Abstraction and Analogy in Artificial Intelligence

Melanie Mitchell, Sante Fe Institute

neuromatch academy

A PROPOSAL FOR THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE

J. McCarthy, Dartmouth College
M. L. Minsky, Harvard University
N. Rochester, I.B.M. Corporation
C.E. Shannon, Bell Telephone Laboratories

August 31, 1955

We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

A PROPOSAL FOR THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE

J. McCarthy, Dartmouth College
M. L. Minsky, Harvard University
N. Rochester, I.B.M. Corporation
C.E. Shannon, Bell Telephone Laboratories

August 31, 1955

We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

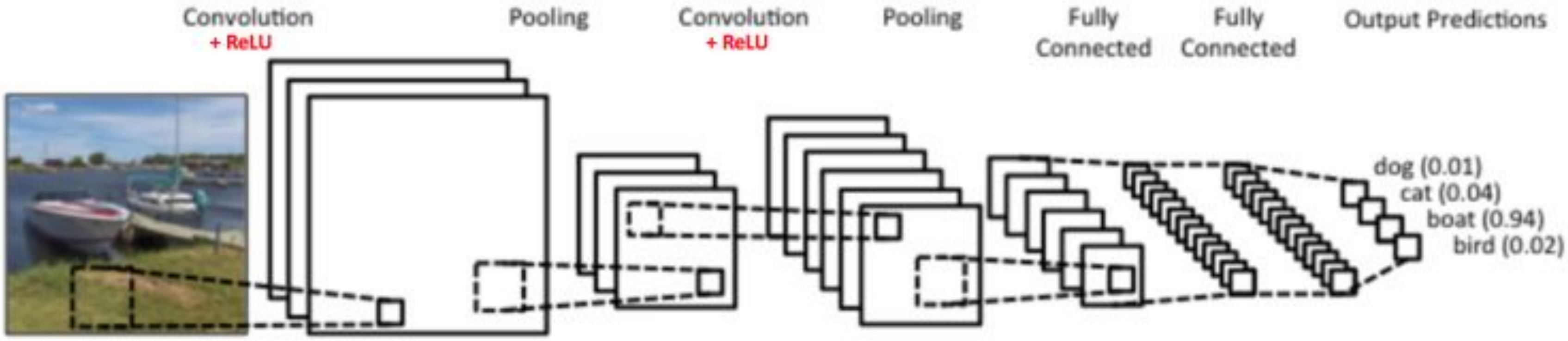
A PROPOSAL FOR THE DARTMOUTH SUMMER RESEARCH PROJECT ON ARTIFICIAL INTELLIGENCE

J. McCarthy, Dartmouth College
M. L. Minsky, Harvard University
N. Rochester, I.B.M. Corporation
C.E. Shannon, Bell Telephone Laboratories

August 31, 1955

We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.

Deep neural networks

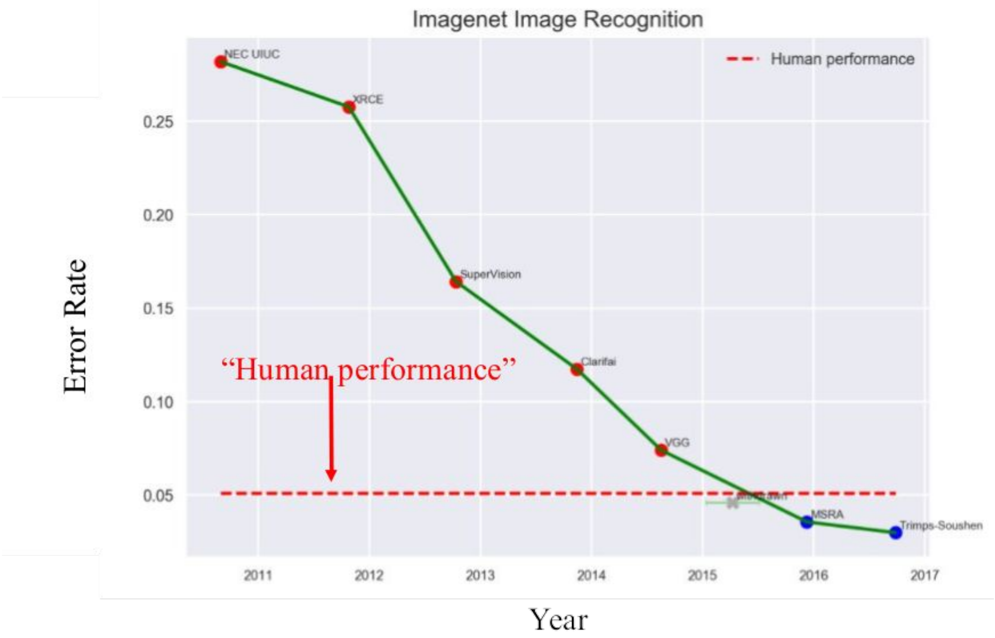


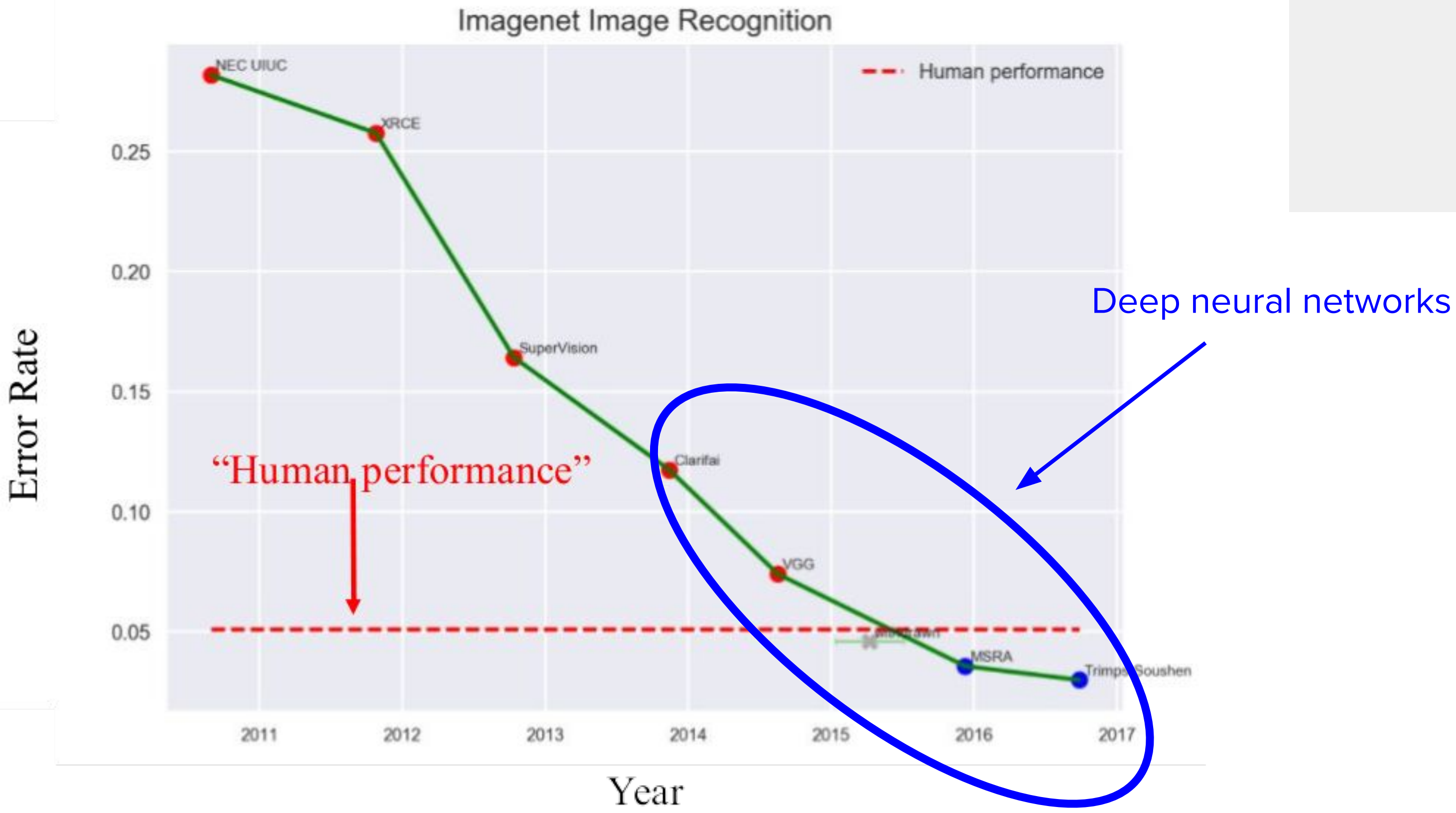
https://ujjwalkarn.me/2016/08/11/intuitive-explanation-convnets/

Train on 1.2 million human-labeled images, Test on 500K images

http://karpathy.github.io/assets/cnntsne.jpeg

ImageNet Object Recognition




Imagenet Image Recognition
Human performance
NEC UIUC
XRCE
SuperVision
Clarifai
VGG
MSRA
Trimps-Soushen
0.25
0.20
0.15
0.10
0.05
2011
2012
2013
2014
2015
2016
2017
Error Rate
Year
"Human performance"
Deep neural networks

What Are These Machines Learning?

What Are These Machines Learning?


“Animal”


“No Animal”

What Are These Machines Learning?

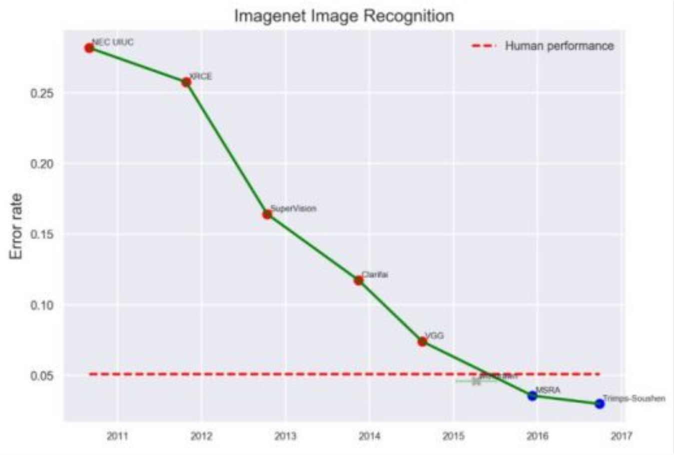


Fig. 1: Glimpse of the data collection process with the robotic platform (left) acquiring data of a cluttered scene populated with everyday objects.

What Are These Machines Learning?

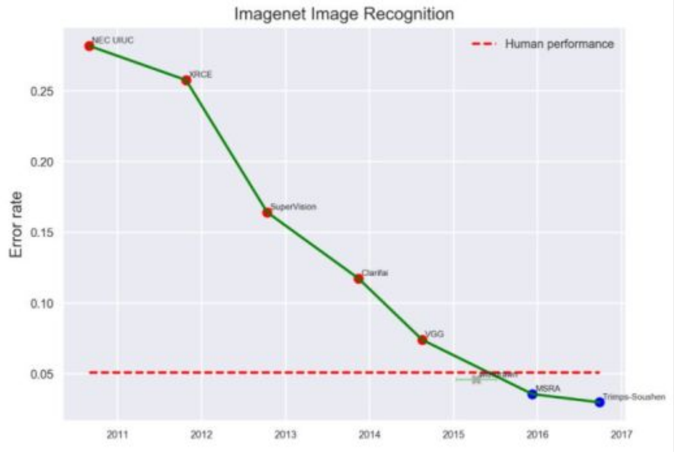


Fig. 1: Glimpse of the data collection process with the robotic platform (left) acquiring data of a cluttered scene populated with everyday objects.

Dataset		Network					Statistics	
Train on	Test on	CaffeNet	VGG-16	Inception-v2	ResNet-18	ResNet-50	Mean	Max
Web	Web	0.924	0.942	0.914	0.953	0.956	0.938	0.956
Web	Robot	0.268	0.297	0.282	0.282	0.388	0.303	0.388

Loghmani et al., 2017, “Recognizing Objects in the Wild: Where Do We Stand?”

What Are These Machines Learning?

Alcorn, Michael A., et al. "Strike (with) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects." arXiv preprint arXiv:1811.11553 (2018).

fire truck 0.99

What Are These Machines Learning?

