Multimodal Semantics for Affordances and Actions

Lecture 5: Reasoning with Affordances

James Pustejovsky and Nikhil Krishnaswamy

ESSLLI 2022 Summer School

Galway, Ireland

August 8-19, 2022

Course Outline

- Monday: Components of Multimodal Communication
- Tuesday: Modeling Human-Object Interactions
- Wednesday: Modeling Multimodal Common Ground
- Thursday: Communicating with Multimodal Common Ground
- Friday: Reasoning with and about Affordances

Friday's Outline

- Operationalizing Multimodal Semantics
- Reasoning with Affordances for Novelty Detection
- Reasoning with Affordances to Ground Terms to Items
- Concluding Remarks
- Open Discussion

VoxSim and Multimodal Simulations

- Human understanding depends on a wealth of common-sense knowledge; humans perform much reasoning qualitatively.
- To simulate events, every parameter must have a value
  - "Roll the ball." How fast? In which direction?
  - "Roll the block." Can this be done?
  - "Roll the cup." Only possible in a certain orientation.
- VoxML: Formal semantic encoding of properties of objects, events, attributes, relations, functions.
- VoxSim: What can situated grounding do? (Krishnaswamy, 2017)
  - Exploit numerical information demanded by 3D visualization;
  - Perform qualitative reasoning about objects and events;
  - Capture semantic context often overlooked by unimodal language processing.

Embedding space vs. Embedding space

The situated common ground consists of the following state information:

(1) a. **A**: The agents engaged in communication;
b. **B**: The shared belief space;
c. **P**: The objects and relations that are jointly perceived in the environment;
d. $\mathcal{E}$: The **embedding space** that both agents occupy in the communication.

This embedding space is not the same as the "embedding space" of a machine learning model.
But they are related/similar/analogous.

Embedding space vs. Embedding space

- An "embedding space" represents the data after dimensionality reduction
- Embedding space is of lower dimensionality than the ambient space where the "real" data lives
- ML embedding space represents the data in a computationally efficient format using vectors
- In a simulation, the numerical event parameters are elements of the ambient space
- The **rendering** and **visualization** creates the human-interpretable representation
- ML embedding space: plays to computational efficiency
- Situated embedding space: plays to human efficiency

Human vs. AI reasoning

Human reasoning

- Humans are efficient at seeking out experiences that are maximally-informative about our environments
- Young children, particularly, rapidly expand concept vocabulary with few to no examples
- Humans explore the environment to test hypotheses and *explore object affordances*

Human vs. AI reasoning

AI reasoning

- Artificial neural networks require large numbers of samples to train
- ~5-8 layers of artificial neurons to approximate the processing power of a single biological neuron
- Artificial neural networks do not easily expand to accommodate new concepts in real time
  - Forced-choice classifiers will assign one of known labels to all inputs
  - Suboptimal options: model retraining, transfer learning with pretrained models, etc.

Human vs. AI reasoning

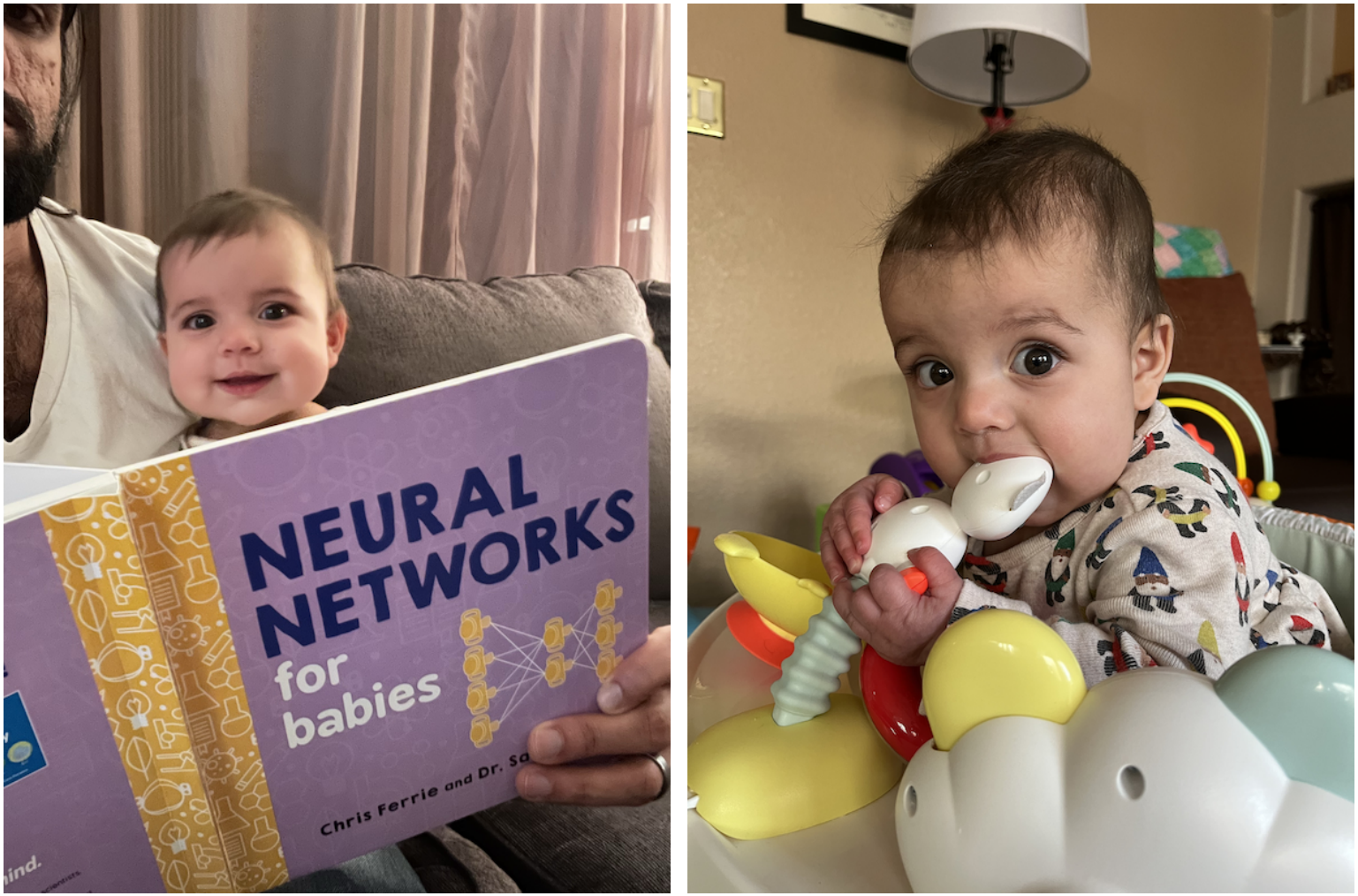


Figure: L: Not how babies learn; R: How babies learn

Reasoning with Affordances

Learning how to stack a cube

- An agent can interact with various objects and see how they behave differently under the same circumstances.
- We train a TD3 reinforcement learning policy to learn to stack two cubes.

Reasoning with Affordances

Learning how to stack a cube

- Goal: select best numerical action to keep stacked block stable
- Optimal action: places theme block centered atop destination block
  - Can be moved (perturbed) anywhere within the search space
- Episode terminates on success, or 10 failures
- Reward shaping:
  - 1000 for stacking successfully first time
    - -100 for each additional attempt (e.g., 900 for success on second try)
  - 9 for touching destination block but falling off
  - -1 for missing destination block entirely e.g., a 3-attempt episode where agent 1) misses destination block; 2) touches destination block but doesn't stack; 3) stacks successfully $=$ -1 $+$ 9 $+$ 800 $=$ 808 total return

Reasoning with Affordances

Learning how to stack a cube

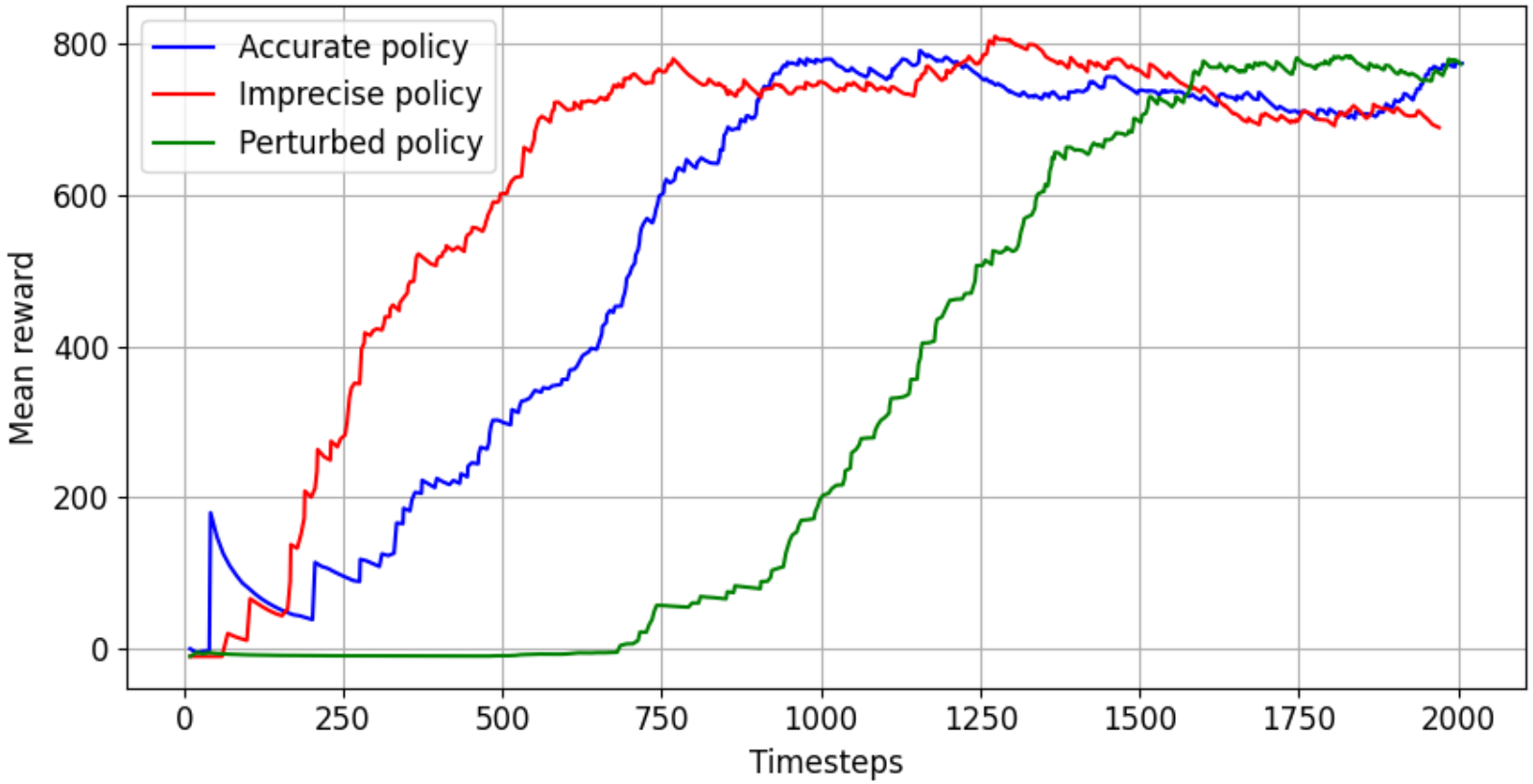
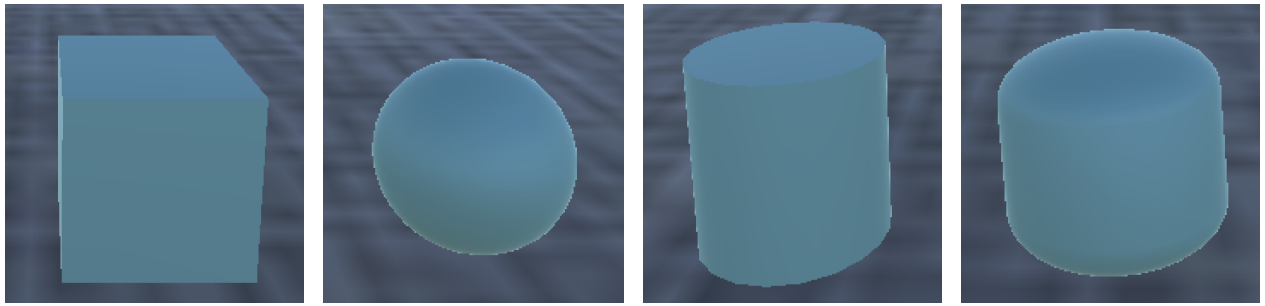


