SynDaCaTE: A Synthetic Dataset For Evaluating Part-Whole Hierarchical Inference

Jake Levi, Mark van der Wilk
Department of Computer Science, University of Oxford
`jake.levi@stcatz.ox.ac.uk`

Engineering and Physical Sciences Research Council

EPSRC Centre for Doctoral Training in Autonomous Intelligent Machines & Systems

University of Oxford — Department of Computer Science

Introduction

- ▶ Better **inductive biases** have better data-efficiency and may address extreme data-inefficiency of modern AI
- ▶ The human visual system uses **part-whole hierarchies** as an inductive bias, and is extremely efficient and robust
- ⇒ Reasonable to expect that part-whole hierarchies = useful inductive bias in computer vision
- ! But this has been tried before in computer vision ("capsule networks"), surely it doesn't work !?!?
- ▶ Well...

Methods: framework for part-whole hierarchical inference

- ! What does "infer part-whole hierarchy" even mean!?
- ▶ Each object (part or whole) has discrete class and continuous pose
- def "Generalised pose" = `concat(one_hot(class), pose)`
- def "Infer object" = infer its generalised pose
- def "Infer part-whole hierarchy" = infer all objects (wholes and parts) from an image
- ▶ **Two necessary sub-tasks** to infer part-whole hierarchy:
  1. **Image-to-parts**: infer {set of parts} from an image
  2. **Parts-to-wholes**: infer {set of wholes} from {set of parts}

Methods: SynDaCaTE dataset

- ▶ Capsule networks *claim* to infer part-whole hierarchies
- ! But if only trained end-to-end on ImageNet, how do you know?!
- ! To evaluate image-to-parts and parts-to-wholes inference, we need full ground-truth part labels = unavailable in existing datasets
- ⇒ We introduce a SYNthetic DAtaset for CApsule Testing and Evaluation (SynDaCaTE) which:
  1. Has **ground-truth part information built in**, which is optionally available for training
  2. Can use images or [class/generalised pose] of [single/multiple] [lines/characters/words] as [input/target]
- ✓ Can **rigorously evaluate** models trained on necessary subtasks in isolation and compare with end-to-end performance