Alcorn, Michael A., et al. "Strike (with) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects." arXiv preprint arXiv:1811.11553 (2018).

What Are These Machines Learning?

Alcorn, Michael A., et al. "Strike (with) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects." arXiv preprint arXiv:1811.11553 (2018).

What Are These Machines Learning?

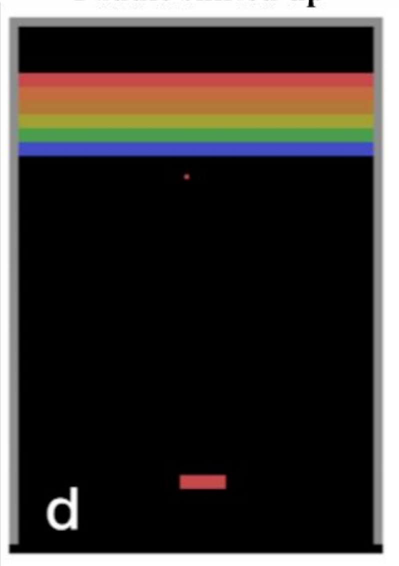
Alcorn, Michael A., et al. "Strike (with) a Pose: Neural Networks Are Easily Fooled by Strange Poses of Familiar Objects." arXiv preprint arXiv:1811.11553 (2018).

fire truck 0.99 school bus 0.98 fireboat 0.98 bobsled 0.79

What Are These Machines Learning?

Standard Breakout

Breakout with
Paddle shifted up



Kansky, K. et al., 2017. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. arXiv preprint arXiv:1706.04317.

What Are These Machines Learning?

“Speed Limit 80”

5’ 0°

5’ 15°

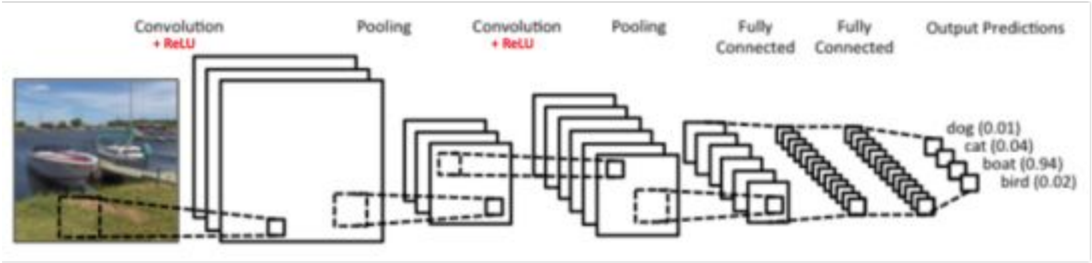
10’ 0°

10’ 30°

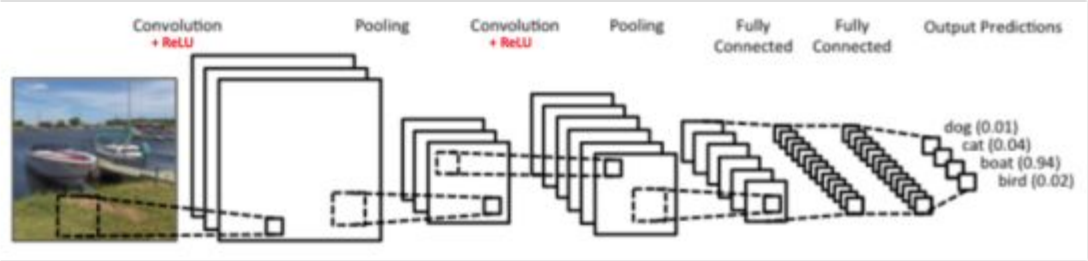
40’ 0°

Evtimov et al., “Robust Physical-World Attacks on Deep Learning Models”, 2017

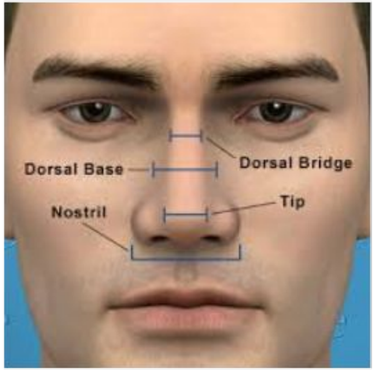
Perceptual Categories” versus **Concepts**

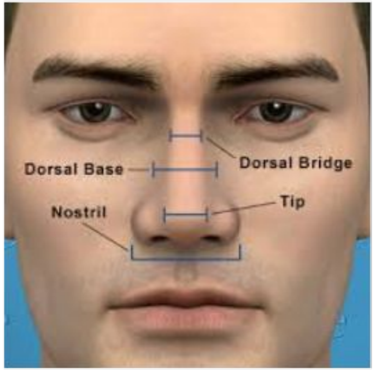


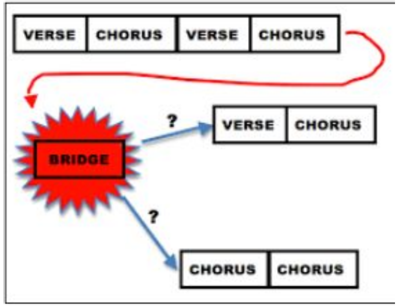
Perceptual Categories" versus **Concepts**

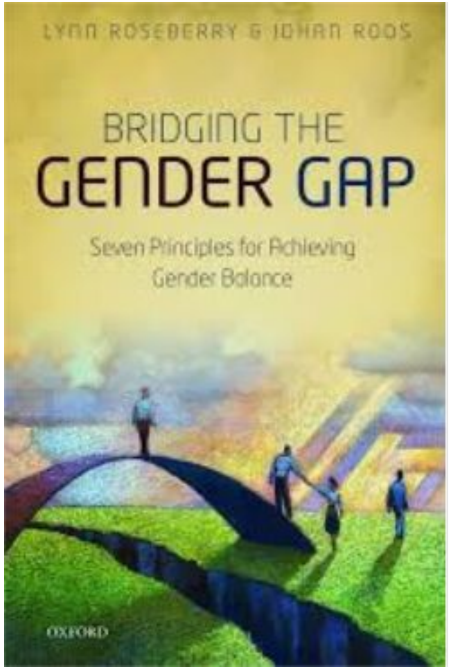


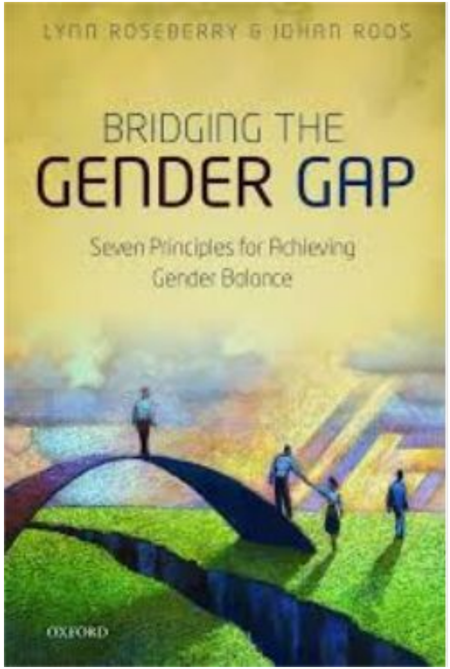
“Bridge”


Dorsal Base
Dorsal Bridge
Tip
Nostril

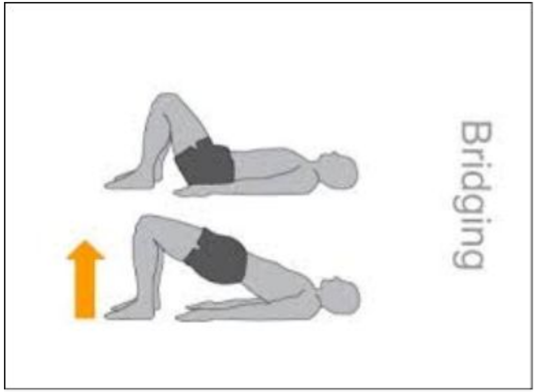

Dorsal Base
Dorsal Bridge
Nostril
Tip


VERSE
CHORUS
VERSE
CHORUS
BRIDGE
?
VERSE
CHORUS
?
CHORUS
CHORUS

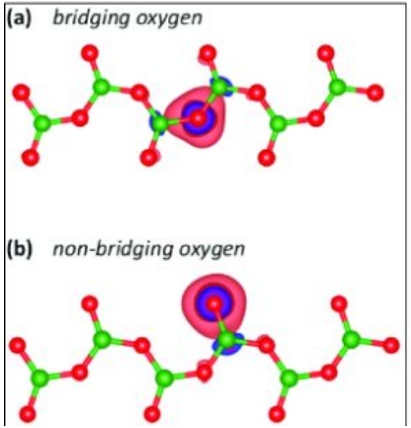

LYNN ROSEBERRY & JOHAN ROOS
BRIDGING THE
GENDER GAP
Seven Principles for Achieving
Gender Balance
OXFORD

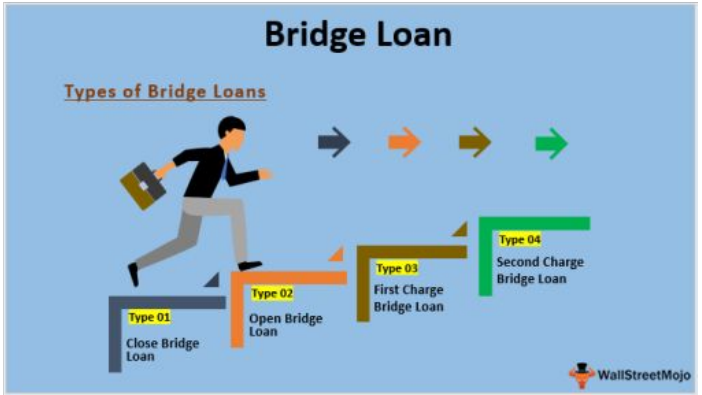


Biden says he's a 'bridge' to new 'generation of leaders' while campaigning with Harris, Booker, Whitmer



“Don’t burn your bridges”





“A concept is a package of analogies.”

—D. Hofstadter, Analogy as the Core of Cognition

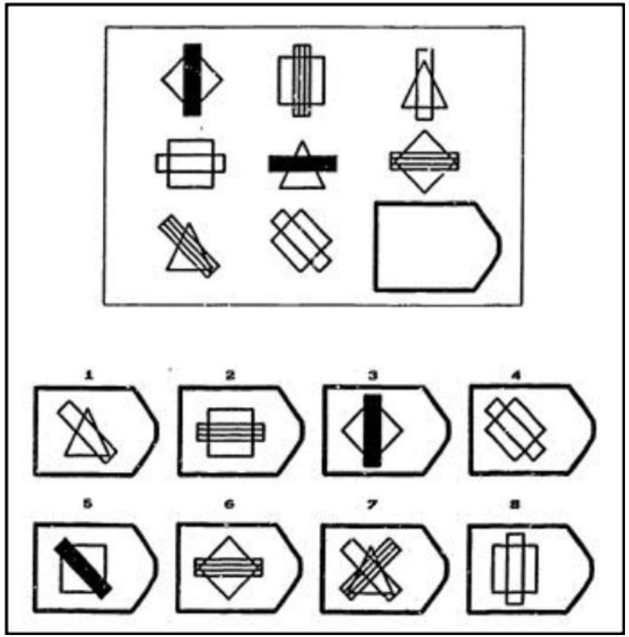
How can we get machines to learn concepts (rather than perceptual categories) and make analogies?

