`n01978287`	dungeness_crab.n.02
4-	`n02099429`	curly-coated_retriever.n.01
5-	`n02669723`	academic_gown.n.01
6-	`n03888257`	parachute.n.01
7-	`n03995372`	power_drill.n.01
8-	`n04005630`	prison.n.01
9-	`n04467665`	trailer_truck.n.01
10-	`n13133613`	ear.n.05

(b) Trial #2 categories:

1-	`n01983481`	american_lobster.n.02
2-	`n02091467`	norwegian_elkhound.n.01
3-	`n02444819`	otter.n.02
4-	`n02607072`	anemone_fish.n.01
5-	`n02817516`	bearskin.n.02
6-	`n02879718`	bow.n.04
7-	`n03530642`	honeycomb.n.02
8-	`n03908618`	pencil_box.n.01
9-	`n04286575`	spotlight.n.02
10-	`n04554684`	washer.n.03

(c) Trial #3 categories:

1-	`n02110063`	malamute.n.01
2-	`n02111277`	newfoundland.n.01
3-	`n03724870`	mask.n.01
4-	`n03775546`	mixing_bowl.n.01
5-	`n03782006`	monitor.n.05
6-	`n03929660`	pick.n.05
7-	`n04201297`	shoji.n.01
8-	`n04487081`	trolleybus.n.01
9-	`n07753113`	fig.n.04
10-	`n07930864`	cup.n.06

(d) Trial #4 categories:

1-	`n01669191`	box_turtle.n.01
2-	`n01773157`	black_and_gold_garden_spider.n.01
3-	`n02106662`	german_shepherd.n.01
4-	`n03733131`	maypole.n.01
5-	`n03929855`	pickelhaube.n.01
6-	`n04116512`	rubber_eraser.n.01
7-	`n04389033`	tank.n.01
8-	`n04590129`	window_shade.n.01
9-	`n04592741`	wing.n.02
10-	`n07836838`	chocolate_sauce.n.01

(e) Trial #5 categories:

1-	`n01774384`	black_widow.n.01
2-	`n02090379`	redbone.n.01
3-	`n02113023`	pembroke.n.01
4-	`n02138441`	meerkat.n.01
5-	`n02444819`	otter.n.02
6-	`n02917067`	bullet_train.n.01
7-	`n03016953`	chiffonier.n.01
8-	`n03180011`	desktop_computer.n.01
9-	`n03207941`	dishwasher.n.01
10-	`n03476684`	hair_slide.n.01

Figure 1. Comparing LCNN and standard CNN on few-example training. LCNN beats standard CNN in all samplings.