Figure: TD3 training reward plots (2,000 training episodes)

Reasoning with Affordances

Learning how to stack a cube

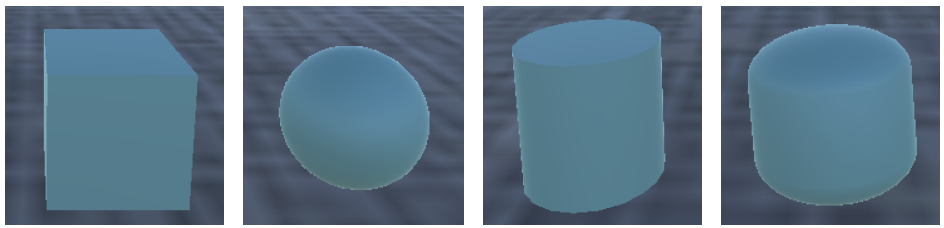
- An RL policy that can successfully stack cubes can... successfully stack cubes
  - and not much else
- We then use the successful cube-stacking policy to make the agent attempt to stack other spheres, cylinders, and capsules on a block:
  - forcing it to stack the other objects *as if they were cubes*.



Reasoning with Affordances

Learning the different affordances of objects

- This control structure allowed us to identify differences in the behaviors of the different objects in the stacking task.
- These behaviors can be described in terms like cubes stack successfully, spheres roll off, cylinders stack if oriented properly, etc.
  - Differences in behavior can be characterized in terms of the object's *affordances*.



Reasoning with Affordances

Learning the different affordances of objects

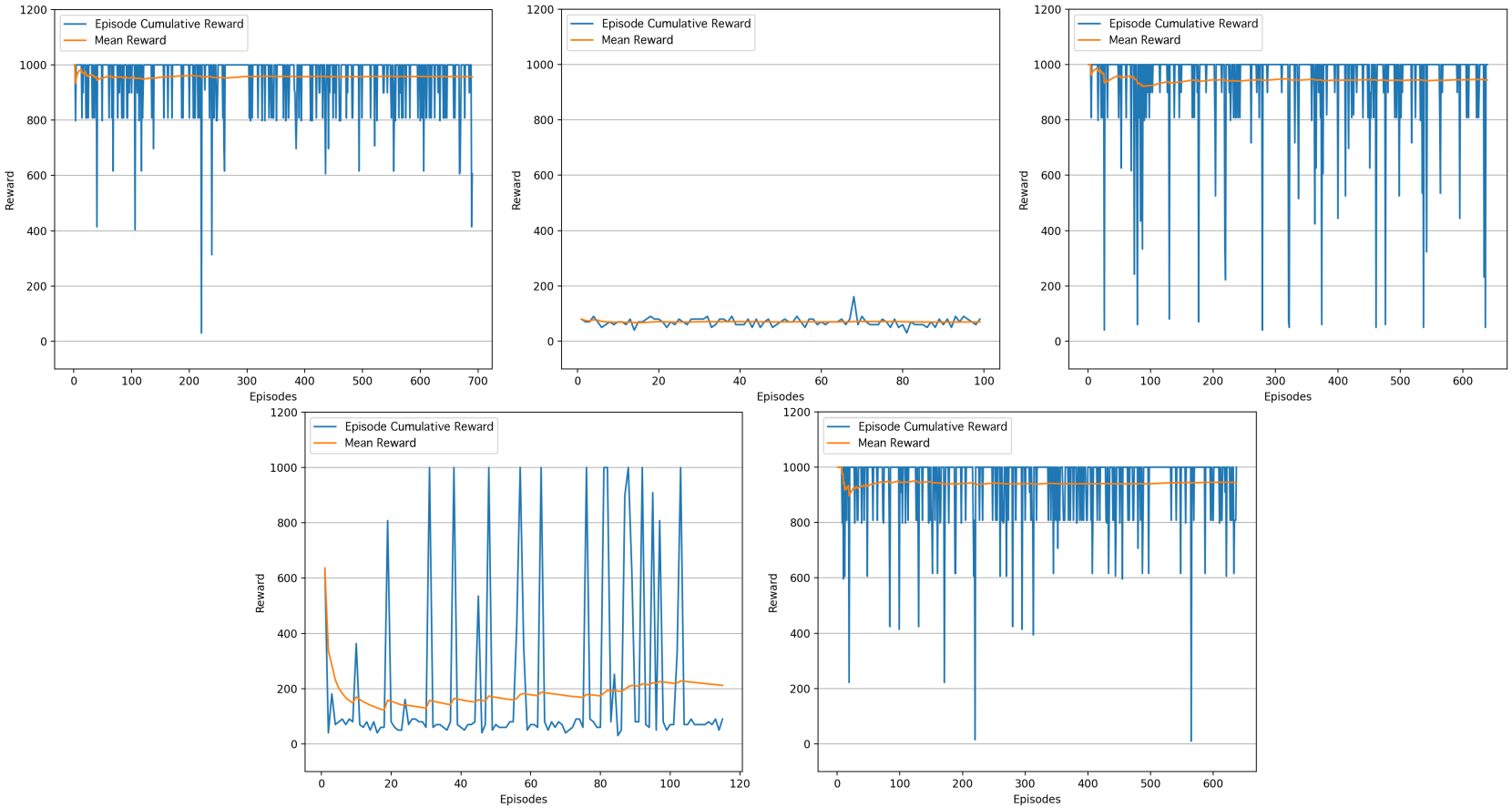


Figure: Evaluation reward plots for stacking (in order) a **cube**, **sphere**, **cylinder**, **capsule**, and **small cube** on a cube

Executing the Trained Policy over Objects

Collect object information for each stacking attempt in VoxSim

- During evalution, we store: Object type, rotation at start, offset (radians) between world and object upright axes at start, action executed, rotation and offset from world after action, state after action, reward, cumulative total and mean rewards over episode.
- At the end of the action, a small "jitter" is applied, to simulate the small force exerted on an object when it is released from a grasp, in an otherwise hyper-precise virtual environment.
- The post-action jitter is a small force perpendicular to the object's axis of symmetry, therefore implicitly encoding information about the object's habitat, *as encoded in VoxML*.
- The stackability (or lack thereof), encoded in the distribution of state observations, implicitly encodes an affordance.

Executing the Trained Policy over Objects

Collect object information for each stacking attempt in VoxSim

Object	Jitter			θ after Action	Stack Height
cube	-1.472×10^{-4}	0	2.021×10^{-4}	0.02238165	2
sphere	8.165×10^{-5}	0	-2.363×10^{-5}	2.134116	1
cylinder	0	0	2.5×10^{-4}	0.01457105	2
cylinder	0	0	2.5×10^{-4}	1.570793	1

Table 1: Observations gathered during stacking task with multiple objects.