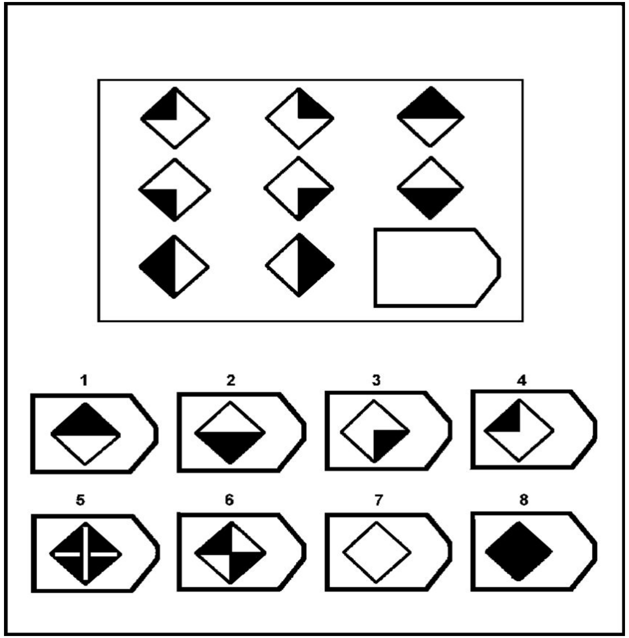
This Talk

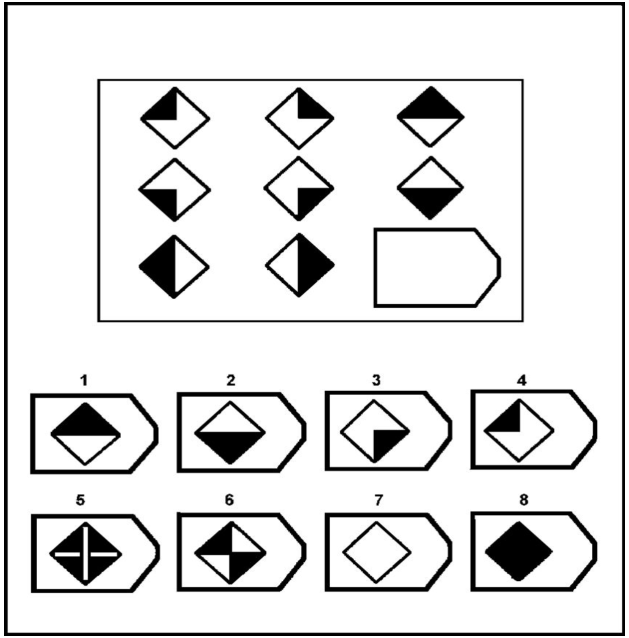
1. Survey of selected AI methods for abstraction and analogy
    - Deep learning
    - Probabilistic program induction
    - Copycat architecture
2. Open questions on how to make progress in this area

Deep Learning Approaches

Raven’s Progressive Matrices



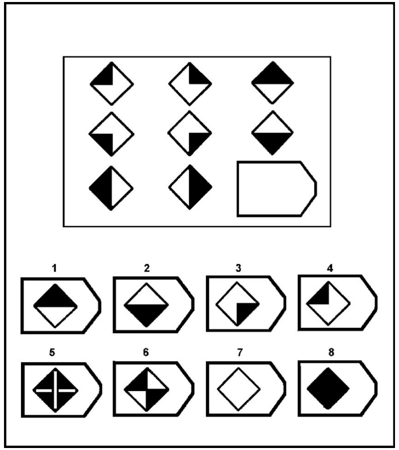

1
2
3
4
5
6
7
8



“highly correlated with human intelligence.”

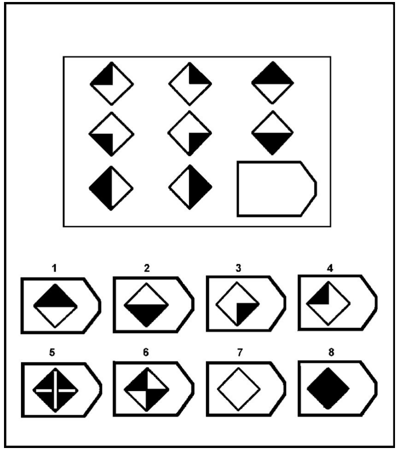
Deep learning approaches

Zhou et al, 2020, “Solving Raven's Progressive Matrices with Neural Networks”



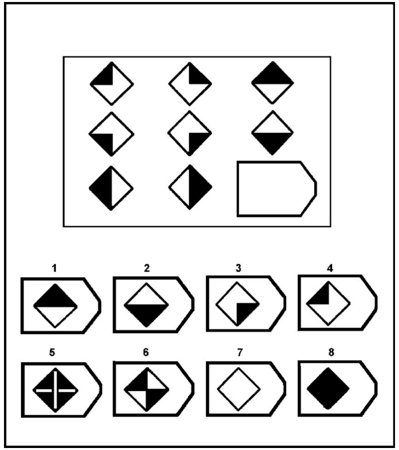
Deep learning approaches

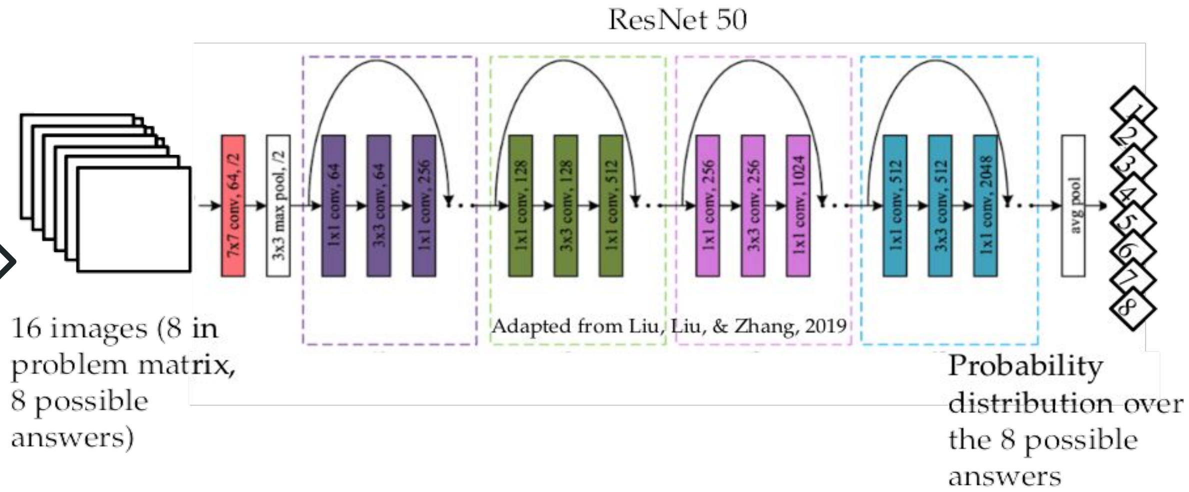
Zhou et al, 2020, “Solving Raven's Progressive Matrices with Neural Networks”



Deep learning approaches

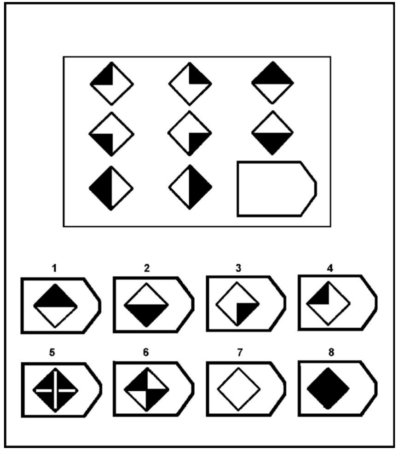
Zhou et al, 2020, “Solving Raven's Progressive Matrices with Neural Networks”

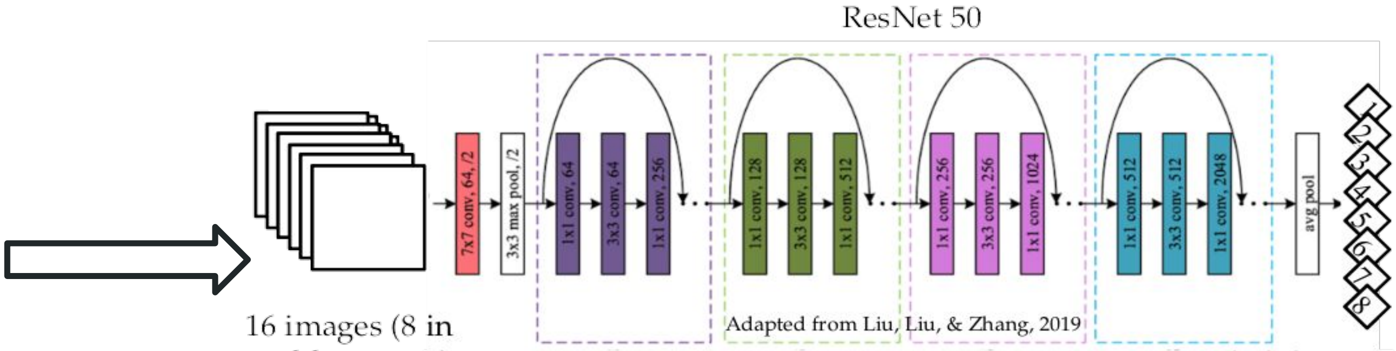




Deep learning approaches

Zhou et al, 2020, “Solving Raven's Progressive Matrices with Neural Networks”

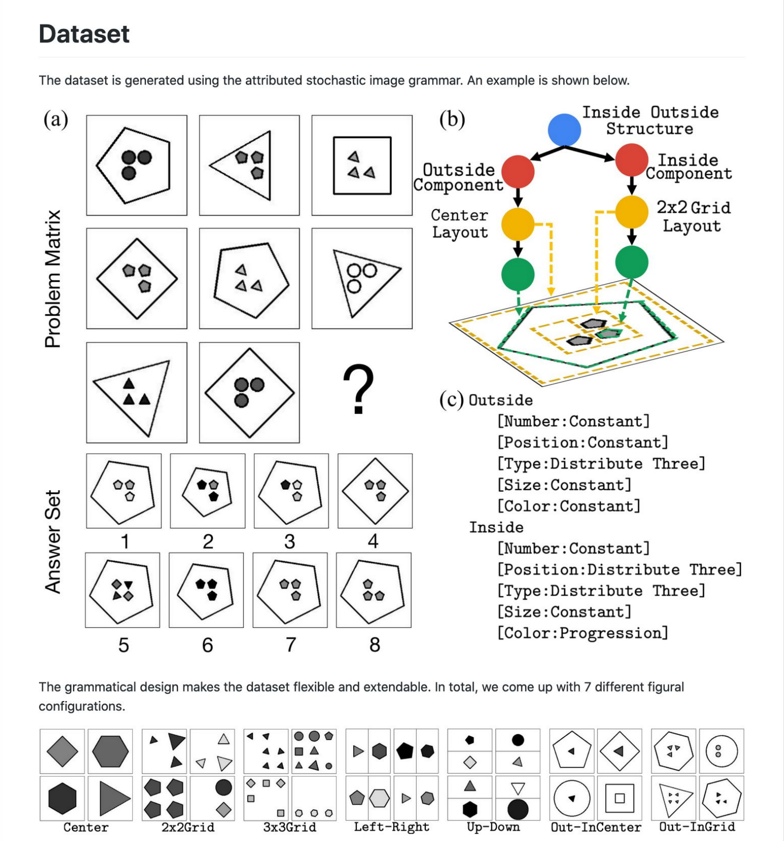




16 images (8 in problem matrix, 8 possible answers)

Probability distribution over the 8 possible answers

42,000 training examples (problems)
14,000 test examples



RAVENS dataset is generated using a stochastic image grammar

https://github.com/WellyZhang/RAVEN

Results:

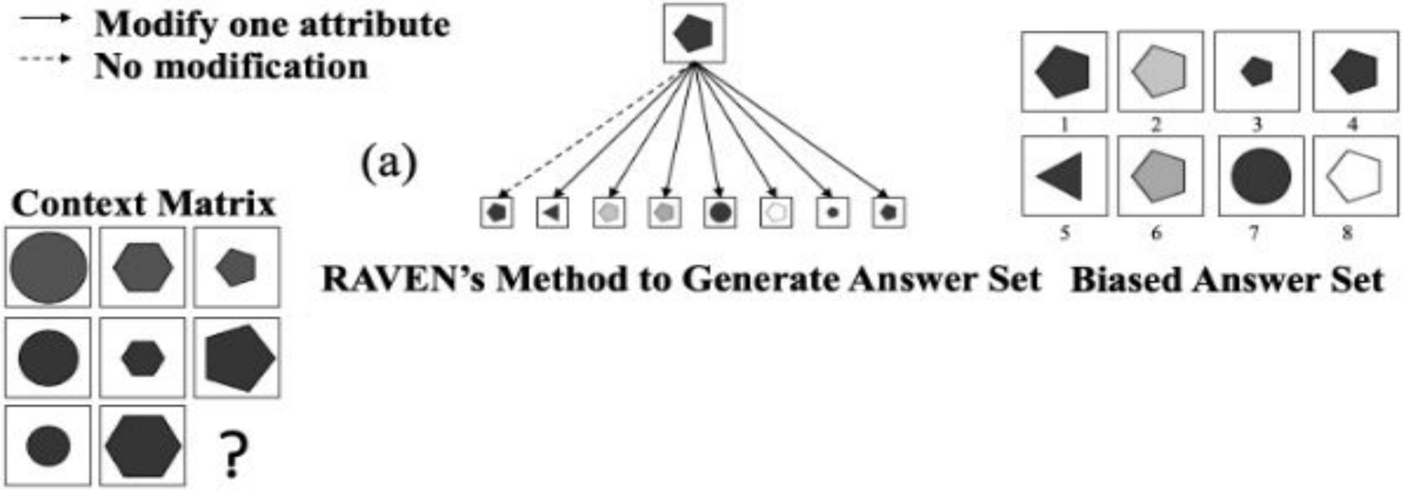
Zhou et al, 2020, “Solving Raven's Progressive Matrices with Neural Networks”

Table 2. Testing accuracy of different models in supervised manner. *Avg* denotes the average accuracy of each model.

