Maximal Multi-layer Specification Synthesis

Yanju Chen
UCSB

Ruben Martins
CMU

Yu Feng
UCSB

FSE 2019
August 29, 2019

What is Program Synthesis?

$$\exists P.\ \forall x.\ \phi(x, P(x))$$

- Find a program P that for all inputs x meets the specification ϕ

Programming-by-Example

- Find a program P that satisfies all input-output examples ϕ

Programming-by-Example

Input

Student	Grade	Score1	Score2
Greg	A	75	76
Greg	B	86	85
Sally	A	85	86
Sally	B	80	78

Output

Student	B_Score1	B_Score2	A_Score1	A_Score2
Greg	86	85	75	76
Sally	80	78	85	86

- Can we find a program that automatically **transforms tables** given input-output examples?

Programming-by-Example

Input

Student	Grade	Score1	Score2
Greg	A	75	76
Greg	B	86	85
Sally	A	85	86
Sally	B	80	78

Output

Student	B_Score1	B_Score2	A_Score1	A_Score2
Greg	86	85	75	76
Sally	80	78	85	86

R program:

```
df1=gather(input,Score,Grade,Score1,Score2)
df2=unite(df1,AllScores,Time,Score)
output=spread(df2,AllScores,Grade)
```

- Component-based synthesis of **table** consolidation and **transformation** tasks from examples. PLDI 2017

Programming-by-Example

Examples ϕ **Program** P

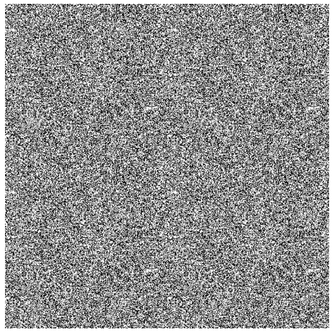
Examples are simple to use

Examples are imprecise

Capture the user intent with a combination of specifications:

- Natural language
- Input-output examples

Challenges



- Real-world textual information is **noisy** and **ambiguous**

Hybrid neural network architecture that combines LSTM-based sequence-to-sequence (**seq2seq**) with **apriori algorithm** for mining rules

- Unclear how to integrate this information with a synthesizer

Encode this information to maximum satisfiability modulo theory (**Max-SMT**) and solve it with an off-the-self SMT solver

Concrete Program

Input

Student	Grade	Score1	Score2
Greg	A	75	76
Greg	B	86	85
Sally	A	85	86
Sally	B	80	78

Output

Student	B_Score1	B_Score2	A_Score1	A_Score2
Greg	86	85	75	76
Sally	80	78	85	86

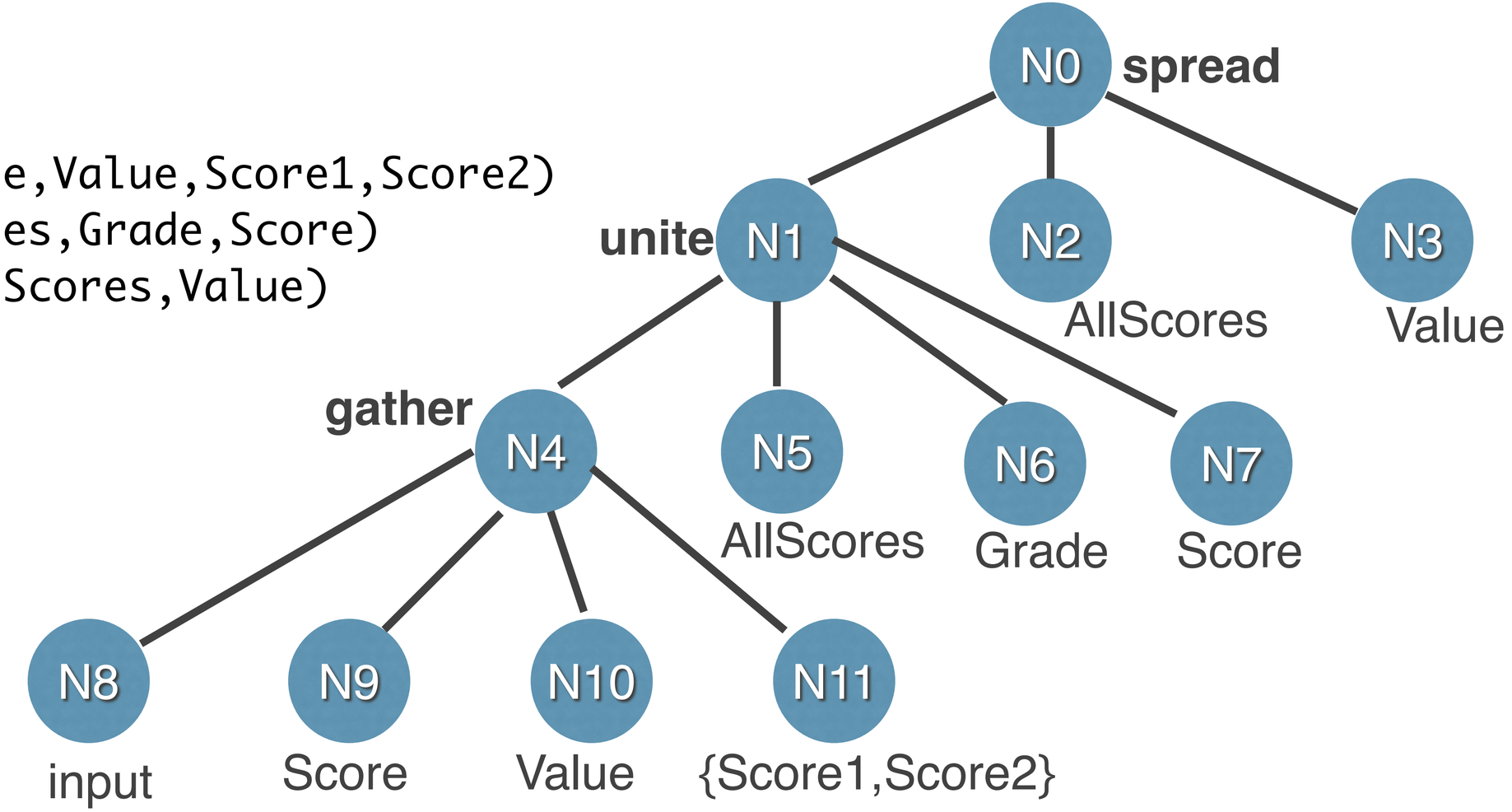
R program:

```
df1=gather(input,Score,Grade,Score1,Score2)
df2=unite(df1,AllScores,Time,Score)
output=spread(df2,AllScores,Grade)
```