- A cube, which is flat on all sides, can rest stably (multiples of $\frac{\pi}{2}$)
- The sphere rolls off, does not stack (height 1), comes to rest at an arbitrary angle.
- The cylinder shares properties with cubes (flat ends) and others with spheres (round sides), in the last two rows.
  - the cylinder stacks successfully (height 2), and is resting upright ($\theta \approx 0$)
  - it rolls off with the cylinder on its side ($\theta \approx \frac{\pi}{2}$)

Object Similarity Analysis

CCA over raw behavioral data

	CUBE	SPH.	CYL.	CAP.	SM. CUBE
CUBE	1	0.396	**0.958**	0.686	**0.808**
SPHERE	0.399	1	0.366	**0.832**	0.376
CYLINDER	**0.974**	0.366	1	0.692	0.528
CAPSULE	0.688	**0.832**	0.692	1	0.511
SM. CUBE	**0.808**	0.376	0.527	0.511	1

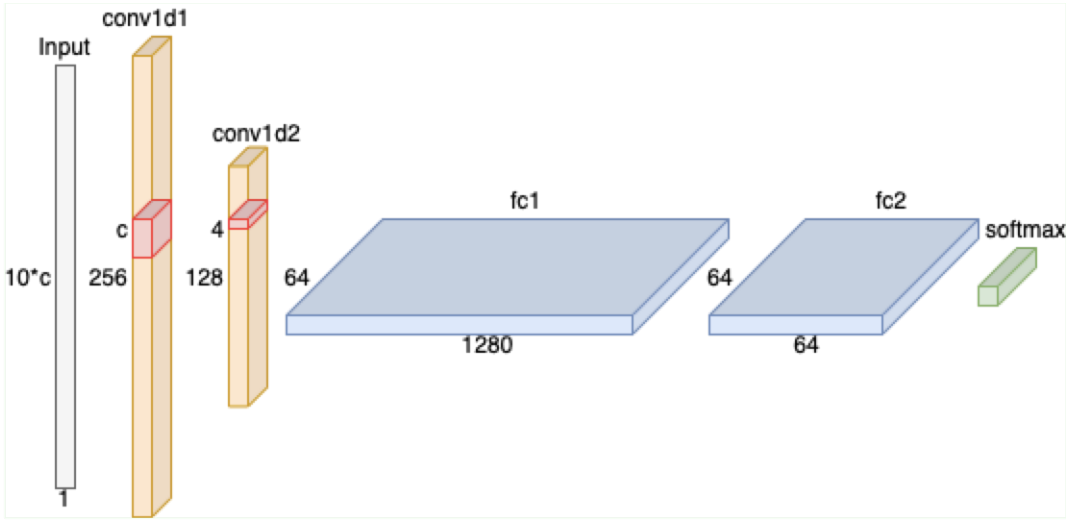
Figure: Canoncical correlation analysis (CCA) results

- CCA confirms intuitions displayed in evaluation reward plots
- Anomalies: small cube is less similar to large cube than cylinder is?
- Raw data still needs some processing

Using Habitat and Affordance Embeddings

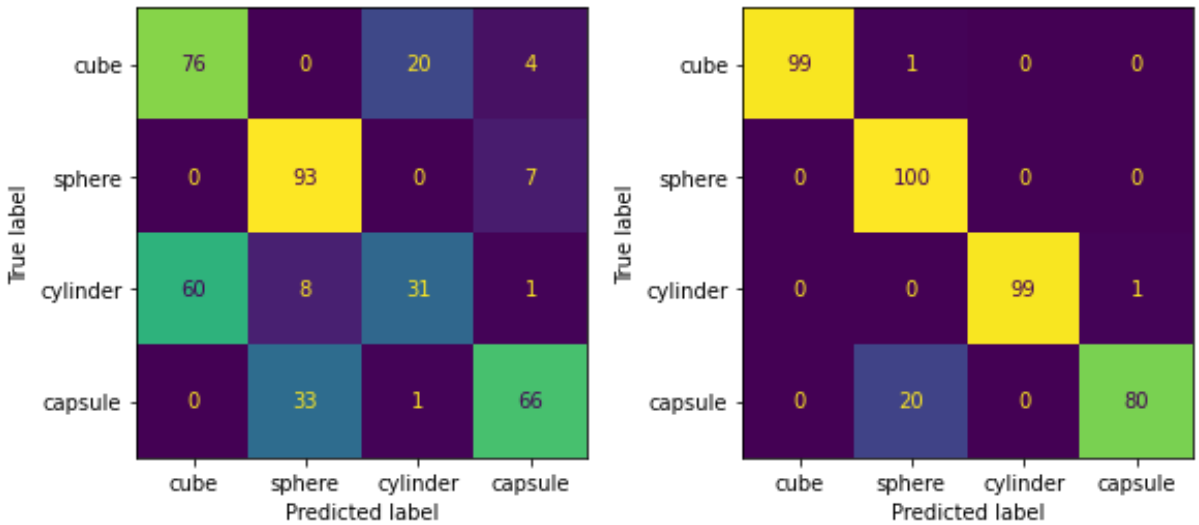
Training a model to predict object type from its behavior in stacking task

- 1D convolutional neural net (2 convolutional layers—256 and 128 hidden units, 2 64-unit fully-connected layers, and a softmax layer)
- We train for 500 epochs using Adam optimizer, batch size of 100 (= 10 episodes), learning rate of 0.001.



Using Habitat and Affordance Embeddings

Training a model to predict object type from its behavior in stacking task



Left chart shows results without input of implicit habitat and affordance information encoded in post-action jitter (66.5%). Right chart shows results with those input features (94.5%).

Using Habitat and Affordance Embeddings

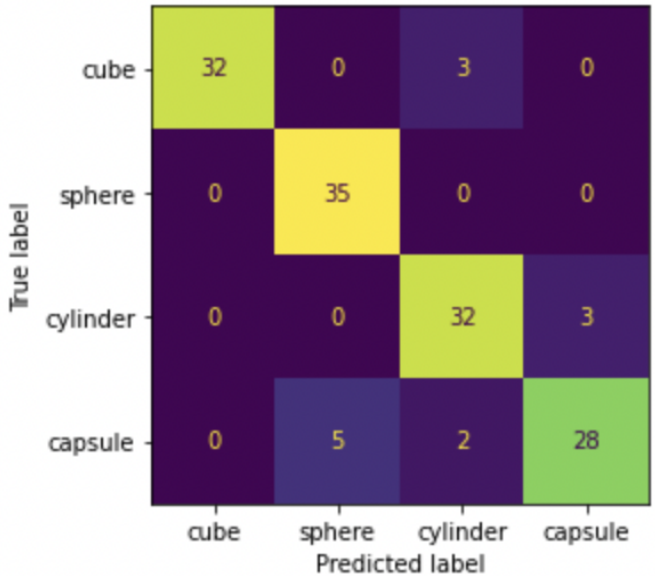
Training a model to predict object type from its behavior in stacking task

Can't you just tell if objects are different by looking at them?

Comparing Behavior with Visual Clues

Humans can tell that objects are different by visual clues

- We compared the performance of the behavior-based classifier to a 2D CNN CIFAR-10 style object detector.
- This classifier achieves a validation accuracy of 97.5%, but when evaluated against an unseen test set of 140 novel images of the four object classes (35 images each), accuracy falls below the behavior-based classifier, to **90.7**%.



Mistakes Made by the Visual Classifier

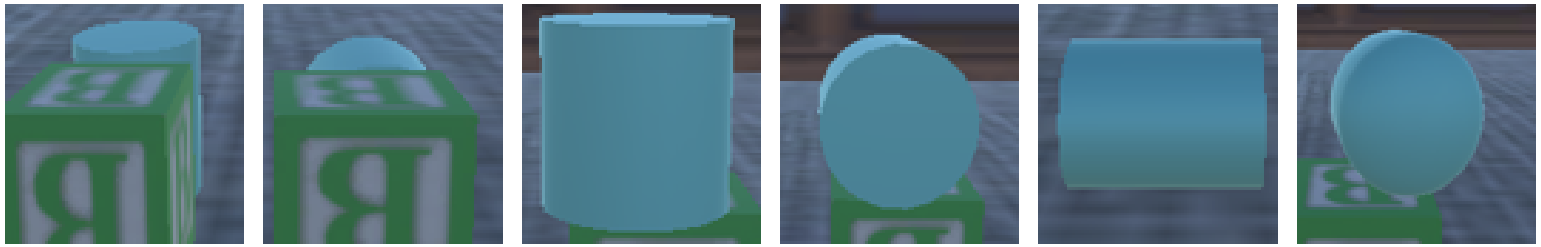
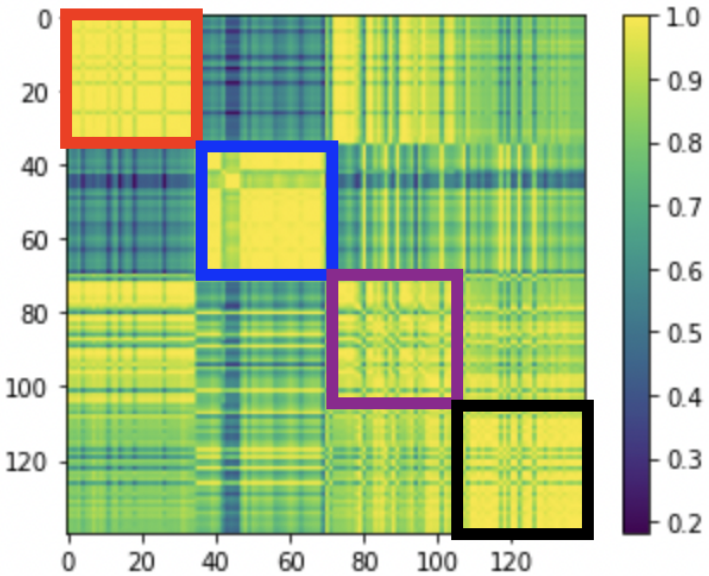


Figure: Sample of misrecognized objects. From left: cylinder misrecognized as cube (1), and capsule (2), cylinder misrecognized as cube (3), sphere (4), and capsule (5), and capsule misrecognized as sphere (6).

Evaluating the Embedding Vectors

- While some objects are visually distinct, like cube vs. sphere, other object classes are more difficult to distinguish visually.
- To confirm this, we draw out the 64-dimensional embedding vectors from the final fully-connected layer.
- These can be used to quantitatively assess the similarity of different input samples to each other



Cosine similarity matrix of visual embedding vectors from 2D CNN.