Method	*Avg*	*Center*	*2*2Grid*	*3*3Grid*	*L-R*	*U-D*	*O-IC*	*O-IG*
LSTM	13.07	13.19	14.13	13.69	12.84	12.35	12.15	12.99
WReN	14.69	13.09	28.62	28.27	7.49	6.34	8.38	10.56
CNN	36.97	33.58	30.30	33.53	39.43	41.26	43.20	37.54
ResNet-18+MLP	53.43	52.82	41.86	44.29	58.77	60.16	63.19	53.12
LSTM+DRT	13.96	14.29	15.08	14.09	13.79	13.24	13.99	13.29
WReN+DRT	15.02	15.38	23.26	29.51	6.99	8.43	8.93	12.35
CNN+DRT	39.42	37.30	30.06	34.57	45.49	45.54	45.93	37.54
ResNet-18+MLP+DRT	59.56	58.08	46.53	50.40	65.82	67.11	69.09	60.11
RseNet-18 (ours, w/o pre-train)	77.18	72.75	57.00	62.65	91.00	89.60	88.40	78.85
RseNet-50 (ours, w pre-train)	86.26	89.45	66.60	67.95	97.85	98.15	96.60	87.20
CoPINet	**91.42**	**95.05**	**77.45**	**78.85**	**99.10**	**99.65**	**98.50**	**91.35**
Human	84.41	95.45	81.82	79.55	86.36	81.81	86.36	81.81

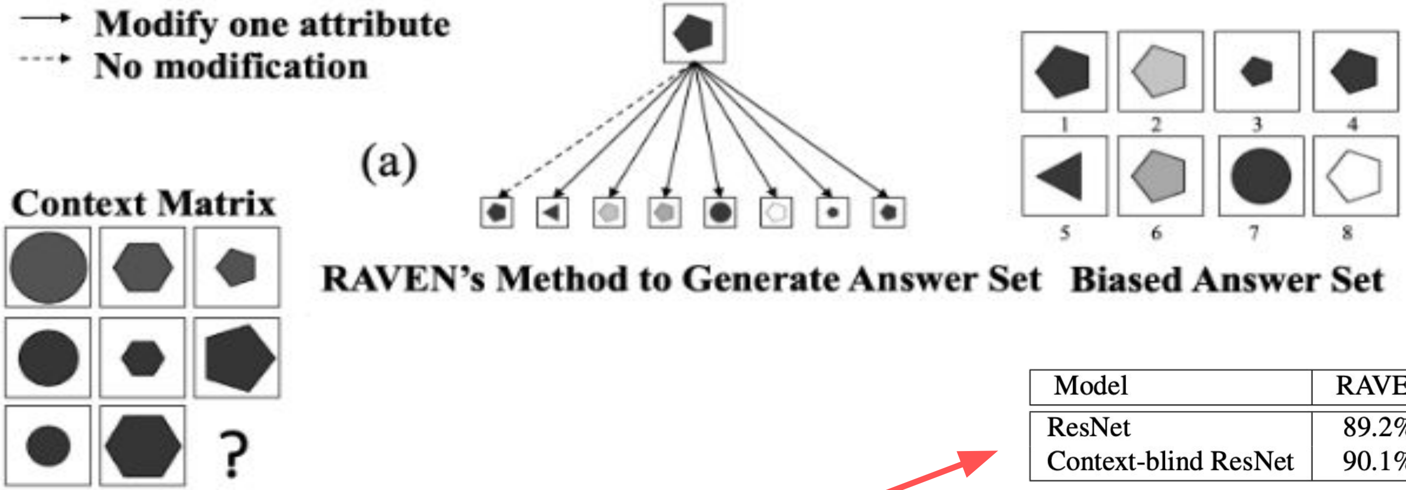
Bias in RAVENS dataset

(Hu et al, Hierarchical rule induction network for abstract visual reasoning)



Bias in RAVENS dataset

(Hu et al, Hierarchical rule induction network for abstract visual reasoning)



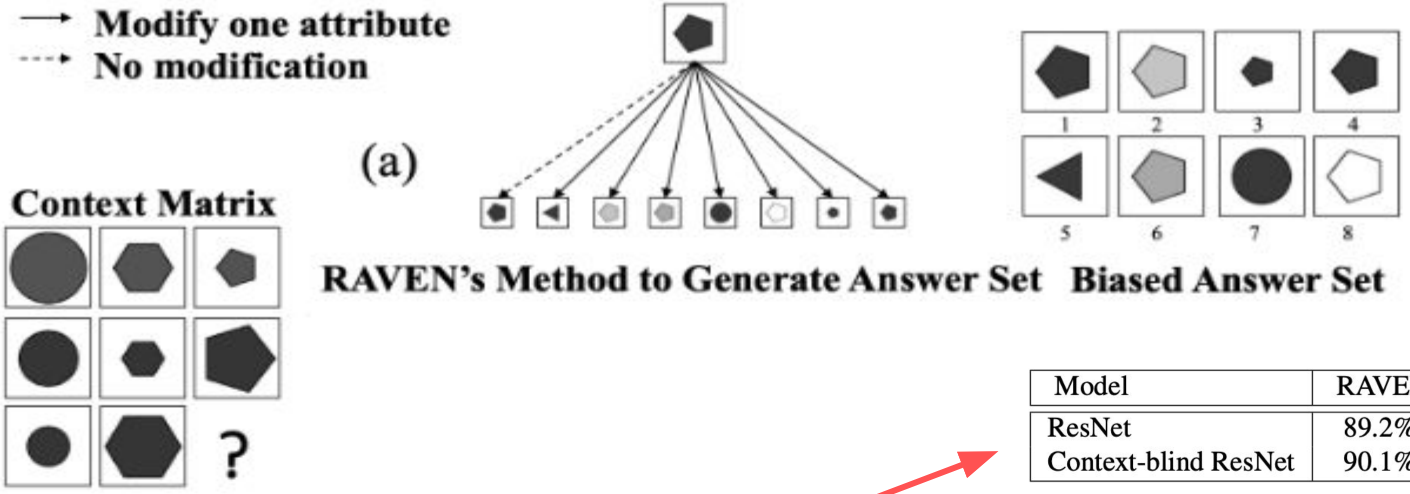
Model	RAVEN	Balanced-RAVEN
ResNet	89.2%	40.3%
Context-blind ResNet	90.1%	12.5%

Table 1. Test on RAVEN and Balanced-RAVEN.

Train on candidate answers only!

Bias in RAVENS dataset

(Hu et al, Hierarchical rule induction network for abstract visual reasoning)



Model	RAVEN	Balanced-RAVEN
ResNet	89.2%	40.3%
Context-blind ResNet	90.1%	12.5%

Table 1. Test on RAVEN and Balanced-RAVEN.

Train on candidate answers only!

Many types of deep learning approaches for Ravens-like problems

- **Wild Relation Network**, Barrett et al. 2018
- **Disentangled Feature Representations**, Steenbrugge et al. 2018
- **Attention Relation Network**, Hahne et al. 2019
- **Contrastive Perceptual Inference Network**, Zhang et al, 2019
- **Logic Embedding Network**, Zheng et al., 2019
- **Multi-Layer Relation Network**, Jahrens & Martinetz, 2020
- **Hierarchical Rule Induction Network**, Hu et al., 2020

Deep Learning Approaches: **Limitations**

Deep Learning Approaches: Limitations

- Requires very large corpus of training examples. Need to generate automatically. Makes NNs susceptible to shortcuts.

Deep Learning Approaches: Limitations

- Requires very large corpus of training examples. Need to generate automatically. Makes NNs susceptible to shortcuts.
- Trained networks are not transparent. What did they learn?

Deep Learning Approaches: Limitations

- Requires very large corpus of training examples. Need to generate automatically. Makes NNs susceptible to shortcuts.
- Trained networks are not transparent. What did they learn?
- If goal is general humanlike abstraction abilities, it doesn't make sense to have to train on tens of thousands of examples.

Deep Learning Approaches: Limitations

- Requires very large corpus of training examples. Need to generate automatically. Makes NNs susceptible to shortcuts.
- Trained networks are not transparent. What did they learn?
- If goal is general humanlike abstraction abilities, it doesn't make sense to have to train on tens of thousands of examples.

The essence of abstraction and analogy is few-shot learning!

Probabilistic Program Induction Approaches

Probabilistic Program Induction

Basic idea:

Probabilistic Program Induction

Basic idea:

- Define a “domain specific language”: Primitives and grammar for programs that learn to generate data

Probabilistic Program Induction

Basic idea:

- Define a "domain specific language": Primitives and grammar for programs that learn to generate data
- Given an input *I*, want to find program *s* (in the domain specific language) that would generate *I*:

$$s = argmax_{s' \in S} P(s' | I)$$

  Where *S* is the space of all programs

Probabilistic Program Induction

Basic idea:

- Define a “domain specific language”: Primitives and grammar for programs that learn to generate data
- Given an input *I*, want to find program *s* (in the domain specific language) that would generate *I*:

  $$s = argmax_{s' \in S} P(s' | I)$$

  Where *S* is the space of all programs
- Define a prior probability *P()* and a likelihood over possible programs *P(I |)*

Probabilistic Program Induction

Basic idea:

- Define a “domain specific language”: Primitives and grammar for programs that learn to generate data
- Given an input *I*, want to find program *s* (in the domain specific language) that would generate *I*:

$$s = argmax_{s' \in S} P(s'|I)$$

  Where *S* is the space of all programs
- Define a prior probability *P()* and a likelihood over possible programs *P(I |)*
- By Bayes’ rule,

$$s = argmax_{s' \in S} P(s'|I) = argmax_{s' \in S} P(I|s')P(s')$$

Probabilistic Program Induction

Basic idea:

- Define a "domain specific language": Primitives and grammar for programs that learn to generate data
- Given an input *I*, want to find program *s* (in the domain specific language) that would generate *I*:

$$s = argmax_{s' \in S} P(s'|I)$$

  Where *S* is the space of all programs
- Define a prior probability *P()* and a likelihood over possible programs *P(I |)*
- By Bayes' rule,

$$s = argmax_{s' \in S} P(s'|I) = argmax_{s' \in S} P(I|s')P(s')$$

- Sample programs according to this distribution until an acceptable one is found

The Omniglot Challenge

Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015

The Omniglot Challenge

Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015

Omniglot: Dataset of multiple examples of 1623 handwritten characters from 50 writing systems (including pen strokes)

The Omniglot Challenge

Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015

Omniglot: Dataset of multiple examples of 1623 handwritten characters from 50 writing systems (including pen strokes)