Concrete Program

R program:

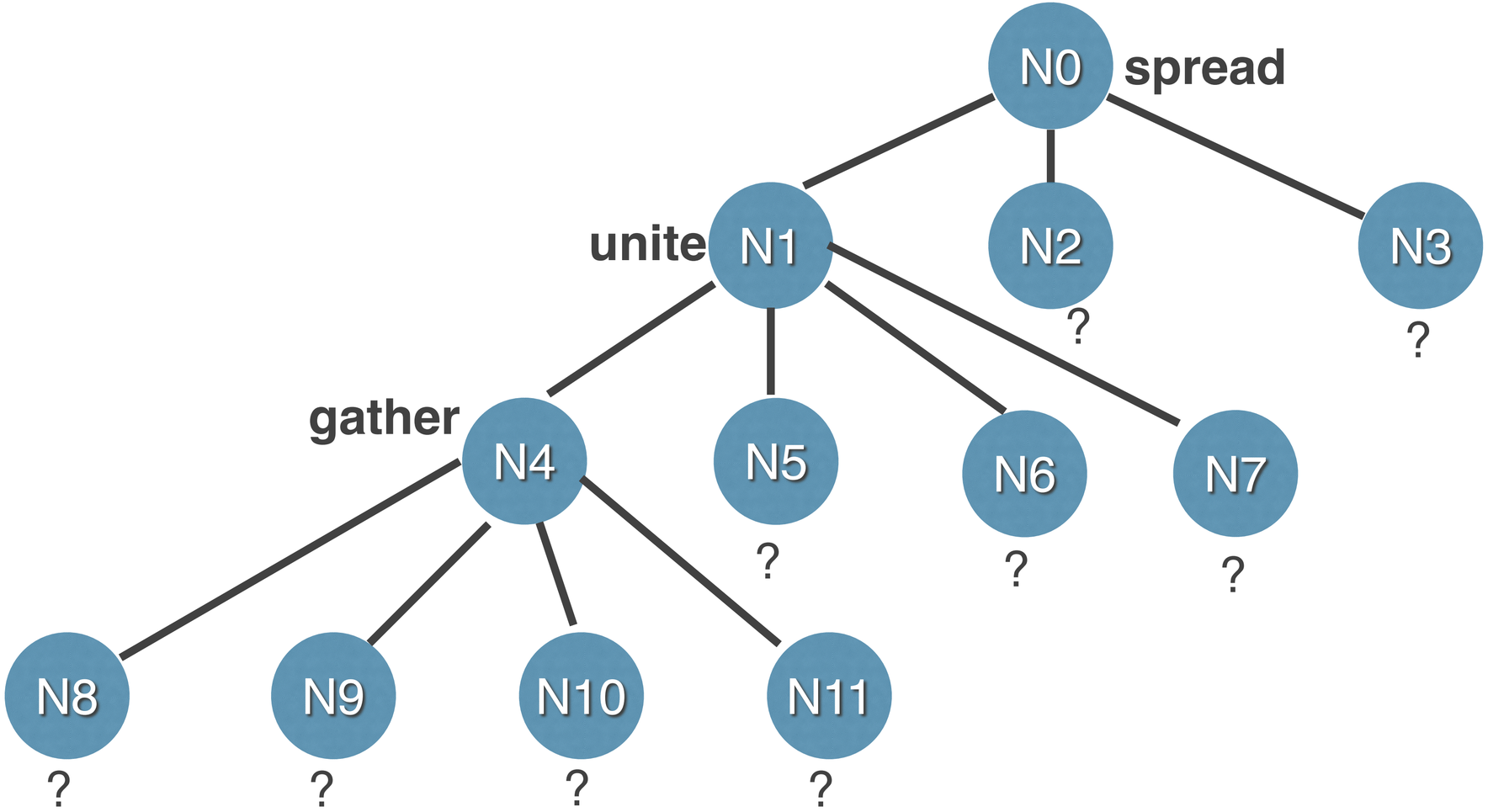
```
df1=gather(input,Score,Value,Score1,Score2)
df2=unite(df1,AllScores,Grade,Score)
output=spread(df2,AllScores,Value)
```



Symbolic Program

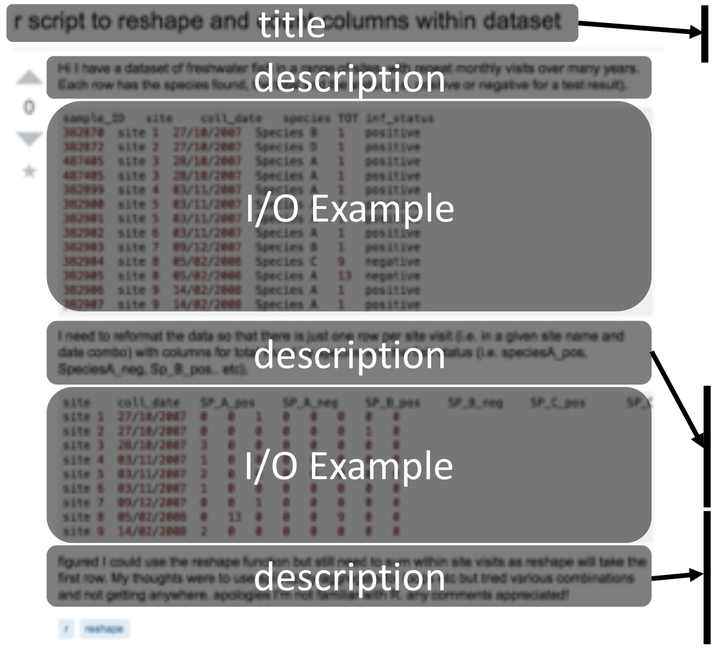
R program:

```
df1=gather(??)
df2=unite(??)
output=spread(??)
```