Multimodal Semantics

Sight and behavior

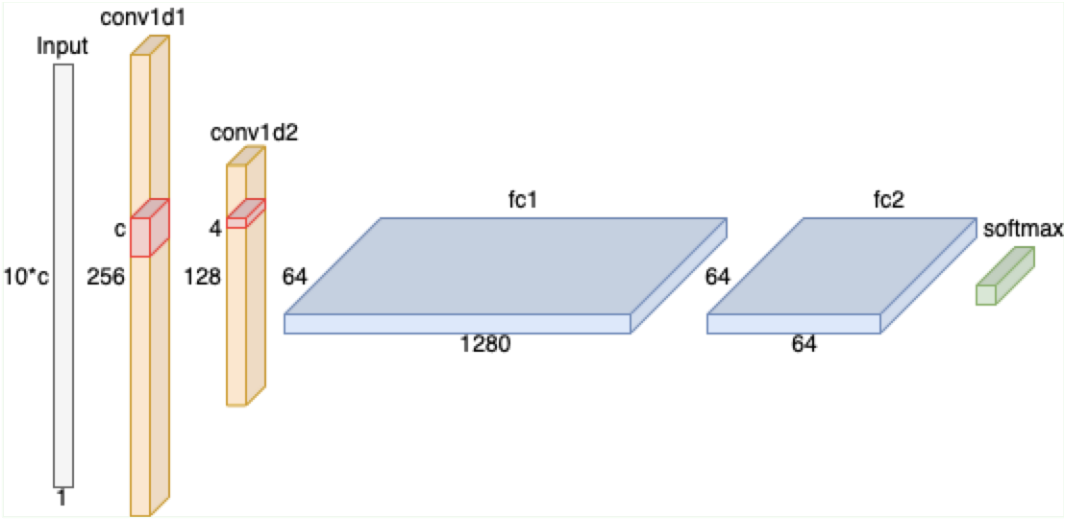
- Where vision may be ambiguous, behavior may be a stronger signal
- **Embodiment** drives **affordance** reasoning
- Embodiment (and grounding) is more than only language+vision
- Integrated multimodal semantics a la VoxML facilitates extraction of quantified *embodied* information from an environment
- Which leads to...

Novel Class Detection

- Weakness of a forced-choice classifier:
  - *Will* output one of its known classes for any input;
  - Softmax layer obscures probability, confidence, representation.
- Strength of a forced-choice classifier:
  - Embedding-level representations preserve similarity across dimensions.
- Novelty detection procedure:
  1. Identify which known class an object is most similar to;
  2. Determine if new object is different enough from most similar known class to be considered novel.

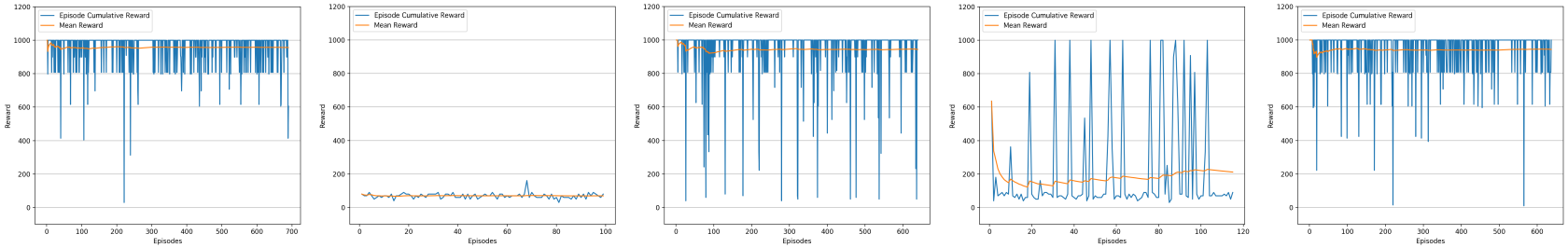
Novel Class Detection

- CCA shows that cube and sphere are the two mos dissimilar objects
- Correlates to prototypical "stackable" and "unstackable" objects
- Instantiate behavior CNN classifier with two classes: cube/sphere



Novel Class Detection

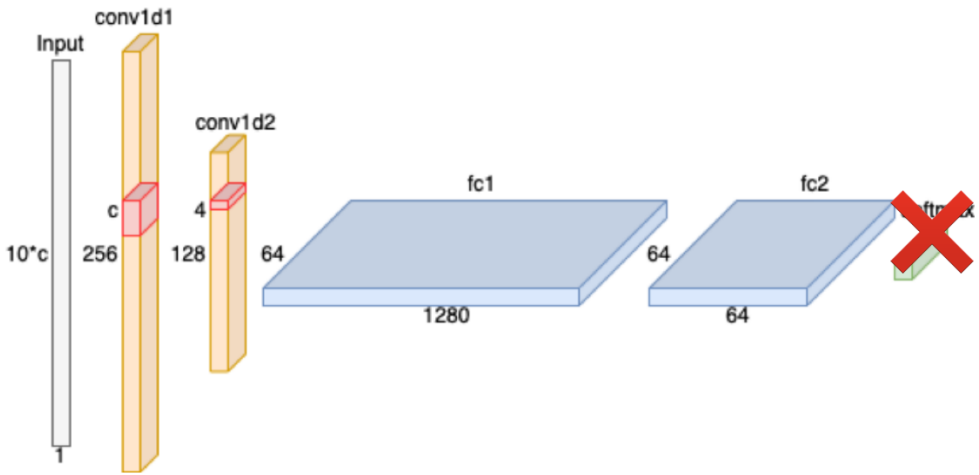
- Given known classes cube and sphere, cylinders usually classified as cubes, capsules usually classified as spheres



- Recapitulates observations from CCA and evaluation reward plots

Novel Class Detection

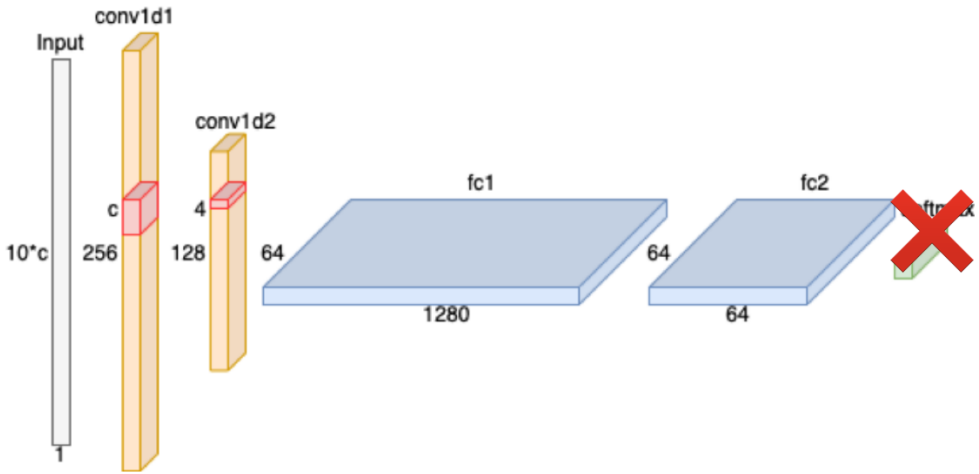
- If something is classified as *sphere* is it actually a sphere?
  - Or just sphere-*like* and I don't know any better?



- Get the embeddings for the input sample(s) and compare them to embeddings model "knows" belong to the predicted class (i.e., training embeddings)

Novel Class Detection

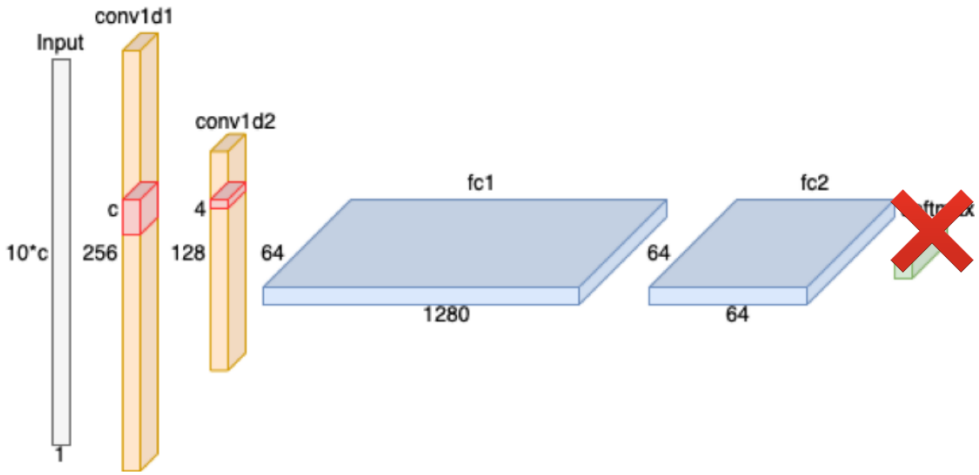
Odd properties of embedding spaces



- Not as simple as constant similarity threshold between embedding vectors
- Differences in weight initialization may mean embeddings are differently distributed each time

Novel Class Detection

Odd properties of embedding spaces



- Same network, same data, same training, different initial weights may mean
  - One trained model has dispersed or even near-isotropic vectors
  - One trained model clusters all vectors close together in high-D space; absolute distance between classes is small

Novel Class Detection

- **This is now an outlier detection problem**
- Let $\overrightarrow{\mu_S}$, $\overrightarrow{\sigma_S}$ be the mean, stdev of the known class
- Let $\overrightarrow{\mu_N}$ be the mean of the new batch
- Let $\overrightarrow{v}$ be a single sample

Novel Class Detection

- Let $\overrightarrow{\mu_S}$, $\overrightarrow{\sigma_S}$ be the mean, stdev of the known class
- Let $\overrightarrow{\mu_N}$ be the mean of the new batch
- Let $\overrightarrow{v}$ be a single sample
- Compute $\rho_{\overrightarrow{v}}$, the ratio of $\overrightarrow{v}$'s distance from $\overrightarrow{\mu_S}$ to the overall spread of embedding vectors in class S
  - $\rho_{\overrightarrow{v}} = \frac{\cos(\overrightarrow{\mu_S}, \overrightarrow{v})}{\cos(\overrightarrow{\mu_S}, \overrightarrow{\mu_S} + \overrightarrow{\sigma_S})}$
- If $\rho_{\overrightarrow{v}} > 1$, add $\overrightarrow{v}$ to the set of outliers $\overrightarrow{o} \in O$
  - (outliers are defined relative to a sample set)

Novel Class Detection

- Compute $\rho_{\overrightarrow{v}}$, the ratio of $\overrightarrow{v}$'s distance from $\overrightarrow{\mu_S}$ to the overall spread of embedding vectors in class S
  - $\rho_{\overrightarrow{v}} = \frac{\cos(\overrightarrow{\mu_S}, \overrightarrow{v})}{\cos(\overrightarrow{\mu_S}, \overrightarrow{\mu_S} + \overrightarrow{\sigma_S})}$
- If $\rho_{\overrightarrow{v}} > 1$, add $\overrightarrow{v}$ to the set of outliers $\overrightarrow{o} \in O$
  - (outliers are defined relative to a sample set)
- Outlying samples may still belong to the known class
  - (e.g., sometimes a cube fails to stack properly due to bad placement)
- If $\frac{\rho_{\overrightarrow{o}} - \mu_\rho}{\mu_\rho} > 3$, remove $\overrightarrow{o}$ from outlier set