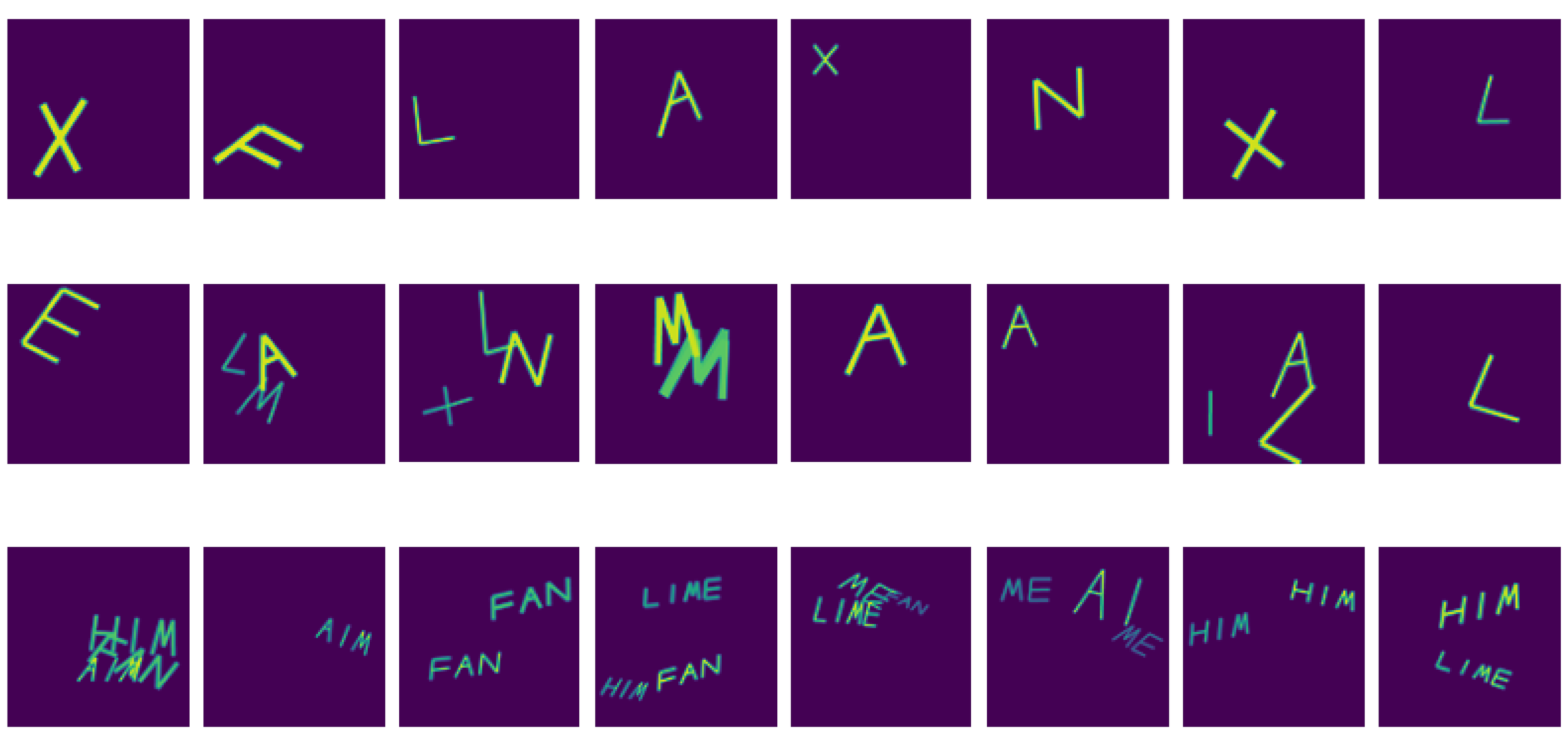

Figure 1: Example images from three different SynDaCaTE tasks. **Top row**: ImToClass. **Middle row**: ImToChars. **Bottom row**: Words.

Experiments: data-efficiency from images

- ! Even MLP can reach perfect test accuracy with enough data
- ⇒ To compare inductive biases, we evaluate **data-efficiency**
- ▶ CNN (red) vs CapsNet (orange) shown in Figure 2
- ▶ **Conclusion:** CNN much more data-efficient than CapsNet!

Experiments: data-efficiency from pre-trained parts

- ? Is CapsNet failing on image-to-parts/parts-to-wholes/both? →
  1. Train CNN on ImToParts to predict sets of parts from images
  2. Use ImToParts-CNN final hidden layer activations as inputs in PreTrainedPartsToClass task (target = class)
- ▶ With pre-trained part representations, CNN and CapsNet (light green/dark green in Figure 2) perform (1) much better vs from images, (2) very similar to each other
- ⇒ **Conclusions:**
  1. Bottleneck in CapsNet = image-to-parts
  2. CapsNet no more efficient at parts-to-wholes than CNN
  3. Parts = useful representation for data-efficient classification

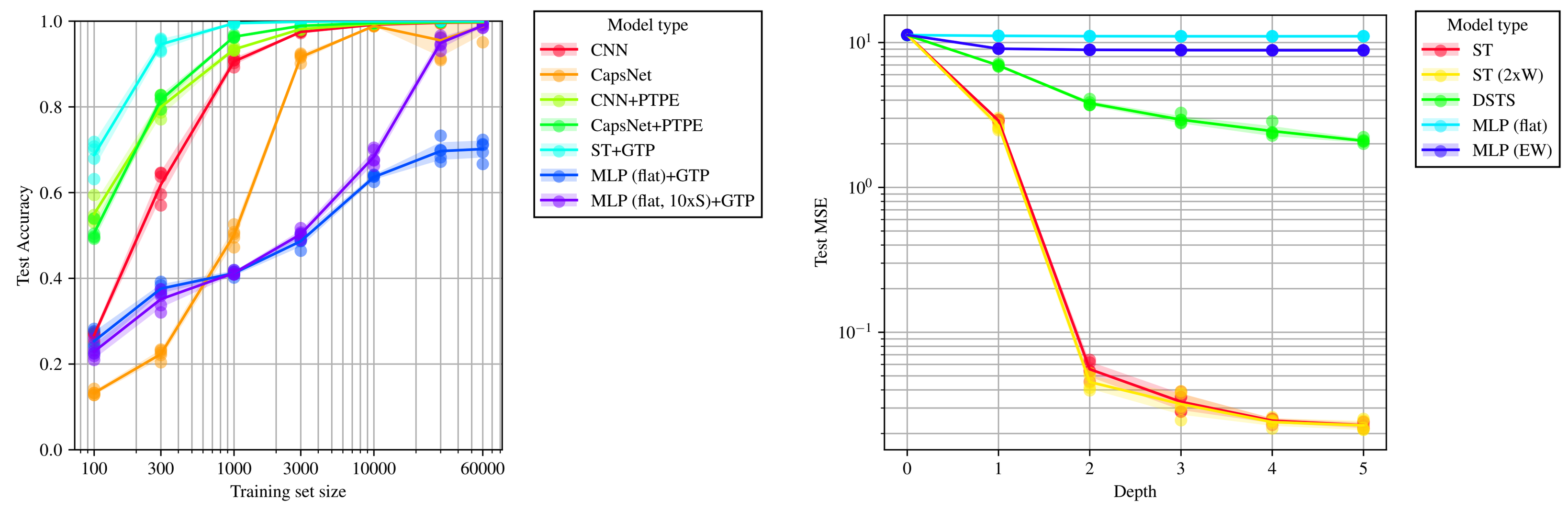

Figure 2: **Left**: data-efficiency of models trained on single-object classifcation tasks. **Right**: MSE of models as a function of depth, trained to predict sets of wholes from sets of parts (PartsToChars).