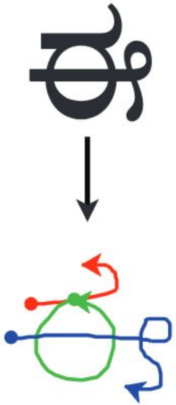
Omniglot Tasks

One-shot classification

Omniglot Tasks

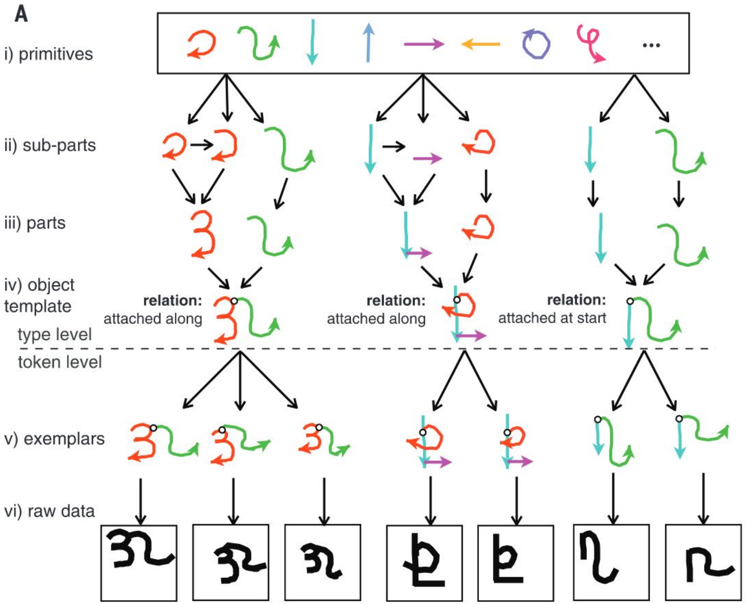
One-shot classification

One-shot generation



Domain-specific “language”

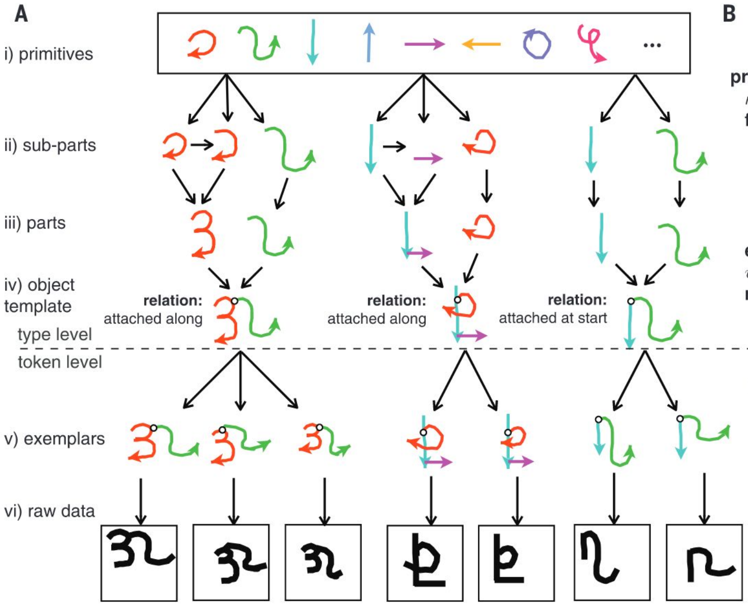
From Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015



Domain-specific “language”

From Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015

Prior probability distributions are learned from data:



B

procedure GENERATETYPE
  $\kappa \leftarrow P(\kappa)$ ▷ Sample number of parts
  for i = 1 ... κ **do**
    $n_i \leftarrow P(n_i|\kappa)$ ▷ Sample number of sub-parts
    for j = 1 ... n_i **do**
      $s_{ij} \leftarrow P(s_{ij}|s_{i(j-1)})$ ▷ Sample sub-part sequence
    end for
    $R_i \leftarrow P(R_i|S_1, ..., S_{i-1})$ ▷ Sample relation
  end for
  $\psi \leftarrow \{\kappa, R, S\}$
  return @GENERATETOKEN(ψ) ▷ Return program

procedure GENERATETOKEN(ψ)
  for $i = 1...\kappa$ **do**
    $S_i^{(m)} \leftarrow P(S_i^{(m)}|S_i)$ ▷ Add motor variance
    $L_i^{(m)} \leftarrow P(L_i^{(m)}|R_i, T_1^{(m)}, ..., T_{i-1}^{(m)})$ ▷ Sample part's start location
    $T_i^{(m)} \leftarrow f(L_i^{(m)}, S_i^{(m)})$ ▷ Compose a part's trajectory
  end for
  $A^{(m)} \leftarrow P(A^{(m)})$ ▷ Sample affine transform
  $I^{(m)} \leftarrow P(I^{(m)}|T^{(m)}, A^{(m)})$ ▷ Sample image
  return $I^{(m)}$

Omniglot Tasks

One-shot classification

Solving Task

Classification:

Let I^t be the example, where t is the character class. Let I^c be the candidates, for $c=1,\ldots,20$.

Omniglot Tasks

One-shot classification

Solving Task

Classification:

Let I^t be the example, where t is the character class. Let I^c be the candidates, for c=1,...,20.

Approximate $P(I^c \mid I^t)$ for each I^c, and choose I^c with the maximum conditional probability.

$$P(I^c|I^t) \propto P(I^t|I^c)P(I^c)$$

Omniglot Tasks

One-shot classification

Solving Task

Classification:

Let I^t be the example, where t is the character class. Let I^c be the candidates, for c=1,...,20.

Approximate $P(I^c \mid I^t)$ for each I^c, and choose I^c with the maximum conditional probability.

$$P(I^c|I^t) \propto \underbrace{P(I^t|I^c)}_{\text{Likelihood}}\underbrace{P(I^c)}_{\text{Prior}}$$

Omniglot Tasks

One-shot classification

Solving Task

Classification:

Let I^t be the example, where t is the character class. Let I^c be the candidates, for c=1,...,20.

Approximate $P(I^c \mid I^t)$ for each I^c, and choose I^c with the maximum conditional probability.

$$P(I^c|I^t) \propto P(I^t|I^c)P(I^c)$$

Likelihood Prior

Approximated
using heuristic search to generate a program to represent I^c

Omniglot Tasks

One-shot classification

Solving Task

Classification:

Let I^t be the example, where t is the character class. Let I^c be the candidates, for c=1,...,20.

Approximate P(I^c | I^t) for each I^c, and choose I^c with the maximum conditional probability.

$$P(I^c|I^t) \propto P(I^t|I^c)P(I^c)$$

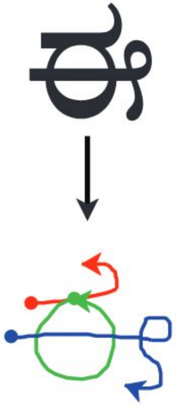
Likelihood Prior

Approximated by attempts to "refit'" the program representing I^t to the program representing I^c

Approximated using heuristic search to generate a program to represent I^c

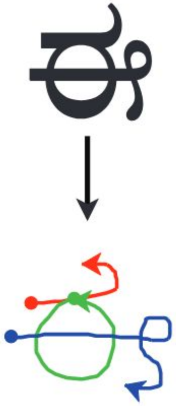
Omniglot Tasks

One-shot generation



Omniglot Tasks

One-shot generation



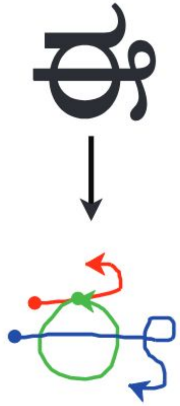
Solving Task

Generation:

Let I^c be the example, where c is the character class.

Omniglot Tasks

One-shot generation



Solving Task

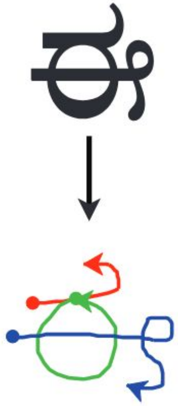
Generation:

Let I^c be the example, where c is the character class.

Find a program to generate I^c.

Omniglot Tasks

One-shot generation



Solving Task

Generation:

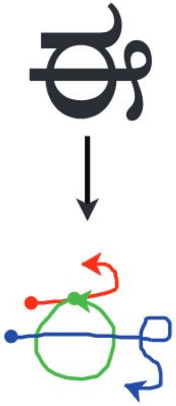
Let I^c be the example, where c is the character class.

Find a program to generate I^c.

Run that program to generate a new example.

Omniglot Tasks

One-shot generation



Solving Task

Generation:

Let I^c be the example, where c is the character class.

Find a program to generate I^c.

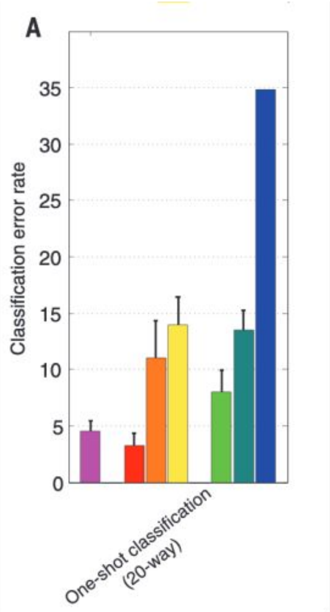
Run that program to generate a new example.

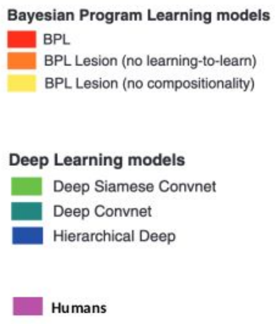
Use human judges to compare characters generated by program with those generated by people under same one-shot conditions.

Results

Results

One-shot classification   One-shot generation

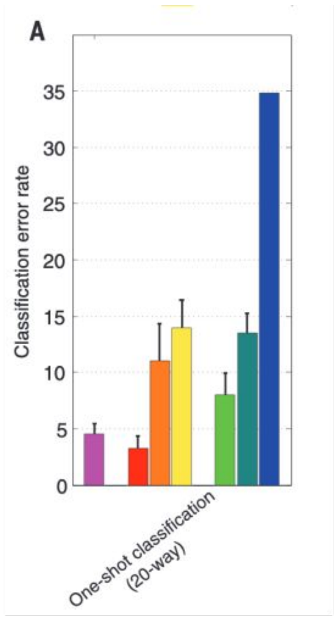


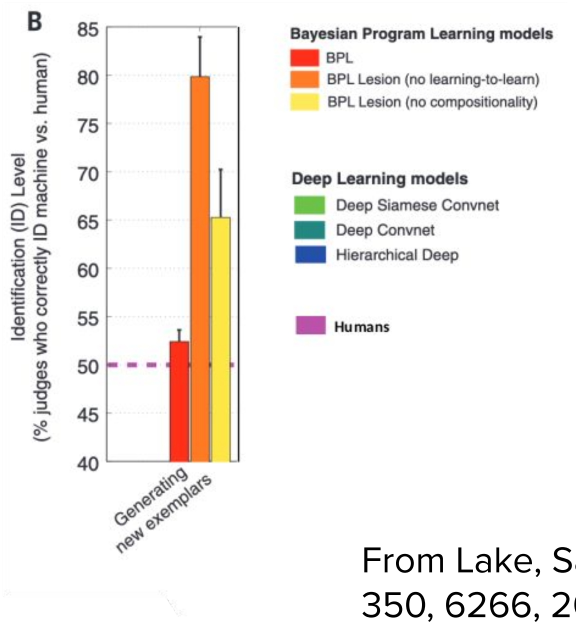


From Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015

Results

One-shot classification   One-shot generation





From Lake, Salakhutdinov, & Tenenbaum, Science, 350, 6266, 2015

Limitations of Probabilistic Program Induction Approaches

Limitations of Probabilistic Program Induction Approaches

- Can require a lot of built-in primitives!

Limitations of Probabilistic Program Induction Approaches

- Can require a lot of built-in primitives!
- Need enough data to learn prior probability and likelihood distributions.

Limitations of Probabilistic Program Induction Approaches

- Can require a lot of built-in primitives!
- Need enough data to learn prior probability and likelihood distributions.
- Can require a lot of search

Limitations of Probabilistic Program Induction Approaches

- Can require a lot of built-in primitives!
- Need enough data to learn prior probability and likelihood distributions.
- Can require a lot of search
- Hard to scale to more complex problems – as domain-specific language becomes more expressive, search becomes less tractable

Limitations of Probabilistic Program Induction Approaches

- Can require a lot of built-in primitives!
- Need enough data to learn prior probability and likelihood distributions.
- Can require a lot of search
- Hard to scale to more complex problems – as domain-specific language becomes more expressive, search becomes less tractable

Interesting extensions: Hybrid symbolic-neuro systems (e.g., Ellis et al, "Dreamcoder", 2020; Feinman & Lake, "Generating new concepts with hybrid neuro-symbolic models", 2021).