Novel Class Detection

- Outlying samples may still belong to the known class
  - (e.g., sometimes a cube fails to stack properly due to bad placement)
- If $\frac{\rho_{\overrightarrow{o}} - \mu_\rho}{\mu_\rho} > 3$, remove $\overrightarrow{o}$ from outlier set
- **What this does is define a subspace in 64D space bounded by non-extreme outlier vectors of each sample**
- Compute ratio of outliers in known class to outliers in new batch:
  - Outlier ratio $OR = \frac{\sum_{\overrightarrow{o_N} \epsilon O_N} \rho_{\overrightarrow{o_N}}}{\sum_{\overrightarrow{o_S} \epsilon O_S} \rho_{\overrightarrow{o_S}}}$
- Scale OR by distance between known class and new batch means, then normalize by spread of embeddings in known class times denominator OR

Novel Class Detection

- What this does is define a "core" subspace in 64D space bounded by non-outlier vectors of each sample
- Compute ratio of outliers in known class to outliers in new batch:
  - Outlier ratio $OR = \frac{\sum_{\overrightarrow{o_N} \in O_N} \rho_{\overrightarrow{o_N}}}{\sum_{\overrightarrow{o_S} \in O_S} \rho_{\overrightarrow{o_S}}}$
- Scale OR by distance between known class and new batch means, then normalize by spread of embeddings in known class times denominator OR
- Define a "dissimilarity threshold" T
- **If** $\frac{OR \times \mathbf{cos}(\overrightarrow{\mu_S}, \overrightarrow{\mu_N})}{\mathbf{cos}(\overrightarrow{\mu_S}, \vec{\mu_S} + \overrightarrow{\sigma_S}) \times \sum_{\overrightarrow{o_S} \in O_S} \rho_{\overrightarrow{o_S}}} > T$**, the new input samples likely belong to new class!**

Novel Class Detection

Results

- Evaluation: for a model beginning with cube and sphere labels:
  - there are 2 novel objects that may occur: cylinder and capsule
  - and small cube, which is the same as cube
    - (not considering size as a distinguishing factor as parameters indicating size are not captured in the raw data)
- Correct result: detecting cylinder and capsule as novel, and cube, sphere, and small cube as not novel
  - Correct result for model including cylinder: detecting capsule as novel, and all other classes as not novel, etc.
- Evaluation is performed 5 times in each condition to allow for differences in weight initialization

Novel Class Detection

Results

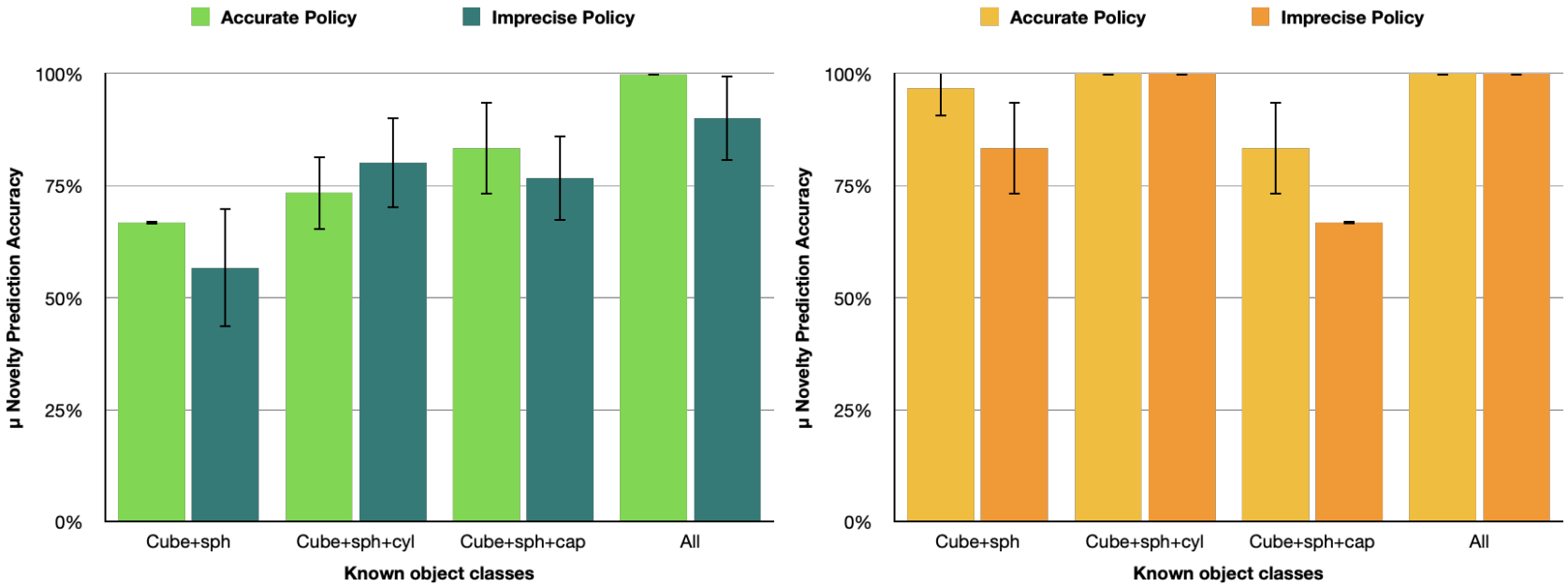
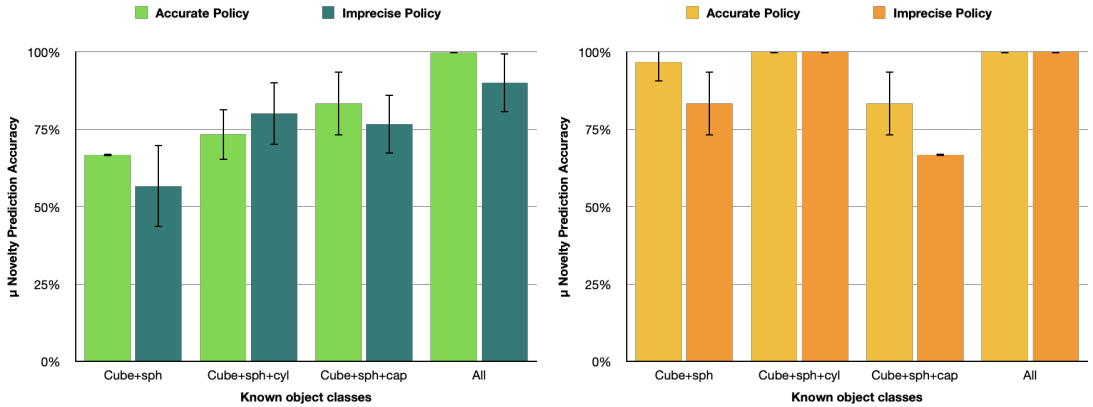


Figure: Novel class detection accuracy without implicit habitat enoding (left) and with (right)

Novel Class Detection

Discussion



- Can correctly identify the novelty of cylinders and capsules based on behavior alone
- Small cubes identified as same type as large cubes
- Imprecise policy data slightly more challenging
- Including implicit habitat/affordance information increases performance by 25%

Novel Class Detection

Discussion

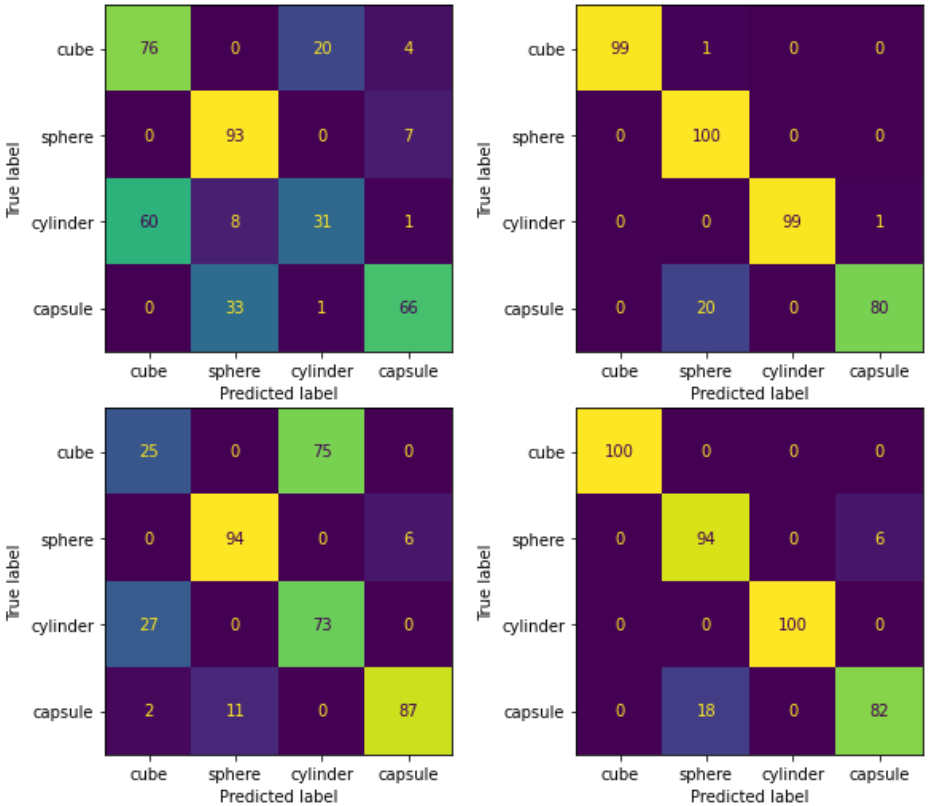
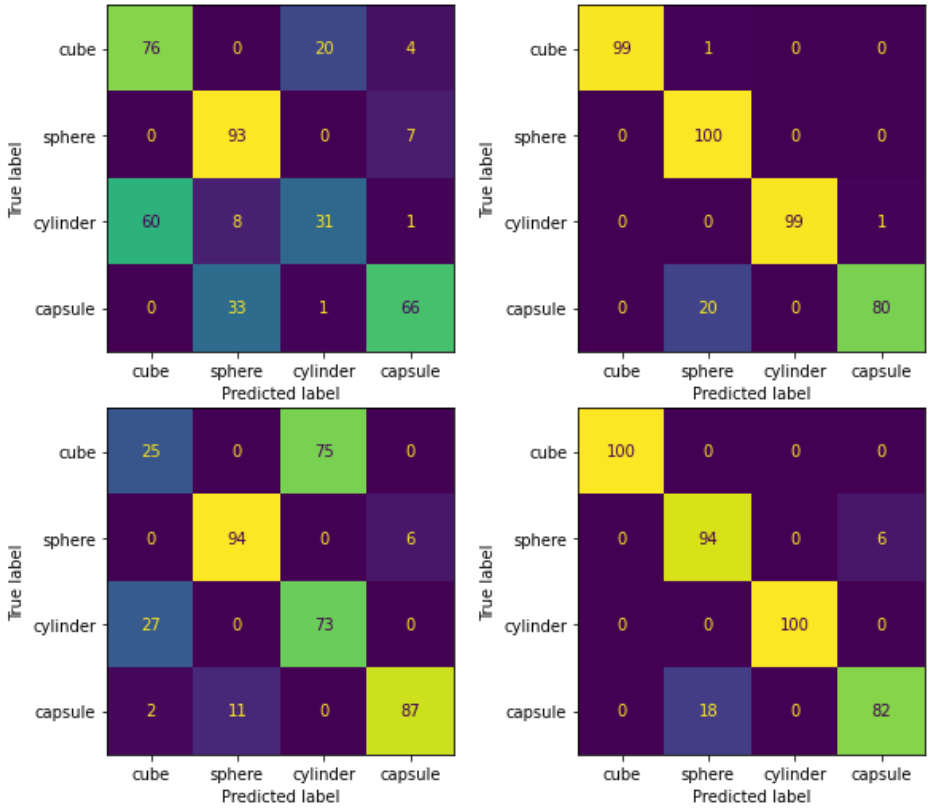