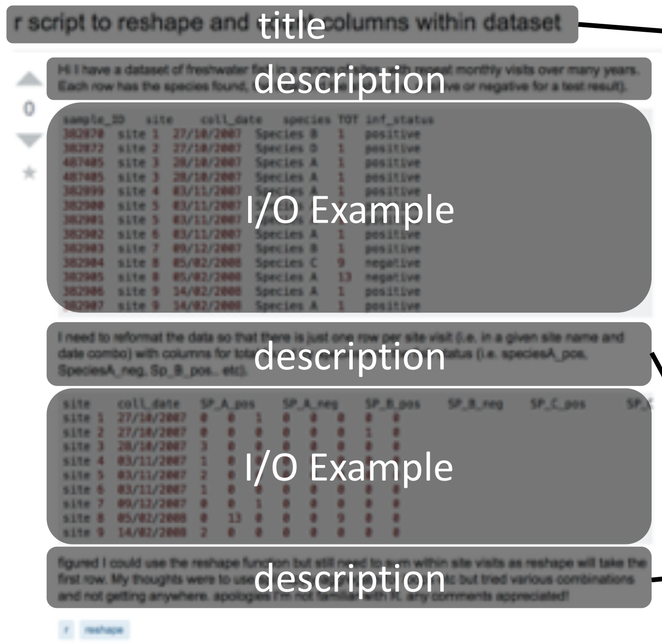
Problem Description

- Problem description provides hints to solutions



Problem Description

- Problem description provides hints to solutions



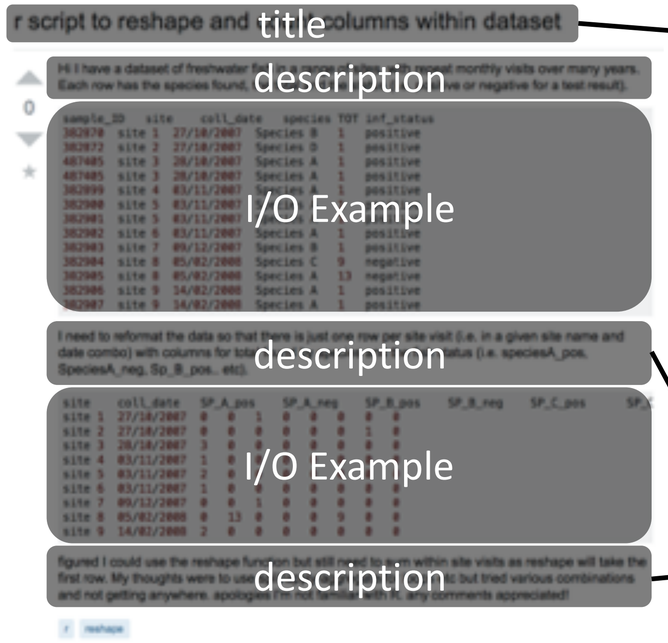
r script to reshape and count columns within dataset

I need to reformat the data so that there is just one row per site visit (i.e. in a given site name and date combo) with columns for total found by species and the fish status (i.e. speciesA_pos, SpeciesA_neg, Sp_B_pos.. etc).

figured I could use the reshape function but still need to sum within site visits as reshape will take the first row. My thoughts were to use split/apply/aggregate/for loops etc but tried various combinations and not getting anywhere. apologies I'm not familiar with R. any comments appreciated!

Problem Description

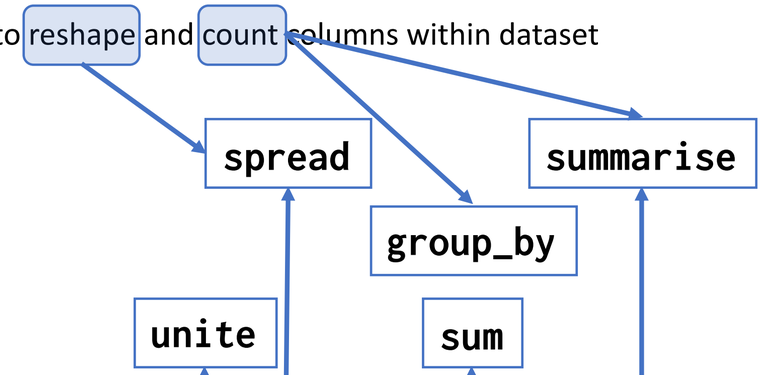
- Problem description provides hints to solutions