Copycat Architecture

Hofstadter & Mitchell, 1995

Letter-String Analogies

(Hofstadter and Mitchell, 1995)

abc → abd

pqrs → ?

Letter-String Analogies

(Hofstadter and Mitchell, 1995)

abc → abd

pqrs → ?

abc → abd

ppqqrrss → ?

Letter-String Analogies

(Hofstadter and Mitchell, 1995)

abc → abd

pqrs → ?

abc → abd

ppqqrrss → ?

abcd → dcba

srqp → ?

Letter-String Analogies

(Hofstadter and Mitchell, 1995)

abc → abd

pqrs → ?

abc → abd

ppqqrrss → ?

abcd → dcba

srqp → ?

axxd → abcd

xqxx → ?

Letter-String Analogies

(Hofstadter and Mitchell, 1995)

abc → abd

pqrs → ?

abc → abd

ppqqrrss → ?

abcd → dcba

srqp → ?

axxd → abcd

xqxx → ?

- Idealized "situations", with objects, relations, groups, actions, events

Letter-String Analogies

(Hofstadter and Mitchell, 1995)

abc → abd

pqrs → ?

abc → abd

ppqqrrss → ?

abcd → dcba

srqp → ?

axxd → abcd

xqxx → ?

- Idealized "situations", with objects, relations, groups, actions, events
- Meant to be a tool for exploring general issues of abstraction and analogy-making

Inspiration from neuroscience/psychophysics

Inspiration from neuroscience/psychophysics

- **Treisman et al., 1980:**
  Temporal dynamics of perception: shift between parallel, random, “pre-attentive” bottom-up processing and more deterministic, focused, serial, “attentive” top-down processing.

Inspiration from neuroscience/psychophysics

- **Treisman et al., 1980:**
  Temporal dynamics of perception: shift between parallel, random, "pre-attentive" bottom-up processing and more deterministic, focused, serial, "attentive" top-down processing.

- **Gilbert & Sigman (2007):**
  "V1 and V2 may work as 'active blackboards' that integrate and sustain the result of computations performed in higher areas."

Inspiration from neuroscience/psychophysics

- **Treisman et al., 1980:**
  Temporal dynamics of perception: shift between parallel, random, "pre-attentive" bottom-up processing and more deterministic, focused, serial, "attentive" top-down processing.

- **Gilbert & Sigman (2007):**
  "V1 and V2 may work as 'active blackboards' that integrate and sustain the result of computations performed in higher areas."

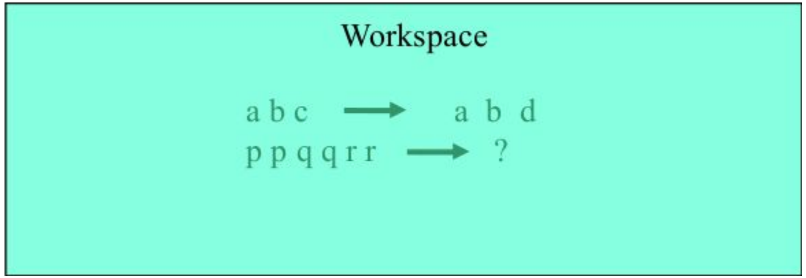
- **Kahneman, Triesman, and Gibbs (1992):**
  Notion of "object files": temporary and modifiable perceptual structures, created on the fly in working memory, which interact with longer-term memory.

Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)

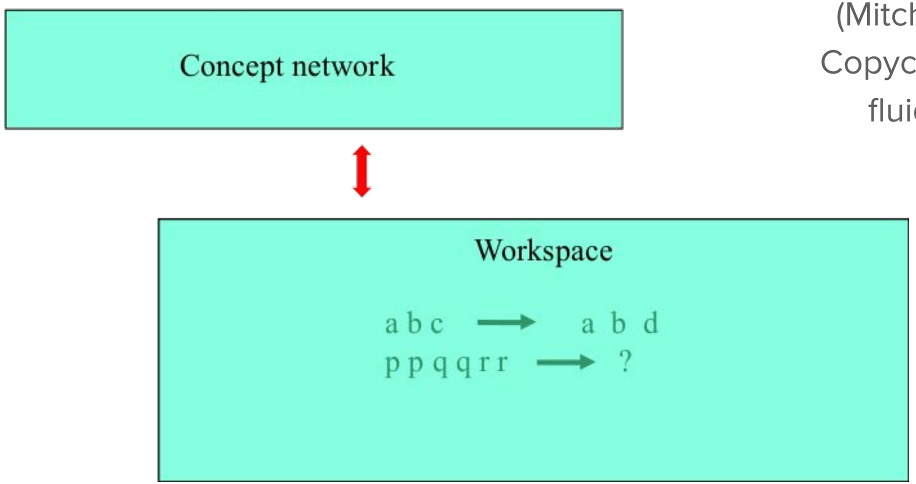
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



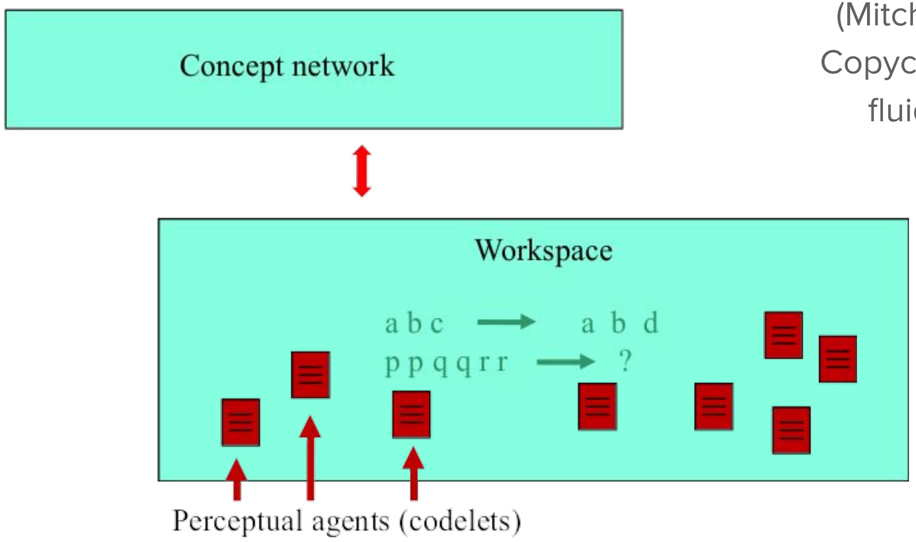
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



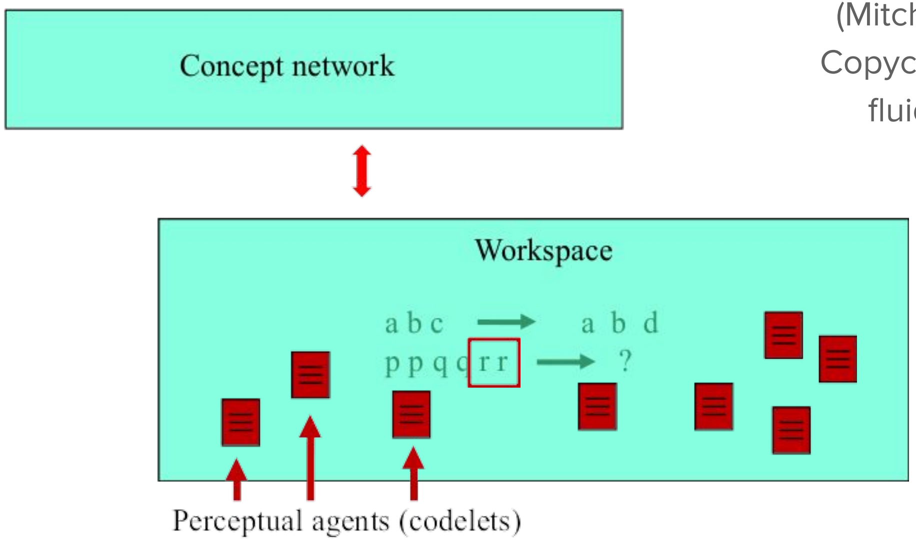
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



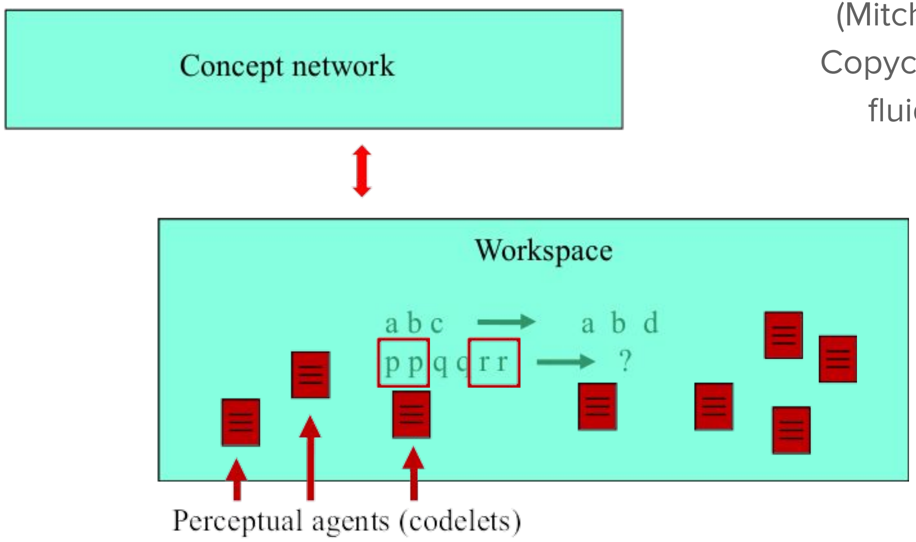
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



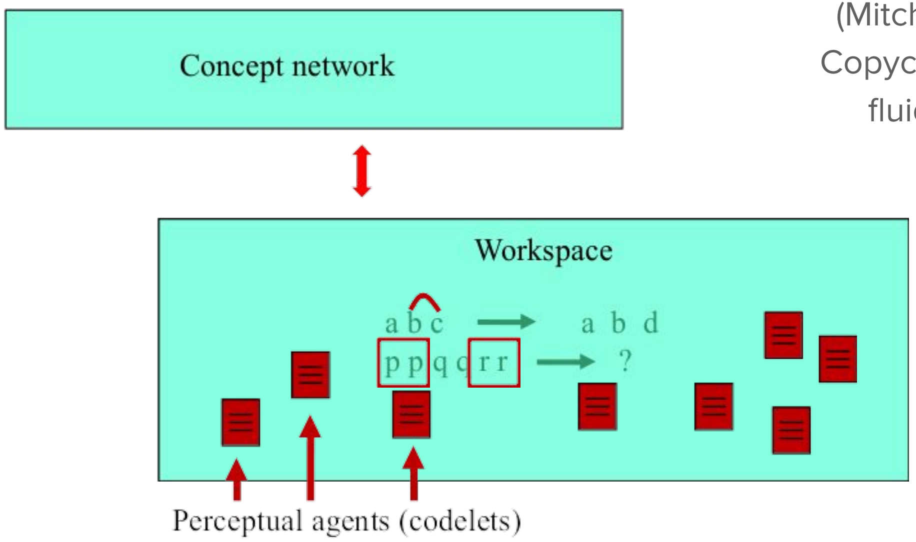
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



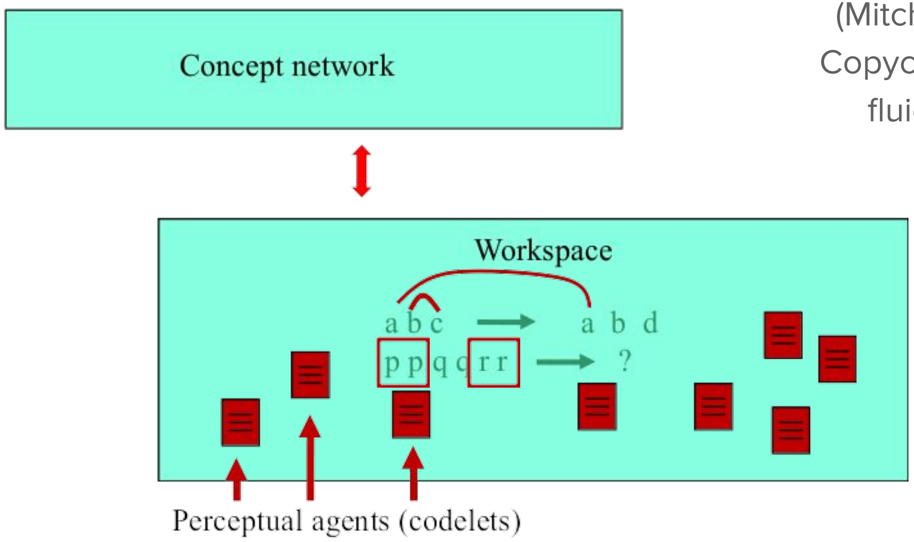
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