Figure: Aggregated 1D CNN classifier outputs over dev-test set. T: accurate policy evaluation; B: imprecise policy evaluation; L: without VoxML-derived inputs; R: with VoxML-derived inputs

Novel Class Detection

Discussion

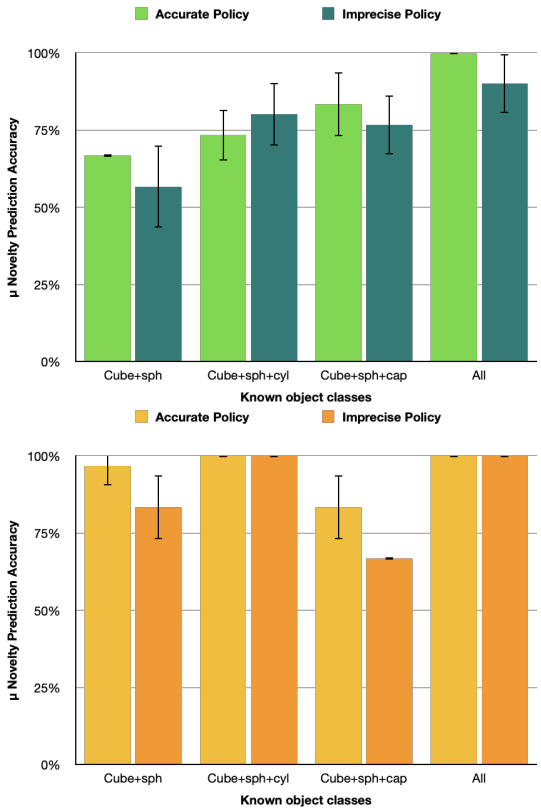
- Without habitat information, classifier confuses cylinders and cubes, capsules and spheres
  - Same patterns as original two-class classifier



Novel Class Detection

Discussion

- Order of concept acquisition matters
  - Detecting capsule concept before cylinder impedes cylinder detection
- Capsule is more distinct from sphere than cylinder is from cube in its behavior
- Capsule embedding vectors take up more "space"
  - Makes the subtle cube/cylinder distinction harder to detect



Metacognitive Agents

- Method approximates certain metacognitive processes
- Provides potential step toward computational "fast mapping"

Isn't fast mapping usually used in the context of language acquisition? Where's the language?

Multimodal Grounding

A new approach

- Modern multimodal grounding is usually achieved through large multimodal models
  - e.g., using contrastive training methods a la CLIP
- While these are often powerful in appropriately designed tasks (e.g., grounding items to terms in a specified region), they do not address
  1. Data and computational power requirements
  2. Grounding items to terms in a dynamically updating embedding space

Multimodal Grounding

A new approach

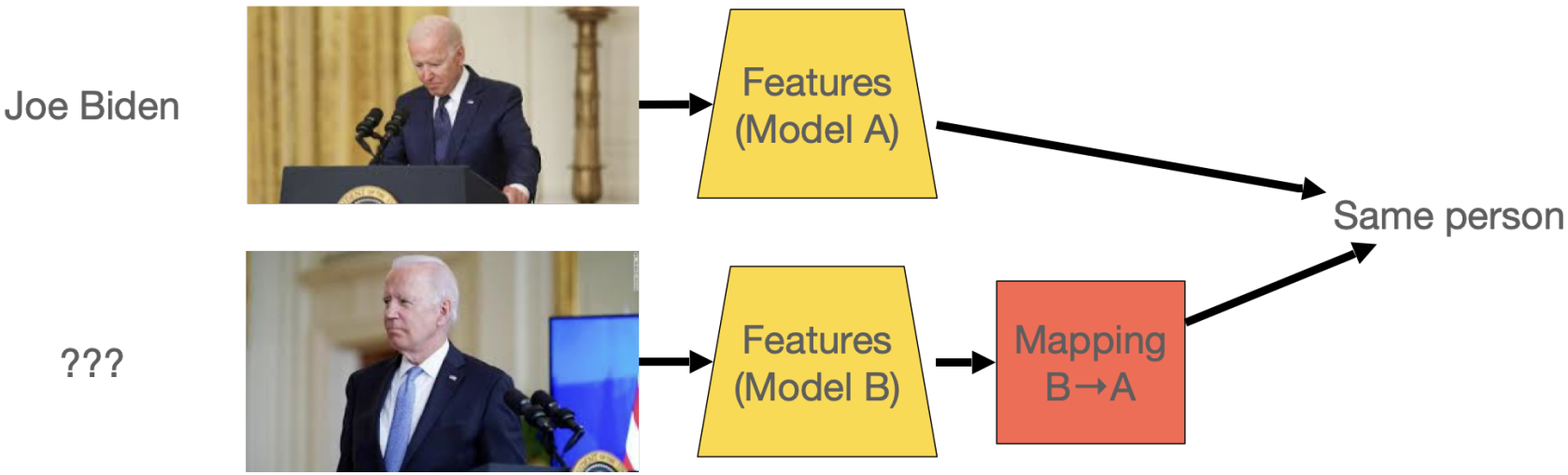


Figure: Transforming features from different embedding spaces into common space

- Findings from the vision community: CNN embedding spaces share interchangeability up to an affine transformation $M_{A \to B}$

Multimodal Grounding

A new approach

- Properties of transformation $M_{A \to B}$:
- Given two embedding spaces A, B, where embeddings in each are of dimensionality d_A, d_B
  - $M_{A \to B} \in \mathbb{R}^{d_A} \times \mathbb{R}^{d_B}$ that transforms the representation of $C \in \mathbb{R}^{d_A}$ to $\sim C \in \mathbb{R}^{d_B}$
  - Computed by minimizing distance between paired (equivalent) points in A and B
- Given two sets of objects (vectors) X and Y, are they the "same" under an affine transformation?
- Task is now to recover that transformation

Multimodal Grounding

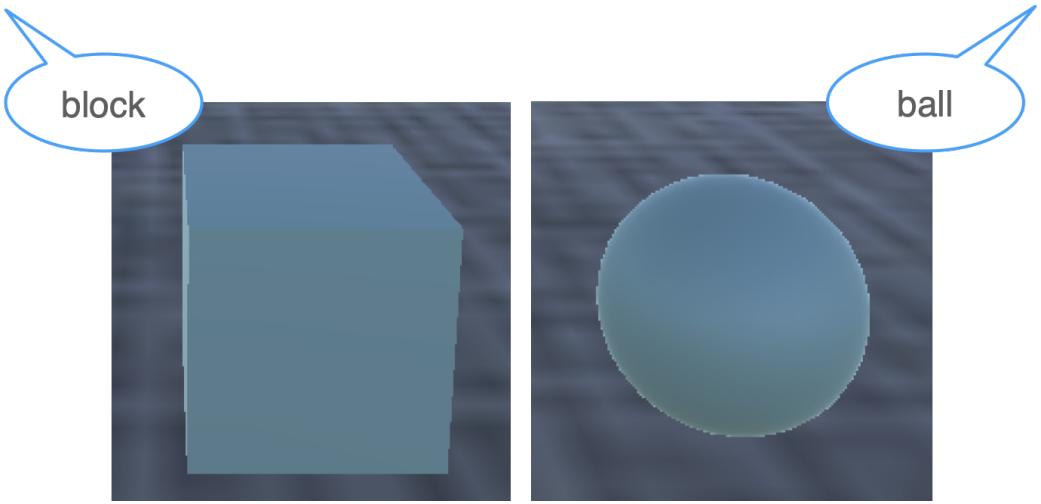
A new approach

- Different images of the same person, if both classified correctly, should have similar but not identical embedding vectors
- **Similarly, object embeddings from behavioral data are distinct**
  - They define a subspace
- Similar to contextualized word embeddings a la BERT
  - Not identical, but point in the same direction
- Use one set of vectors as inputs to a regressor and the other as outputs
- **If inputs and outputs preserve similar information and distinctions...**

Multimodal Grounding

A new approach

- $M_{A \to B}$ is the weights of a mapping function $f(x; W)$
- $f(x; W) : U \subset \mathbb{R}^{d_A} \to V \subset \mathbb{R}^{d_B}$, where $d_B \leq d_A$, s.t. $f(x \in U) \approx x \in V$
  - inputs: U; outputs: V
  - Application of f to any element of U should approximate the equivalent element of V



Multimodal Grounding

A new approach

- Imagine two agents, a "child" agent playing with objects, and a "parent" agent generating utterances containing object references.
  - Model child agent as 1D CNN behavior classifier, parent agent as Transformer language model

Child:

Parent:

"The **block** is flat."