r script to reshape and count columns within dataset

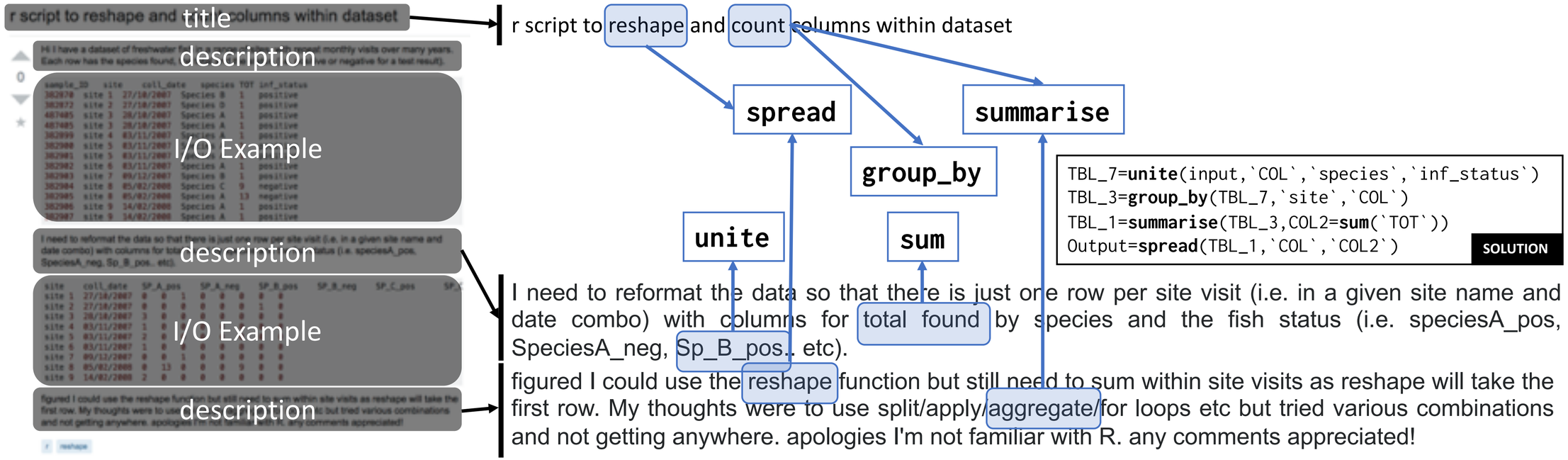
I need to reformat the data so that there is just one row per site visit (i.e. in a given site name and date combo) with columns for total found by species and the fish status (i.e. speciesA_pos, SpeciesA_neg, Sp_B_pos.. etc).

figured I could use the reshape function but still need to sum within site visits as reshape will take the first row. My thoughts were to use split/apply/aggregate/for loops etc but tried various combinations and not getting anywhere. apologies I'm not familiar with R. any comments appreciated!



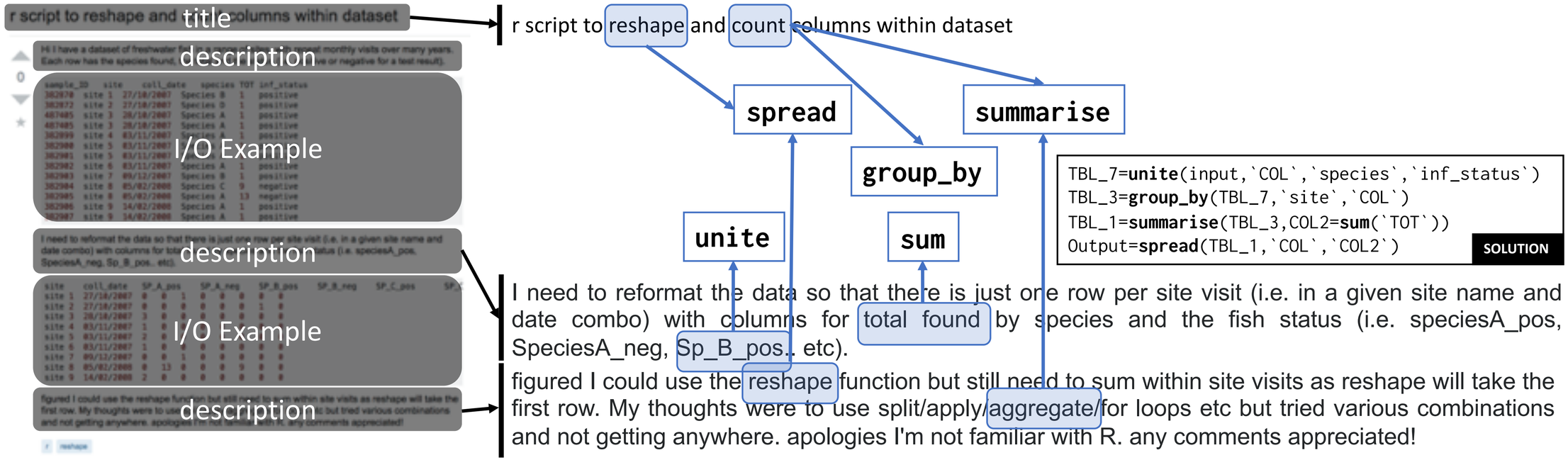
Problem Description

- Problem description provides hints to solutions



Problem Description

- Problem description provides hints to solutions



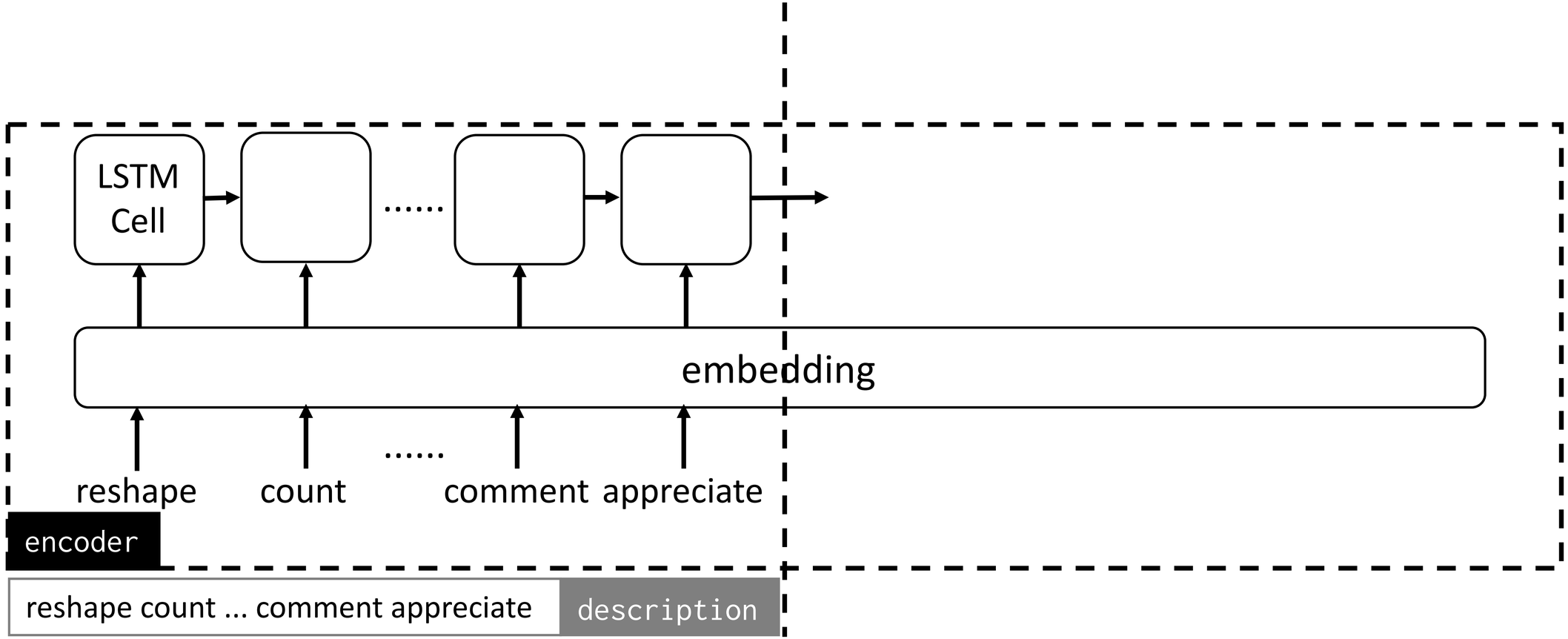
- To capture hints from natural language, we **model description** and **symbolic programs** relationships using **seq2seq** model