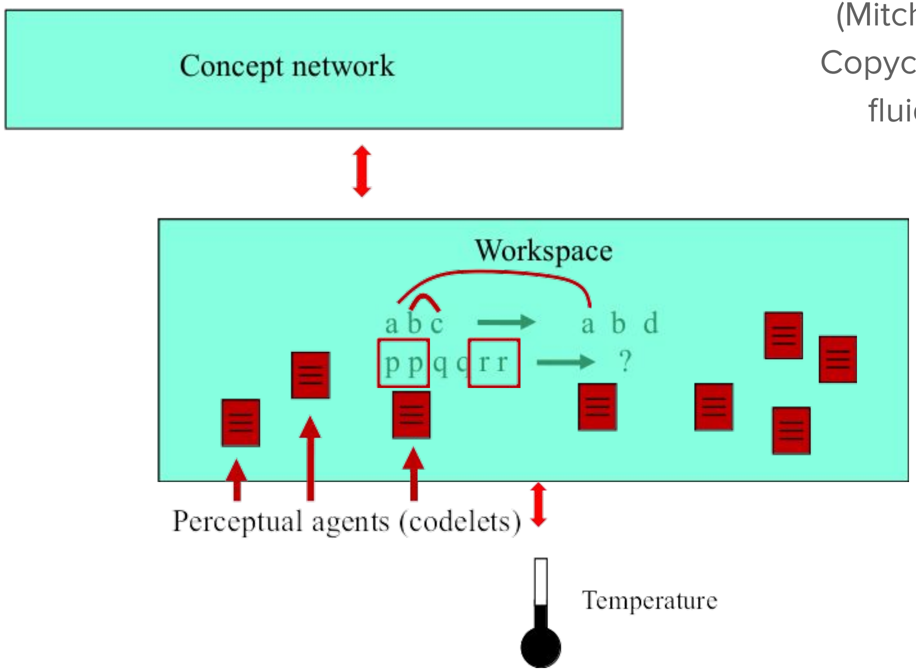
Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)

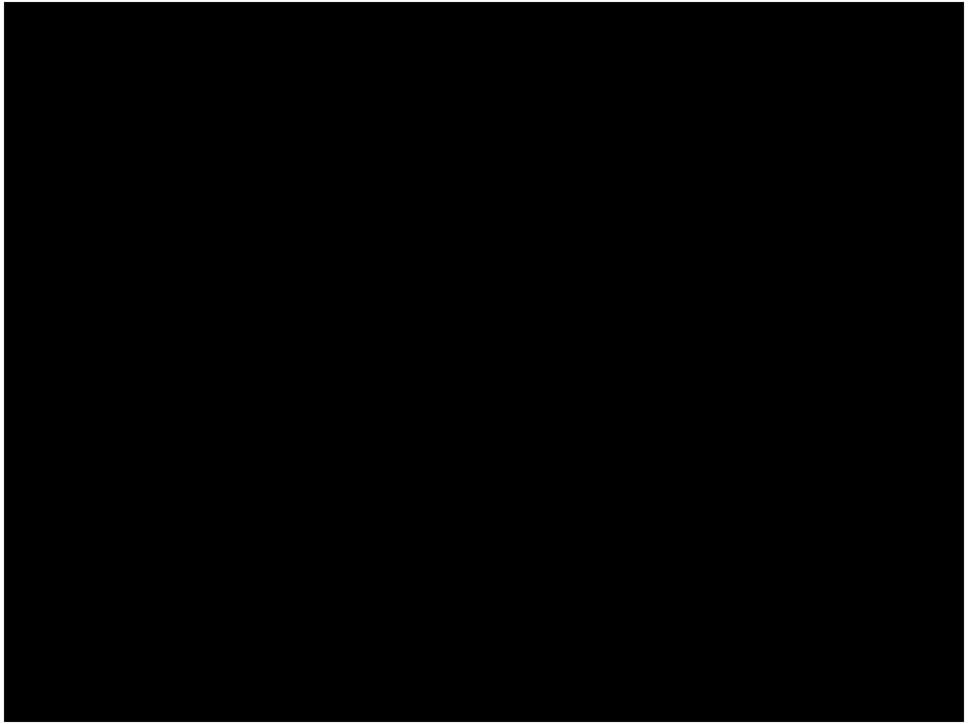


Copycat Architecture

(Mitchell & Hofstadter, 1995, “The Copycat project: A model of mental fluidity and analogy-making”)



Copycat (Metacat) demo



Some important ideas from Copycat

Modeling analogy-making—and other “high-level” cognitive processes—as perception, where a representation is actively built up over time

Some important ideas from Copycat

Modeling analogy-making—and other “high-level” cognitive processes—as perception, where a representation is actively built up over time

Perception unfolds dynamically, continually integrating symbolic/subsymbolic and top down/bottom-up processes

Some important ideas from Copycat

Modeling analogy-making—and other “high-level” cognitive processes—as perception, where a representation is actively built up over time

Perception unfolds dynamically, continually integrating symbolic/subsymbolic and top down/bottom-up processes

Continual integration of prior knowledge with bottom-up perceptions and perceived context

Some important ideas from Copycat

Modeling analogy-making—and other "high-level" cognitive processes—as perception, where a representation is actively built up over time

Perception unfolds dynamically, continually integrating symbolic/subsymbolic and top down/bottom-up processes

Continual integration of prior knowledge with bottom-up perceptions and perceived context

Emergent transition from bottom-up, parallel, random to top-down, serial, deterministic, attentive modes of processing

Copycat: Limitations

Copycat: Limitations

- Copycat's architecture is too ad hoc

Copycat: Limitations

- Copycat’s architecture is too ad hoc

- Not clear how general the architecture is

Copycat: Limitations

- Copycat's architecture is too ad hoc
- Not clear how general the architecture is
- How to form new concepts beyond what is given in its prior conceptual repertoire?

Abstraction and Reasoning Corpus (ARC)

Chollet, 2019

On the Measure of Intelligence

François Chollet *
Google, Inc.
fchollet@google.com

November 5, 2019

Abstract

To make deliberate progress towards more intelligent and more human-like artificial systems, we need to be following an appropriate feedback signal: we need to be able to define and evaluate intelligence in a way that enables comparisons between two systems, as well as comparisons with humans. Over the past hundred years, there has been an abundance of attempts to define and measure intelligence, across both the fields of psychology and AI. We summarize and critically assess these definitions and evaluation approaches,

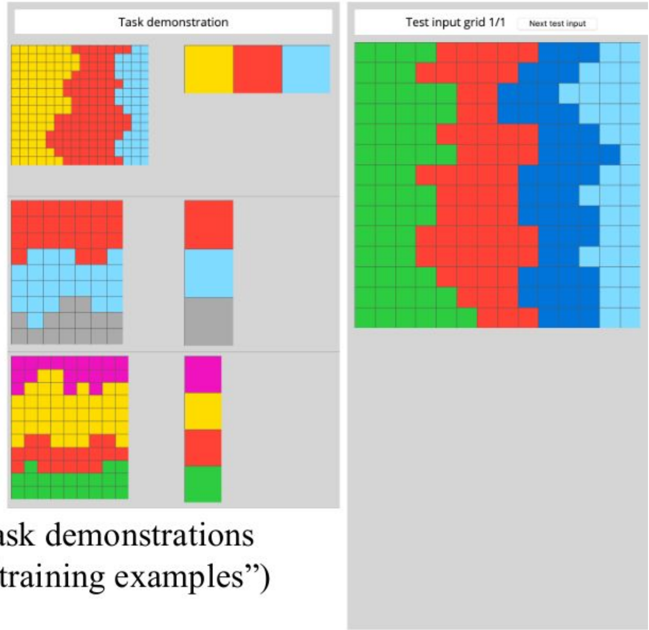
Core Knowledge Systems

(Spelke, 2000)

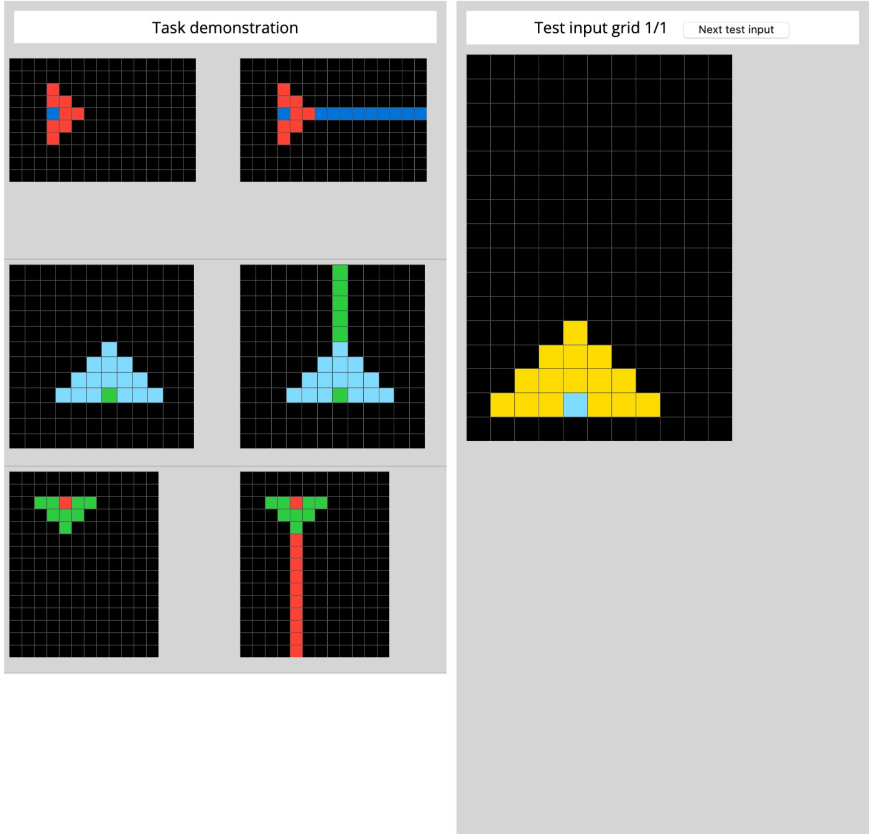
- Objects
- Space & geometry
- Numbers & numerosity
- Agents & actions

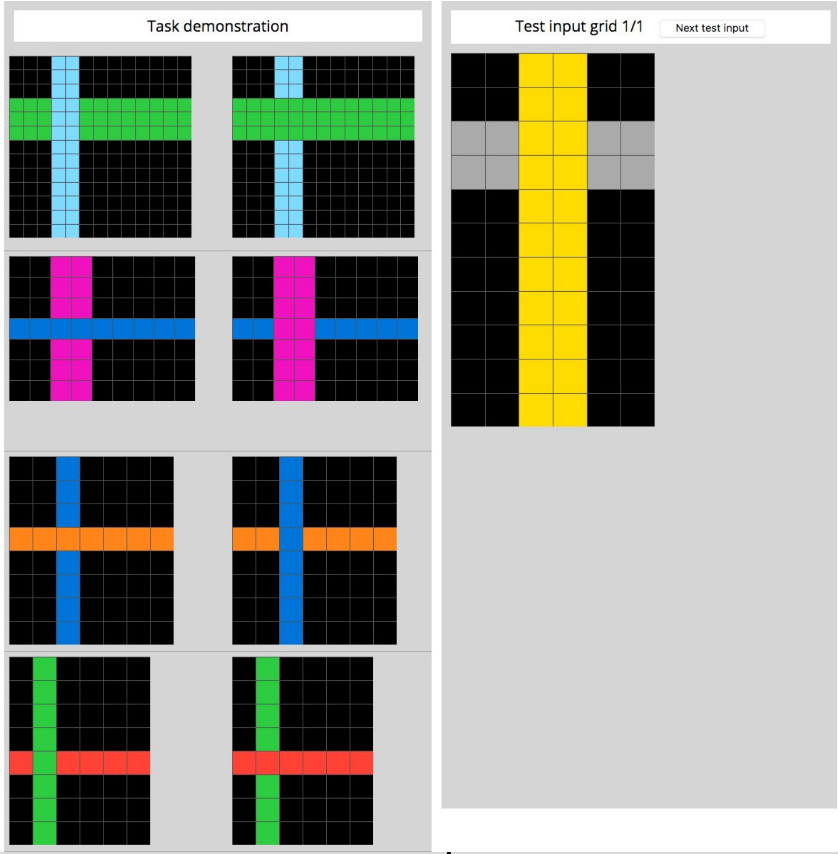
Abstraction and Reasoning Corpus (ARC)