Multimodal Grounding

A new approach

Child:

Parent:

"The **block** is flat."

- Take 64D embeddings from CNN model (grounded object instances) as outputs
- Take 768D embeddings from Transformer model (contextualized token embeddings) as inputs
- Compute $M_{A \to B}$ as 768×64 affine matrix
  - Do new mentions of the same contextualized tokens cluster with the correct objects?
  - Are different senses of the same word distinct from the object mentions?

Multimodal Grounding

A new approach

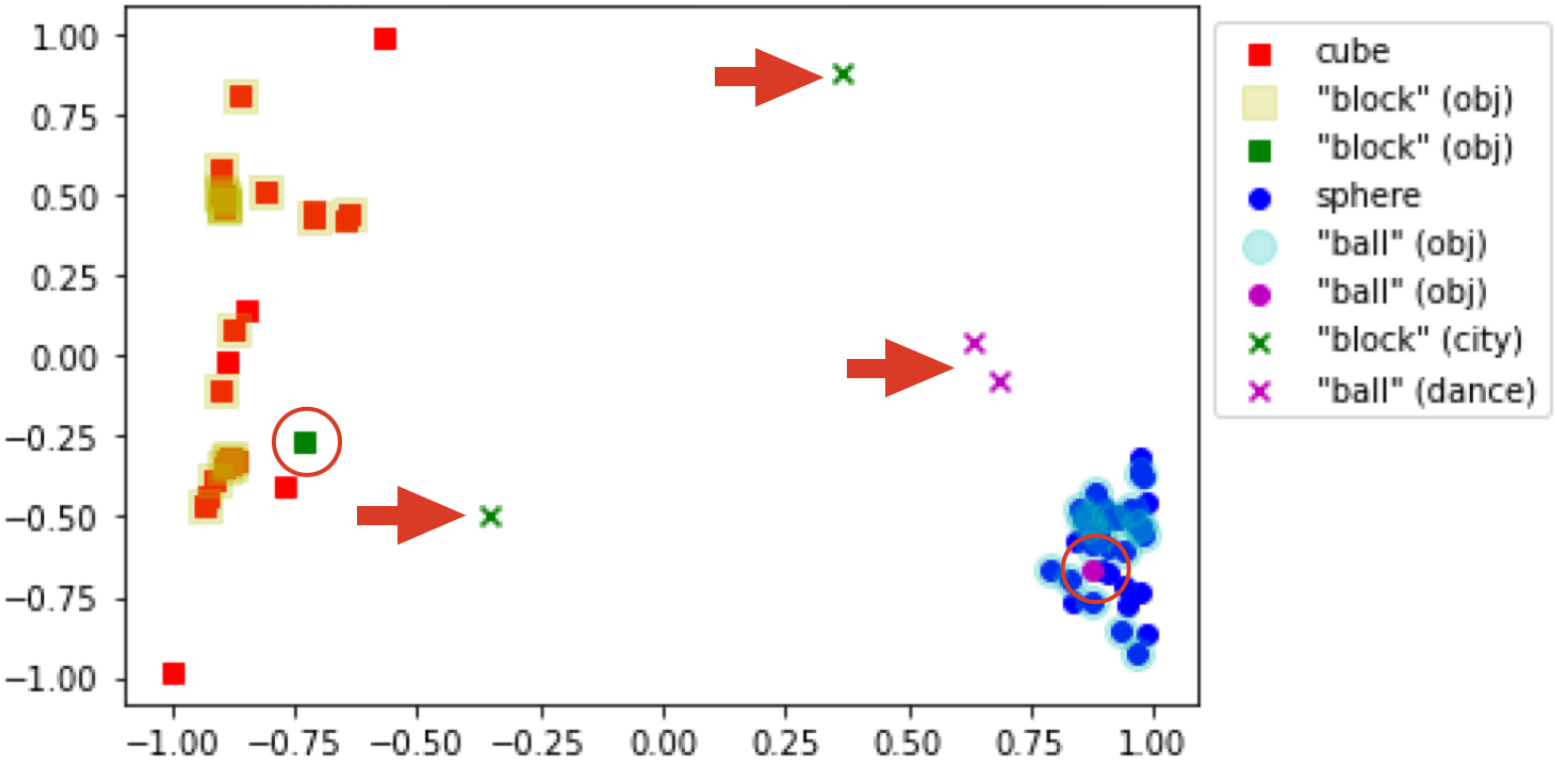
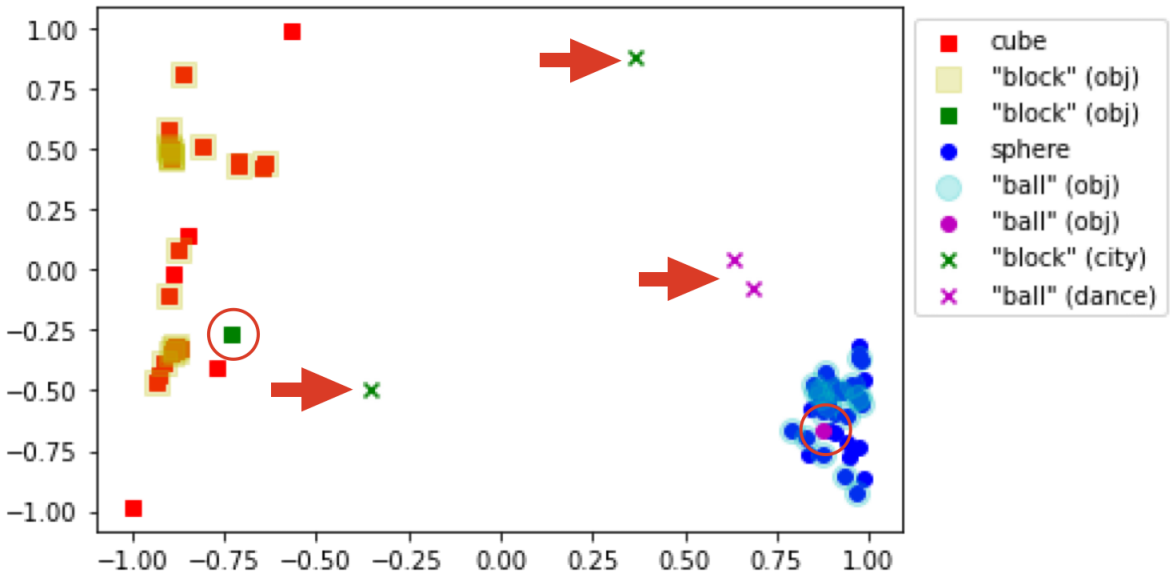


Figure: BERT word vectors mapped into object vector space (reduced to 2 dimensions)

Multimodal Grounding

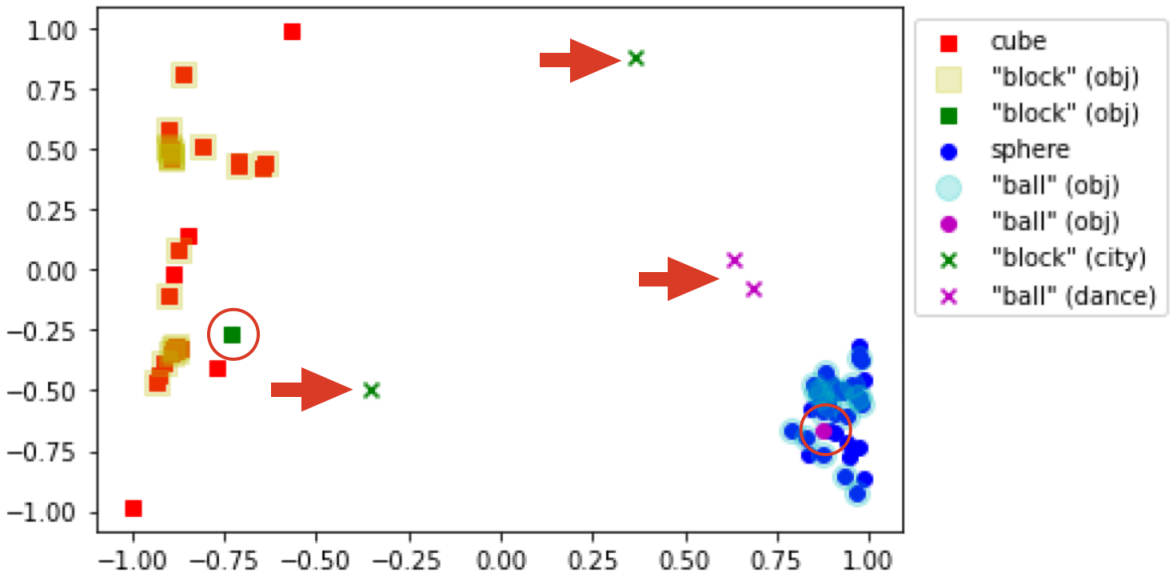
A new approach



- Novel instances of the known sense of the word "block" and "ball" map into the sphere/cube subspaces
- Different senses of the same words fall outside of those subspaces

Multimodal Grounding

A new approach



- Enables "decontextualized reference"
- Agent can discuss items not present while maintaning grounding to concept
- Facilitates transfer between otherwise dissimilar models

Computing Requirements

- Through effective use of existing resources, situated grounding through simulation creates rich data for tasks like affordance reasoning, metacognition, and grounding
- None of the methods discussed above require lengthy GPU training or specialized hardware
- More compute helps, but problems are still tractable on workstation or even a laptop

Computing Requirements

- Each model uncovers different information
- *Common ground* brings different information spaces together
- One model can use its strengths to ameliorate another's weaknesses
- Model augmentation through simple techniques by "grafting" representations together

Interfacing VoxWorld to New Domains and Applications

- Teacher Assistant in Classroom
  - VoxWorld can model classroom dynamics and actions
- Interpreted Augmented Reality
  - AR objects are semantically interpreted in VoxML, can be reasoned about.
- Sequence-to-sequence models;
  - description ⇒ animation
  - animation ⇒ description