Neural Network

- Question-solution pair (D,S):
  - **Question** (D) is a user description composed by word tokens D = (d1, d2, …, dn):
    - D = (“r”, “script”, “to”, “reshape”, “and”,”count”,…)
  - **Solution** (S) is a symbolic program composed by a sequence of functions S = (s1, s2, …, sn)
    - S = (“unite”,”group_by”,”summarise”,”spread”,…)
  - **seq2seq** model is used to estimate the **probability** of **P(S|D)**

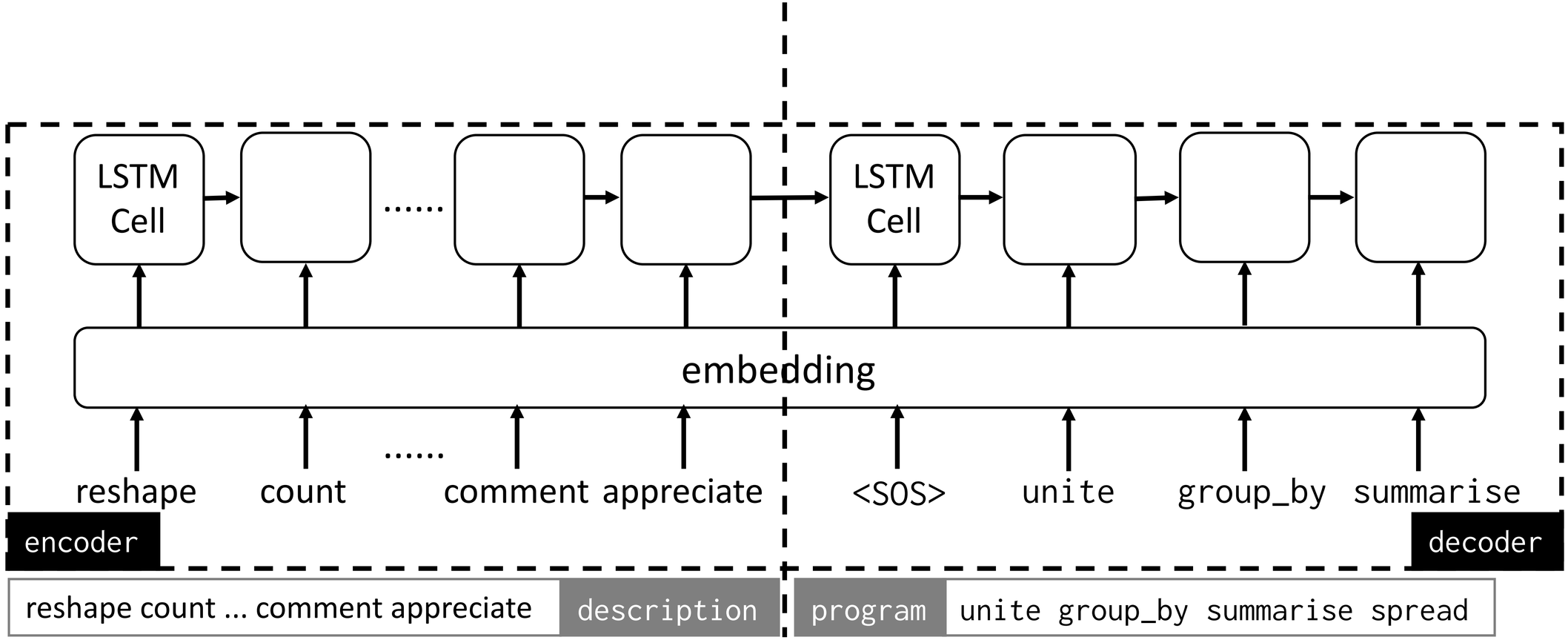
Neural Network

- Modeling the description and symbolic program relations



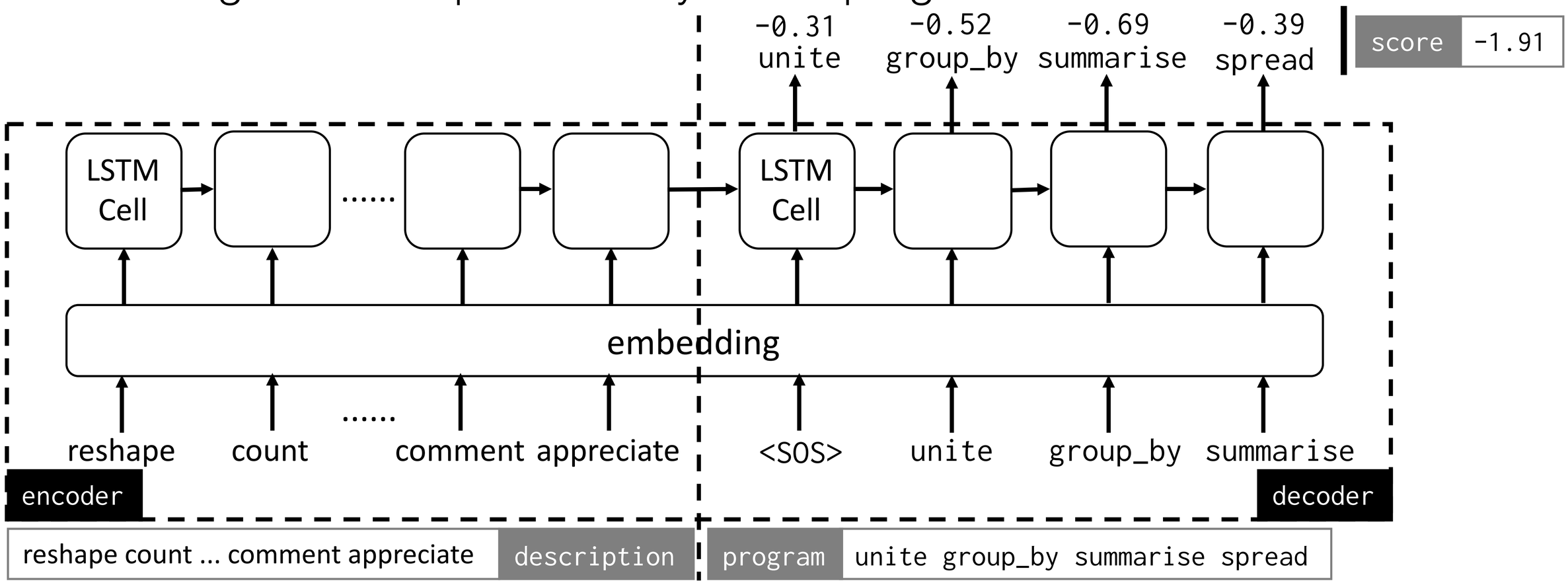
Neural Network

- Modeling the description and symbolic program relations



Neural Network

- Modeling the description and symbolic program relations



Neural Network

- Use beam search to generate a ranked list (P, w):