Experiments: PartsToChars vs depth

- ? What is the *best* model for parts-to-wholes inference?
- ▶ Train models to predict multiple objects from randomly-ordered sets of parts, results in Figure 2 (right)
- ⇒ **Conclusion:** SetTransformer with depth ≥ 2 is better than other models at parts-to-wholes by *more than an order of magnitude*
- ? Is depth ≥ 2 necessary or simply more parameters?
- ▶ SetTransformer with $2\times$ width has $\approx$ same performance profile
- ⇒ **Conclusion:** depth ≥ 2 *is* necessary (CF "induction heads")

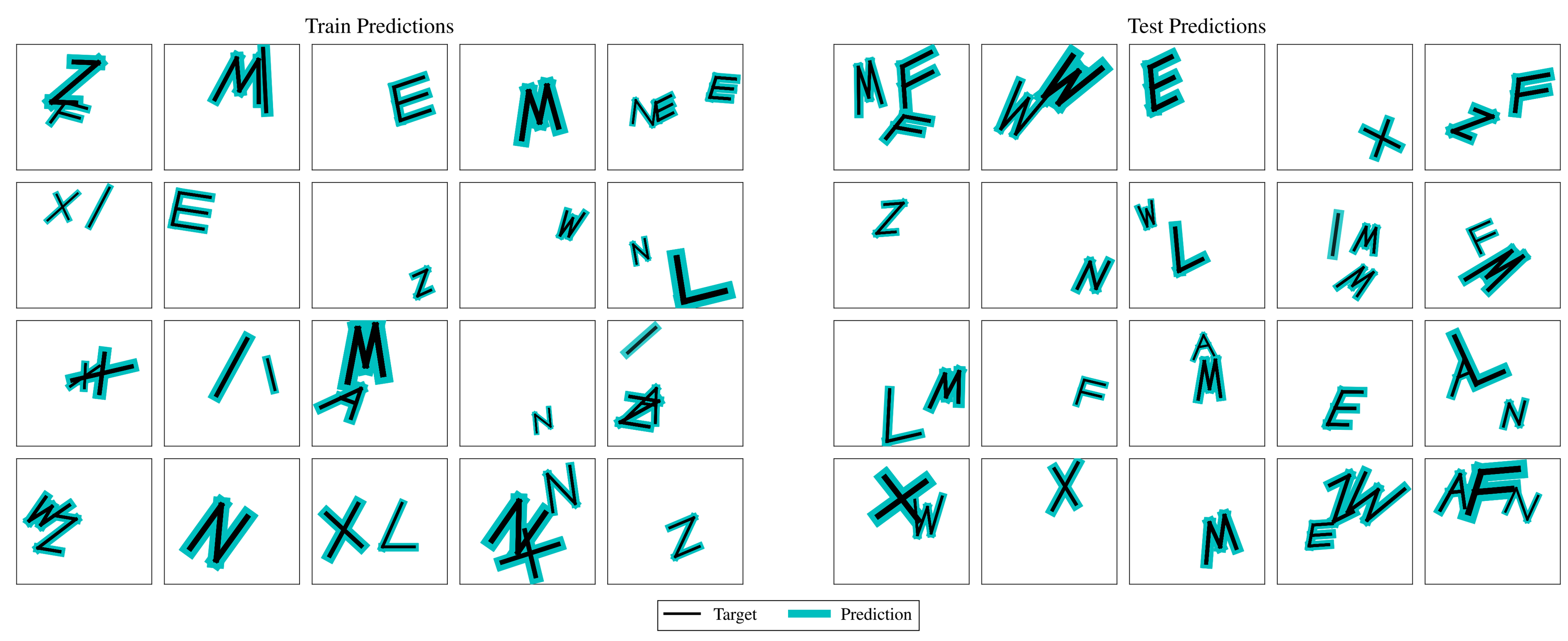

Figure 3: Predictions of best SetTransformer trained on PartsToChars.

Experiments: data-efficiency from ground-truth parts

- ✓ SetTransformer from ground-truth parts is more data-efficient than [CNN/CapsNet] from [pre-trained parts/images]
- ⇒ **Motivates future data-efficient vision models**
- ▶ MLP which is not permutation-invariant (CF CNN filter weights) performs very badly
- ▶ More training steps does not improve data efficiency
- ? What might this imply about data-efficiency of CNNs?

Links

- ▶ `https://arxiv.org/pdf/2506.17558`
- ▶ `https://github.com/jakelevi1996/syndacate-public`
- ▶ `https://jakelevi1996.github.io/`
- ▶ `https://mvdw.uk/`