VoxML Representation of Object Transformations

- Objects afford actions (pealing)
- Actions create new habitats for new actions (slicing)

$$
\left[\begin{array}{l}
\textbf{onion} \\
\text{LEX} = \left[\begin{array}{l}\text{PRED} = \textbf{onion} \\ \text{TYPE} = \textbf{physobj}\end{array}\right] \\
\text{TYPE} = \left[\begin{array}{l}\text{HEAD} = \textbf{ellipsoid[1]} \\ \text{COMPONENTS} = \textbf{vegetable[1], skin[2]} \\ \text{CONCAVITY} = \textbf{flat} \\ \text{ROTATSYM} = \{Y\} \\ \text{REFLECTSYM} = \{XY, YZ\}\end{array}\right] \\
\text{HABITAT} = \left[\begin{array}{l}\text{INTR} = \ldots \\ \text{EXTR} = \text{[2]}\left[\text{UP} = align(Y, \mathcal{E}_{\perp Y})\right]\end{array}\right] \\
\text{AFFORD\_STR} = \left[\begin{array}{l}\text{A}_1 = H \rightarrow [grasp(x, [1])] \\ \text{A}_2 = H \rightarrow [hold(x, [1])]lift(x, [1]) \\ \text{A}_3 = H \rightarrow [roll(x, [1])] \\ \text{A}_4 = H_{[2]} \rightarrow [grasp(x, [1]), \\ \quad peel(x, [2], from([1])))]peeled(x, [1])[\textbf{3}] \\ \text{A}_5 = H_{[3]} \rightarrow [grasp(x, [1]), \\ \quad slice(x, [1])]sliced(x, [1])[\textbf{4}]\end{array}\right] \\
\text{EMBODIMENT} = \left[\begin{array}{l}\text{SCALE} = <\textbf{agent} \\ \text{MOVABLE} = \textbf{true}\end{array}\right]
\end{array}\right]
$$

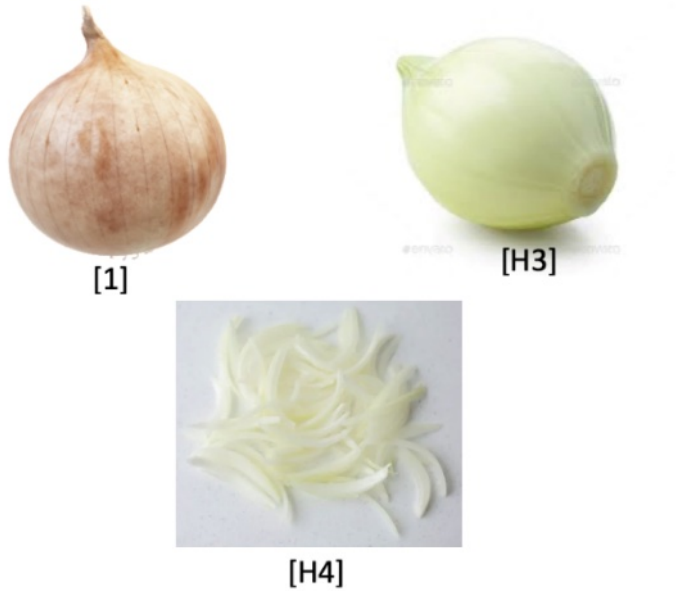
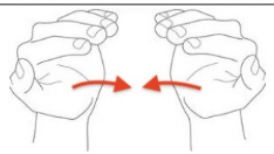
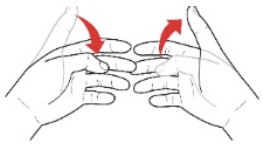
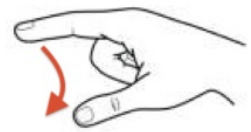


Figure 1: Left: VoxML representation of an onion; Right: typing (1: ellipsoid) and habitats (H3: peeled; H4: sliced) of an onion.

Cross-modal VoxML representation of Actions

Alignment of multiple channels

- Video sequence and key frames
- Gesture actions depicting stages of a plan
- Textual descriptions of actions
- Iconic displays denoting orientations

Frames			
		4s	8s
Time			
Word	*fold*	*roll*	*tuck*
Speech	*Bring both the left and the right side of your wrap about 1–3 in (2.5–7.6 cm) towards the middle of the wrap.*	*To make your roll, lift up on the bottom edge of your wrap, and move it towards the center about one third of the way up.*	*Tuck your filling back into the wrap as you roll it up.*
Gesture			
Icon	⇒ ⇐	⇑	⇓

Multimodal Processing Model (MPM) with VoxWorld

Grounding between VoxML, vision, and blackboard

- Video regions ground symbolic representations in VoxML
- VoxML representations align with blackboard states

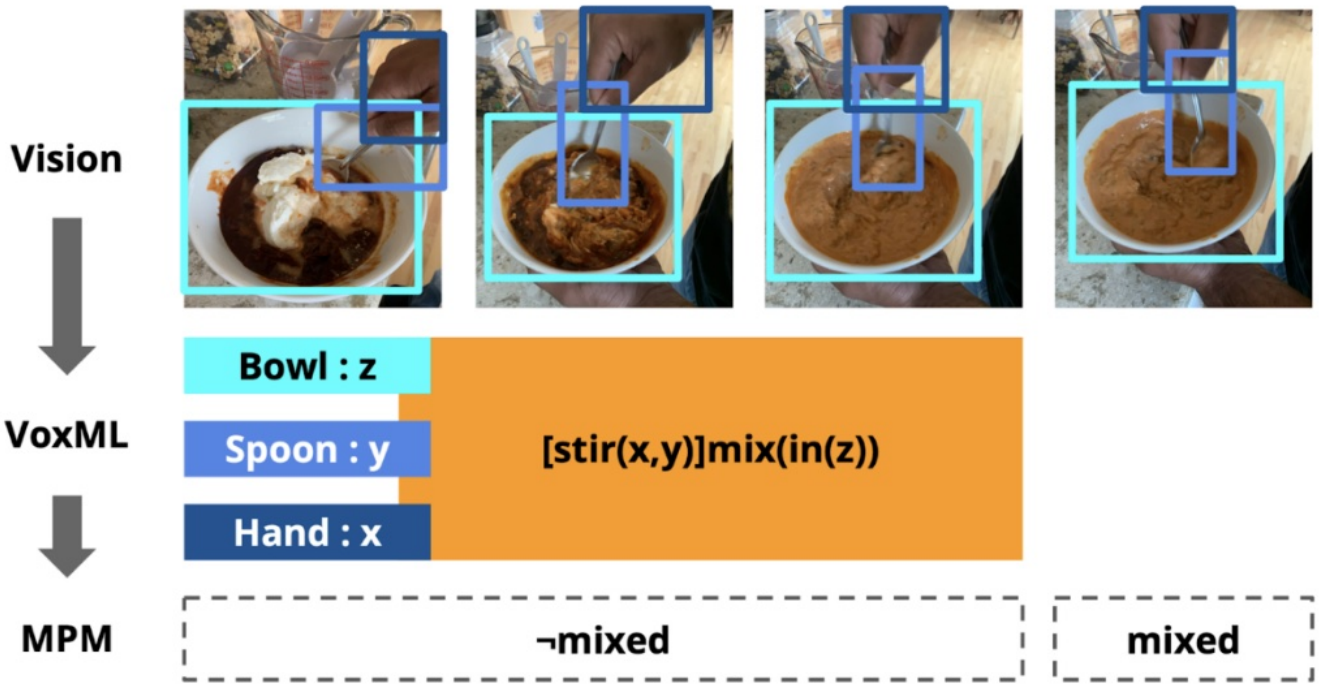
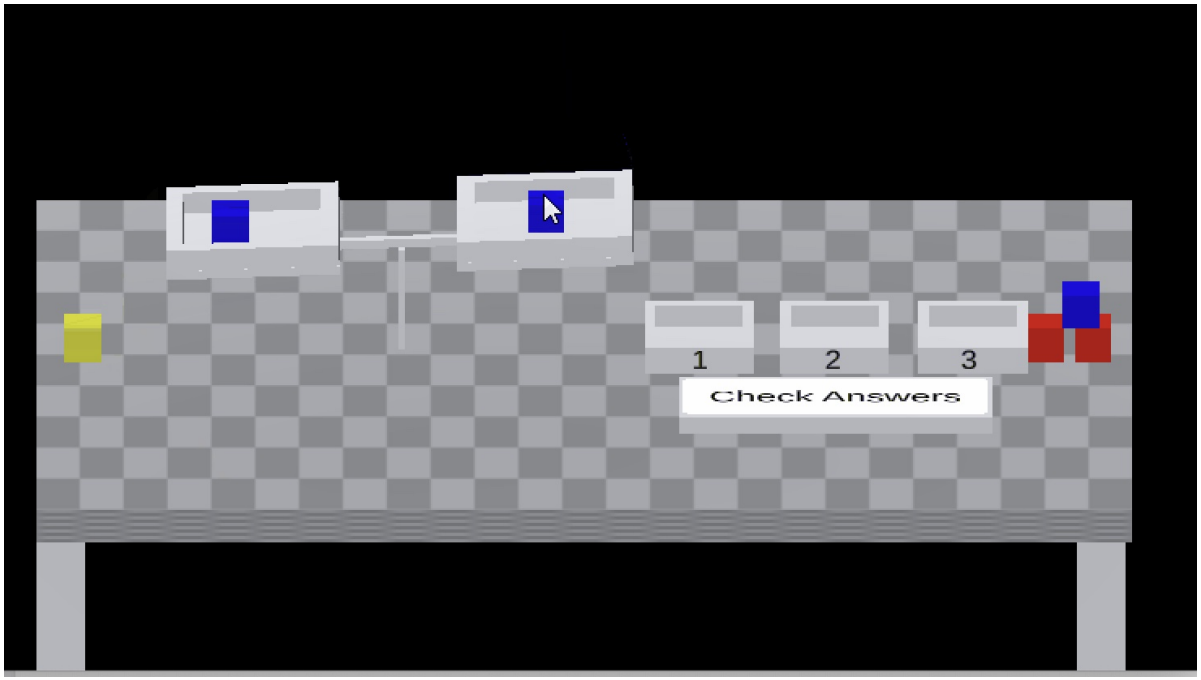


Figure 3: Flow of information between vision (neural), VoxML representation (neurosymbolic) and MPM, incl. PDDL (symbolic).

Simulating Classroom Lesson Plans

- Modeling classroom interactions allows us to generate hundreds of alternative dialogues using multimodal channels.



Concluding Remarks