  - P - Symbolic Program
  - w - likelihood of being part of the solution
- Score is computed by summing log likelihood in every step

1. {mutate, group_by, summarise, spread}(92)
2. {group_by, summarise, mutate, select}(91)

…

130. {mutate, group_by, summarise, spread}(79)

…

Ranking improves

Still needs to explore many symbolic programs

Mining Association Rules

- Association rules
  - **X** => **Y** ; **X**: *keywords* ; **Y**: *functions*
- **Unsupervised learning** using the **apriori algorithm**
- Example of association rules:
  - {“reshape”, “count”} => {spread}
  - {“_”,“reshape”} => {unite}

Ranking further improves

1. {unite, group_by, summarise, spread}(109)

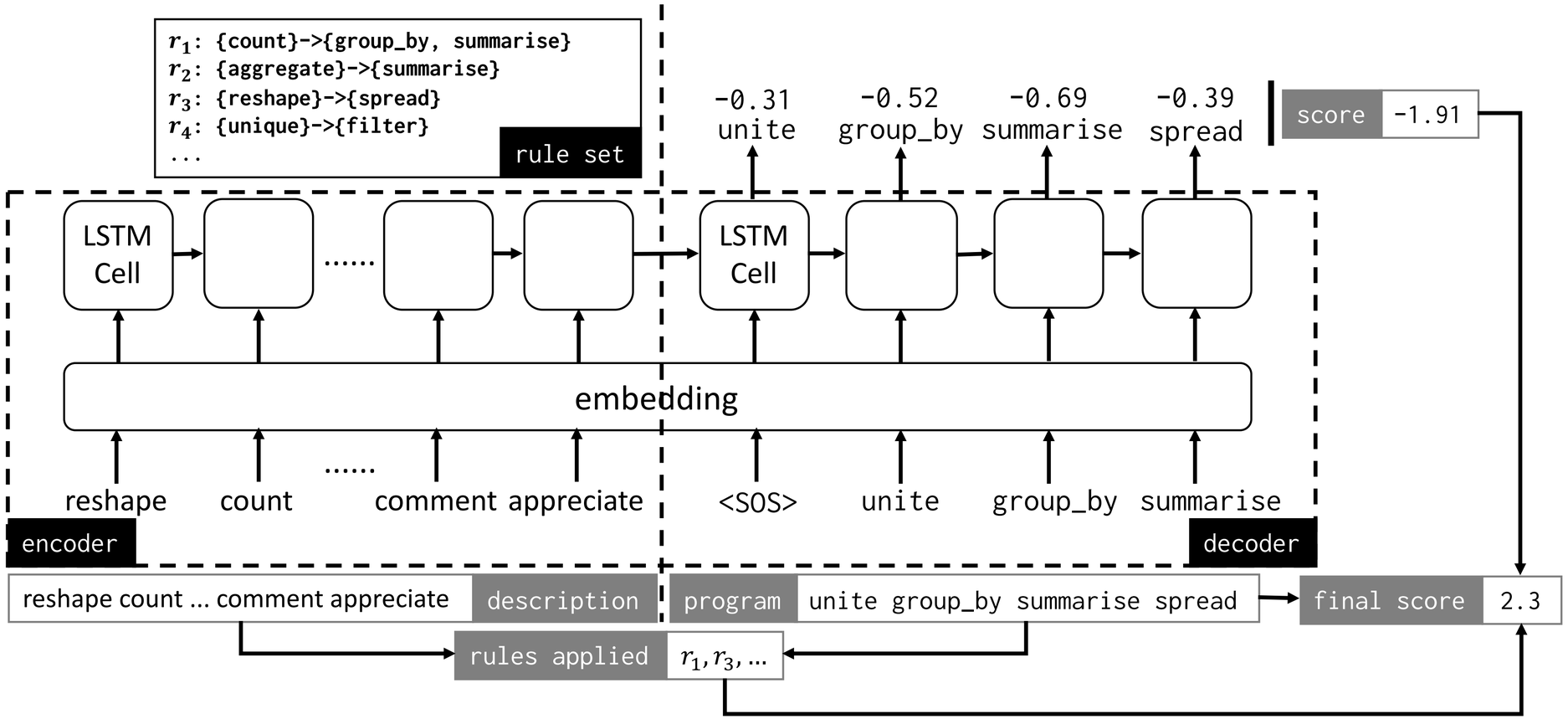
…

30. {mutate, group_by, summarise, spread}(94)

