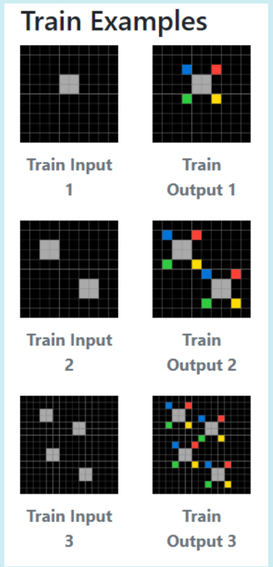
# Efficient learning through programatic representations

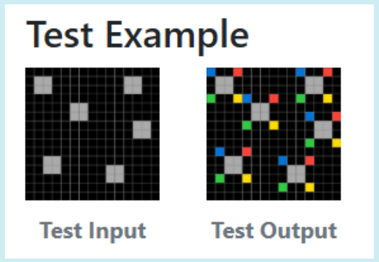
*How can we better find the matching objects from the inputs and outputs?*

ALEXANDRA-LAURA ZAGHĂR

*CSE3000 Research Project - Q4 2026*
*Responsible professor: Sebastijan Dumančić*
*Supervisor: Dekel Zak*

## 1. Background



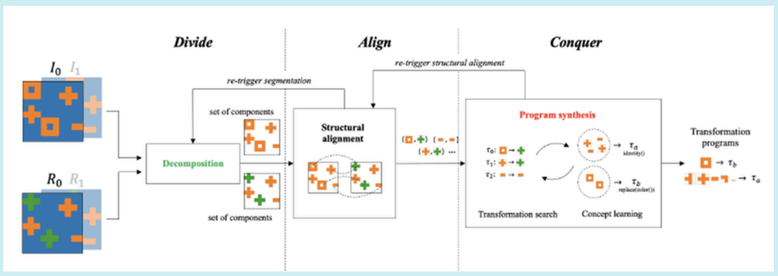




ARC benchmark - made to test AI on the capability to learn like humans, with very few training examples

An approach to solving this benchmark is with program synthesis - generating a program satisfying a given specification from a given grammar

A solution with good results using this approach and Divide, align and conquer: BEN

- divide: segment the images into objects
- align: match input and output objects
- conquer: find transformations for the matches and when to apply them



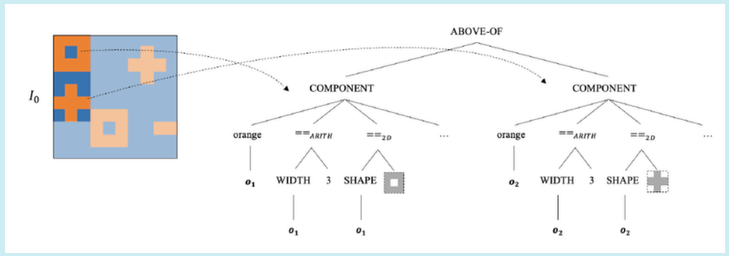

## 2. Research Question

How can we better find the matching objects from the inputs and outputs?

Sub-questions:

- How does the improved version compare to the initial one?
- What heuristics could be used to decide what features help the most?
- What other features could improve the algorithm?

## 3. How the align part in BEN works

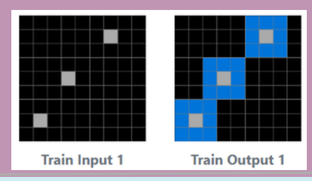



Each of the segmented objects is encoded based on its features using propositional encoding

The encoded objects are matched using a Structure Mapping Engine - create match hypotheses, get global mappings from these and then rank them.

## 4. What my changes are

- New features:
  - spatial relations: ranked versions of coordinates and mirroring
  - colour: similarity based on percentage of shared colours
- Weighting features: when some are less important or should be removed
  - individual features are less important when the number of objects is the same in input and output
  - coordinates should be removed when the girds have different sizes

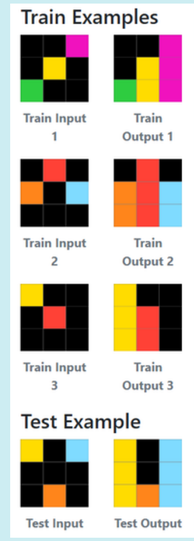





## 5. How testing is conducted

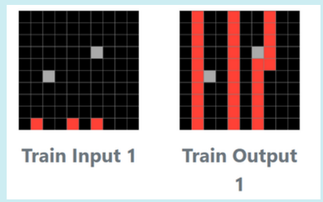
- Overall performance of BEN before and after the changes
- To test the Align part in isolation from the rest of the algorithm:
  - a sample of 25 examples that are relevant to the changes made were manually chosen and and encoded as input and output objects, as well as expected matches
  - a sample of 25 examples were randomly chosen to test how the the rest of the benchmark is affected

## 6. Results and discussion

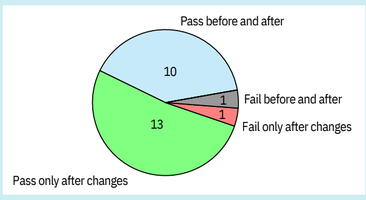



Additional solved task

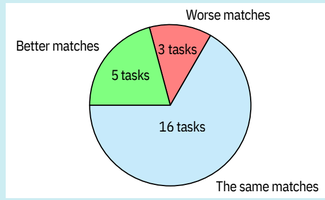
- With the entire algorithm: almost no changes in average time, 1 additional task solved and 2 no longer solved
- Less transformations are searched on average
- Overall better performance on the isolated tests. Even in a random sample, more tasks have improved matches than worse




Task with worse matches found after changes




Manually selected tasks




Randomly selected tasks

## 7. Conclusion and future work

The changes improve matching based on the tests in isolation, but highlight the diversity of the benchmark and some trade-offs.

Limitations:

- matching is tested in isolation on only a sample
- only one repetition of the experiments on the entire benchmark

Future work:

- richer spatial representations
- colour encoding
- matching multiple objects in the input with one in output
- add features to encode patterns or symmetry