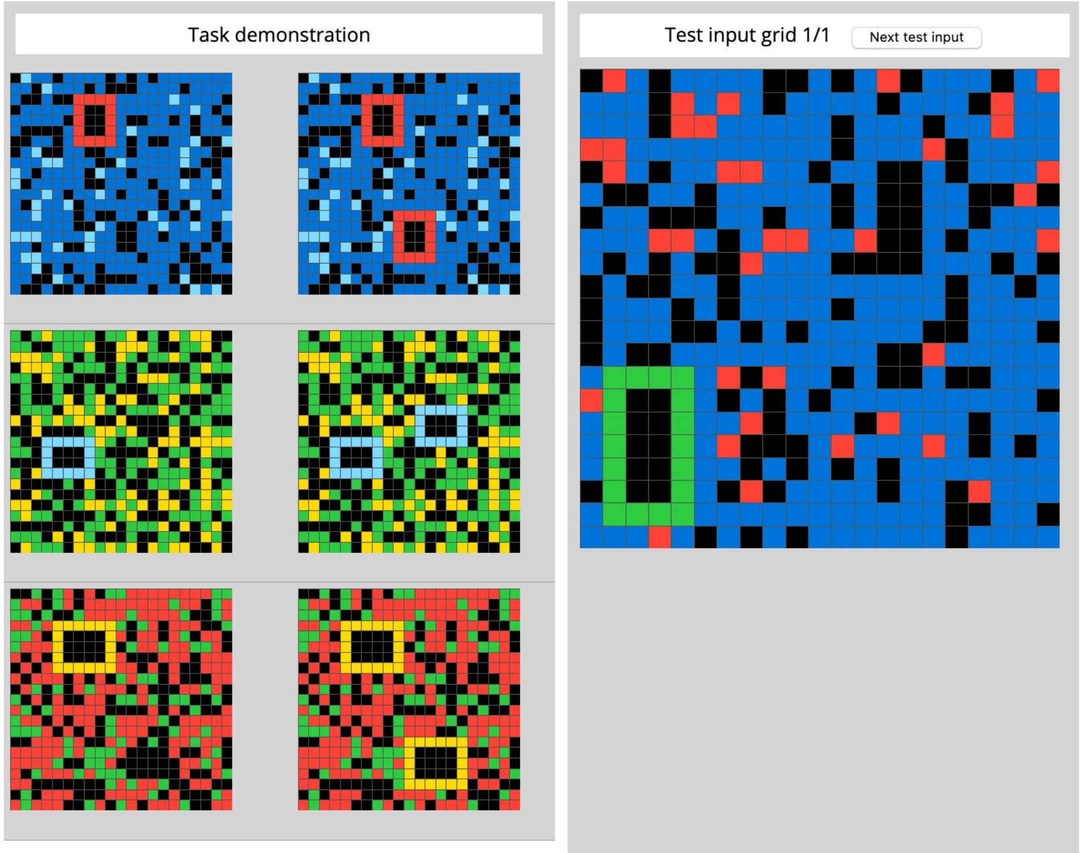
Chollet, 2019

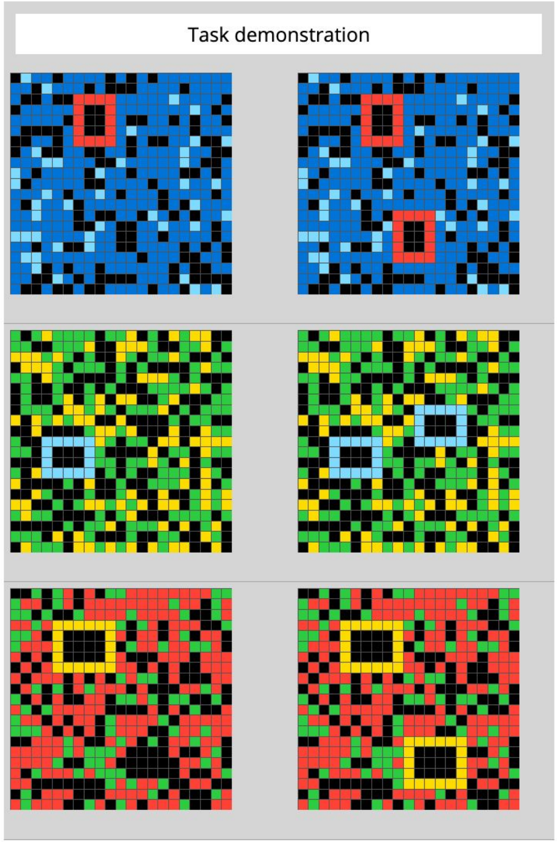


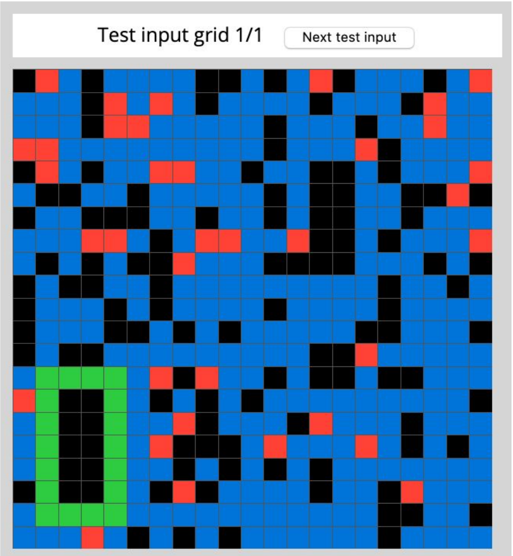
Task demonstrations
("training examples")


Task demonstration
Test input grid 1/1
Next test input


Task demonstration
Test input grid 1/1
Next test input

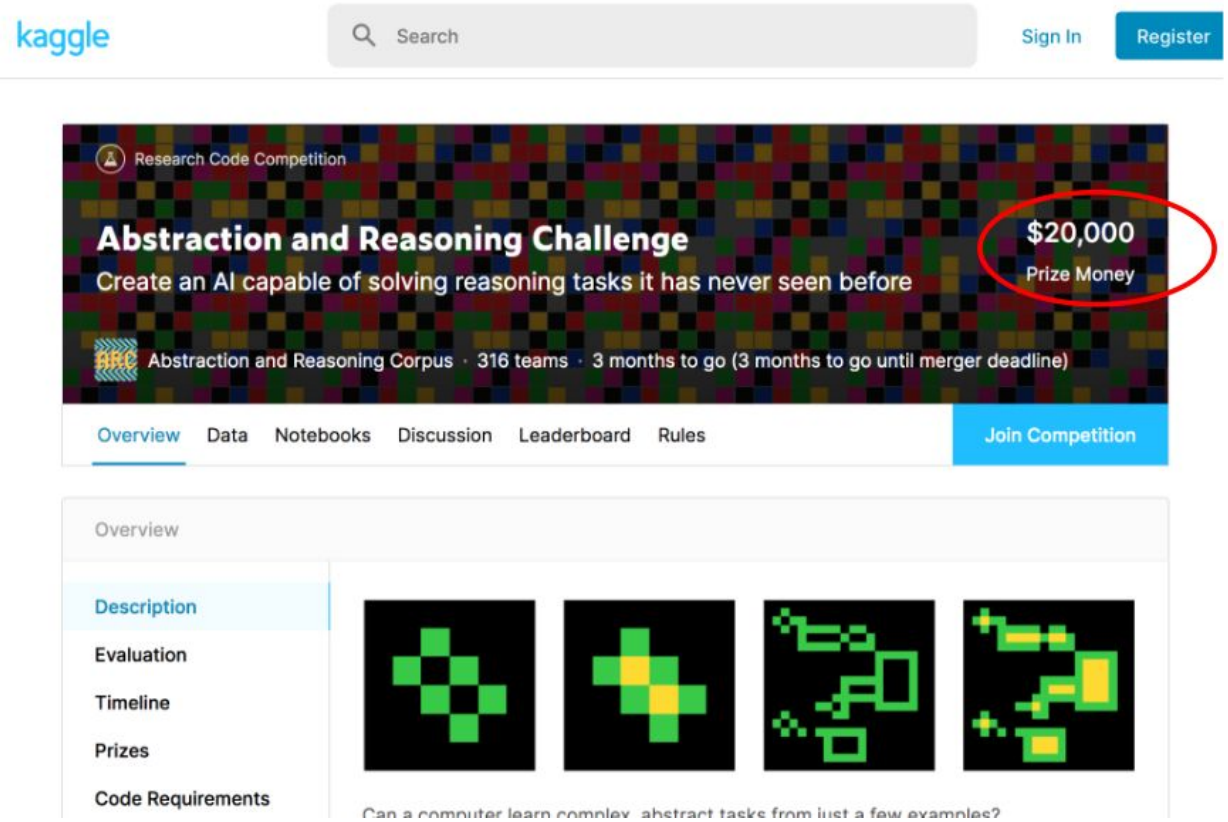

Task demonstration
Test input grid 1/1
Next test input





Training set: 400 "tasks".

Hidden evaluation set: 600 "tasks".


kaggle
Search
Sign In
Register
Research Code Competition
Abstraction and Reasoning Challenge
Create an AI capable of solving reasoning tasks it has never seen before
$20,000
Prize Money
ARC
Abstraction and Reasoning Corpus · 316 teams · 3 months to go (3 months to go until merger deadline)
Overview
Data
Notebooks
Discussion
Leaderboard
Rules
Join Competition
Overview
Description
Evaluation
Timeline
Prizes
Code Requirements
Can a computer learn complex, abstract tasks from just a few examples?

Results

Best programs: ~30% of test cases solved (given three guesses).

But not very general.

How to make progress on abstraction and analogy in AI?

How to make progress on abstraction and analogy in AI?

- Need common suite of challenging tasks

How to make progress on abstraction and analogy in AI?

- Need common suite of challenging tasks
- Advantage of idealized domains:

How to make progress on abstraction and analogy in AI?

- Need common suite of challenging tasks
- Advantage of idealized domains:
  - We can be explicit about what prior knowledge and assumptions are needed for each task domain.

How to make progress on abstraction and analogy in AI?

- Need common suite of challenging tasks
- Advantage of idealized domains:
    - We can be explicit about what prior knowledge and assumptions are needed for each task domain.
    - By avoiding language-based tasks, we can avoid anthropomorphizing what a system has achieved.

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation "test sets").
- Evaluation should be based on several factors, including:

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).
- Evaluation should be based on several factors, including:
  - accuracy on hidden examples

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).
- Evaluation should be based on several factors, including:
    - accuracy on hidden examples
    - performance across different tasks (generality)

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).
- Evaluation should be based on several factors, including:
    - accuracy on hidden examples
    - performance across different tasks (generality)
    - ability to generate (as opposed to simply recognize) solutions

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).
- Evaluation should be based on several factors, including:
    - accuracy on hidden examples
    - performance across different tasks (generality)
    - ability to generate (as opposed to simply recognize) solutions
    - ability to abstract with only a few examples

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).
- Evaluation should be based on several factors, including:
    - accuracy on hidden examples
    - performance across different tasks (generality)
    - ability to generate (as opposed to simply recognize) solutions
    - ability to abstract with only a few examples
    - robustness to modifications in tasks

- AI methods should be evaluated on hidden human-created examples that periodically change (no static evaluation “test sets”).
- Evaluation should be based on several factors, including:
    - accuracy on hidden examples
    - performance across different tasks (generality)
    - ability to generate (as opposed to simply recognize) solutions
    - ability to abstract with only a few examples
    - robustness to modifications in tasks
    - scalability to more complex examples of tasks

Thank you for listening!