…

Hybrid Neural Network

- Use the learned association rule set to adjust the ranking scores on the fly



Program Synthesizers

Enumerative Search

Constraint Solving

Stochastic Search

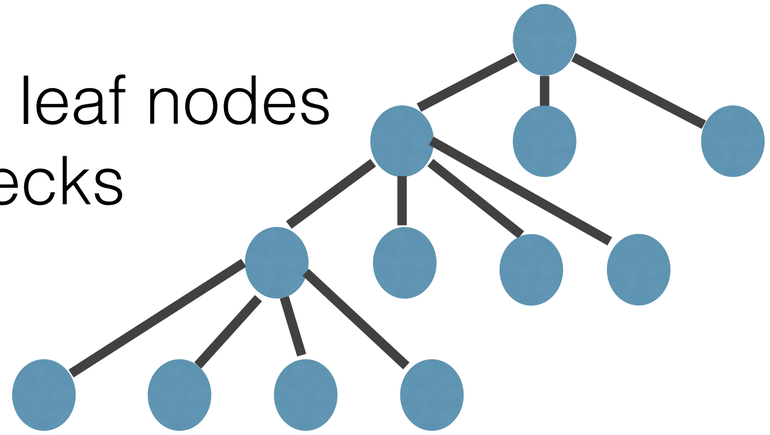
Most synthesizers do not combine different techniques in an **unified** way

Synthesis using Statistical Models

- **Hard specifications** (must be satisfied)
  - Input-output examples
  - Program type-checks
- **Soft specifications** (may be satisfied)
  - User preference in the form of natural language
- **Satisfy all hard** specifications while **maximizing** the sum of the weights of the **satisfied soft** specifications

Satisfiability Modulo Theories

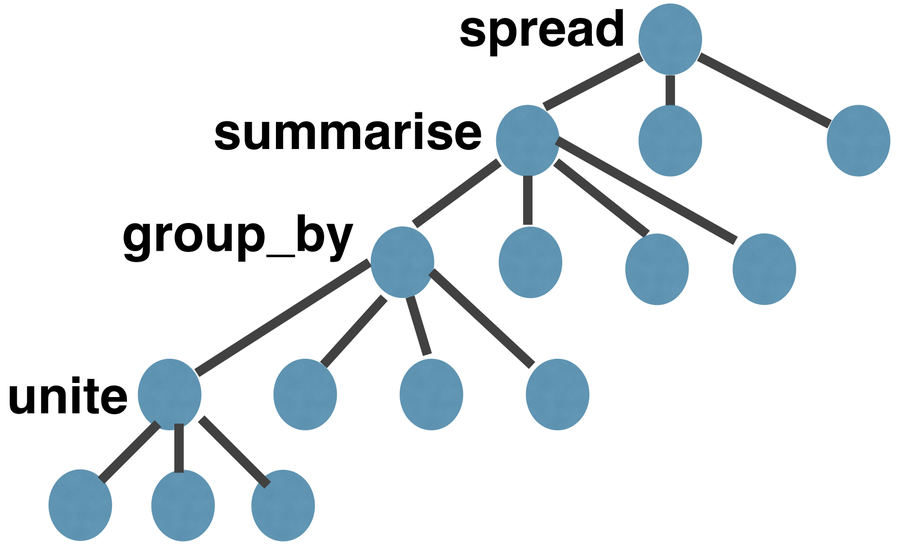
- **Satisfiability Modulo Theories** (SMT) problem is a decision problem for formulas that are composed with multiple theories
- We can encode the enumeration of concrete programs into SMT using Linear Integer Arithmetic (LIA)
- **Hard specifications:**
  1. Assign functions to the root node that are consistent with input-output examples
  2. Only constants and inputs can be assigned to the leaf nodes
  3. The concrete program is well formed, i.e. type-checks



Soft Specifications

- Symbolic program:

  TBL_7 = **unite**(??)
  TBL_3 = **group_by**(??)
  TBL_1 = **summarise**(??)
  Output = **spread**(??)



- Predicates:
  - (**occurs**(s), w): s occurs in the solution with likelihood w:
    - Example: (occurs(summarise), 109)
  - (**hasChild**(s, s'),w): s is a parent of s' in the solution with likelihood w:
    - Example: (hasChild(group_by,unite), 109)

Setup

- **seq2seq** neural network:
  - PyTorch framework
  - Hyperparameters are obtained through a simple grid search
  - Embedding layer:
    - Maps 25,004 words and 14 functions
    - To vectors of the dimension 256

- **Association rule mining:**
  - Efficient-Apriori package
  - Filter out association rules with low confidence

Data Collection and Preparation

- 80 benchmarks for data wrangling tasks using R libraries (*tidyr*, *dplyr*)
- Collected 20,640 pages from StackOverflow:
  - Testing benchmarks excluded
  - Each page contains a single question and multiple answers
- Removing duplicate and questions with no solutions:
  - 16,459 question-solution pairs used to train seq2seq
- Extract descriptions from answers and their corresponding solutions:
  - 37,748 transactions as input of the Apriori algorithm
  - Learned 187 association rules

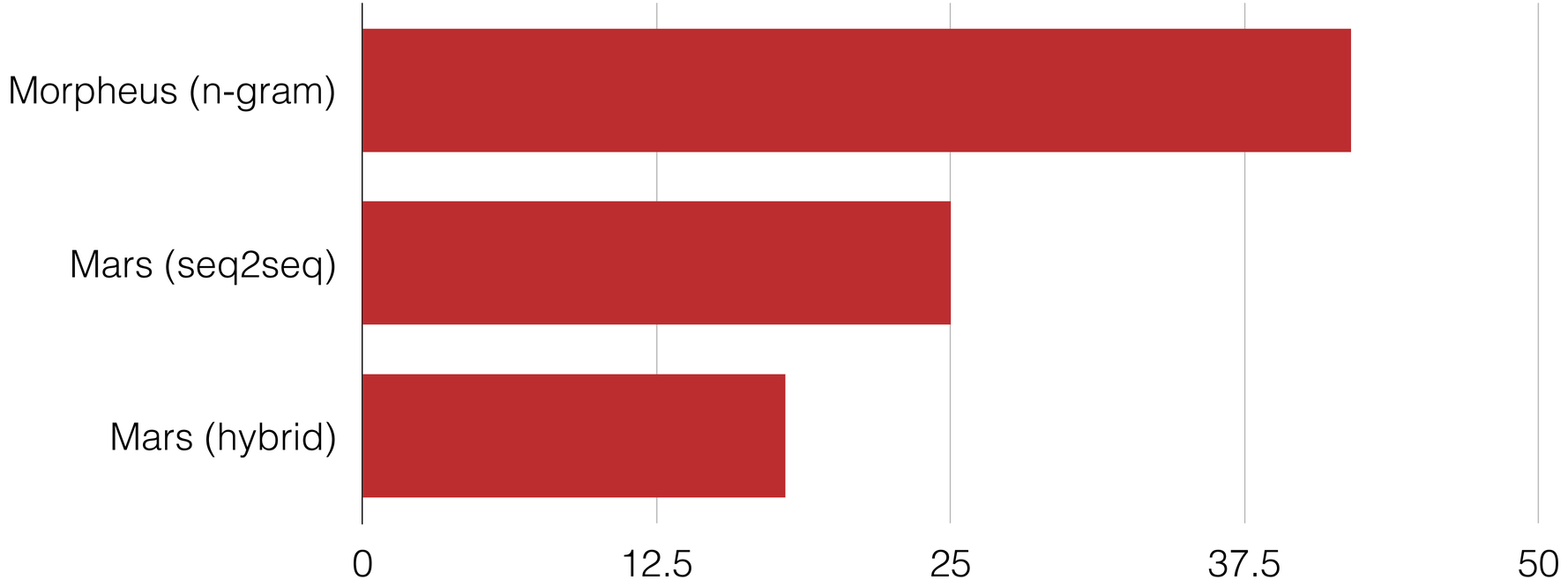
Evaluation — Data Wrangling

- **Morpheus:**
  - n-gram model
  - Global ranking that does not consider user description

- **Mars:**
  - Considers the user description for each task
  - seq2seq
  - Hybrid (seq2seq + apriori algorithm)

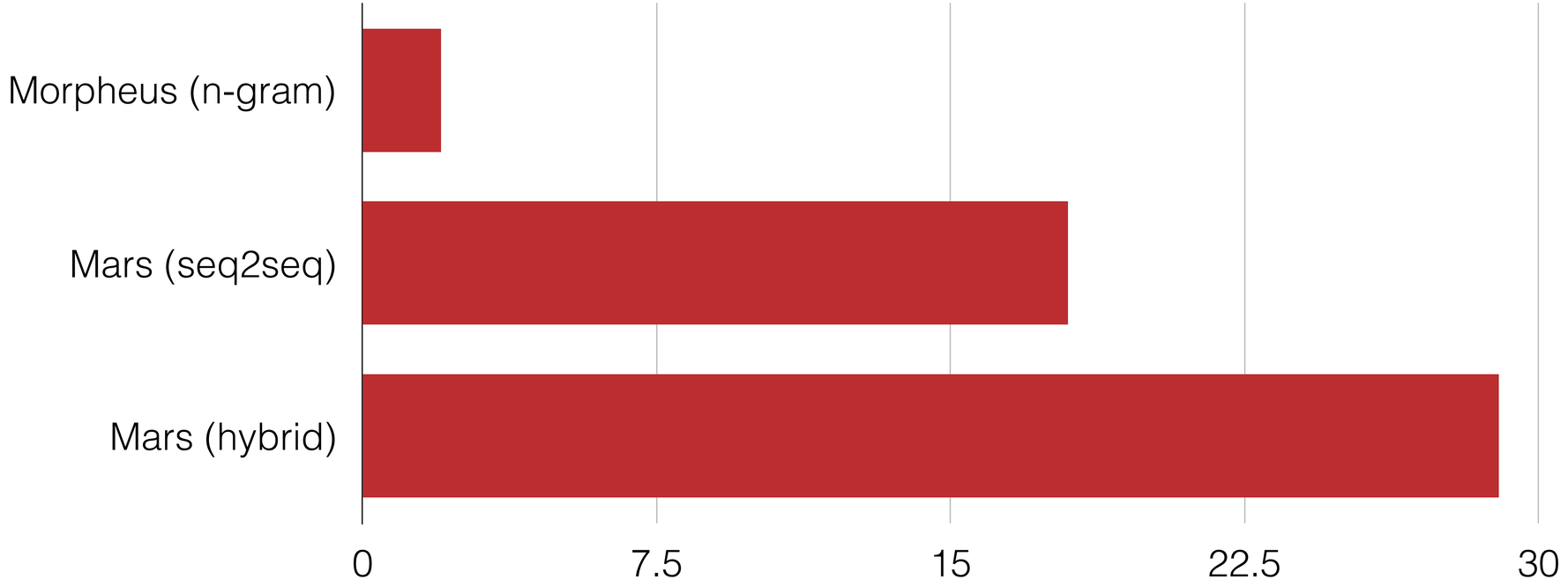
Quality of Suggested Candidates

- How many symbolic programs need to be enumerated until a solution is found? (smaller is better)

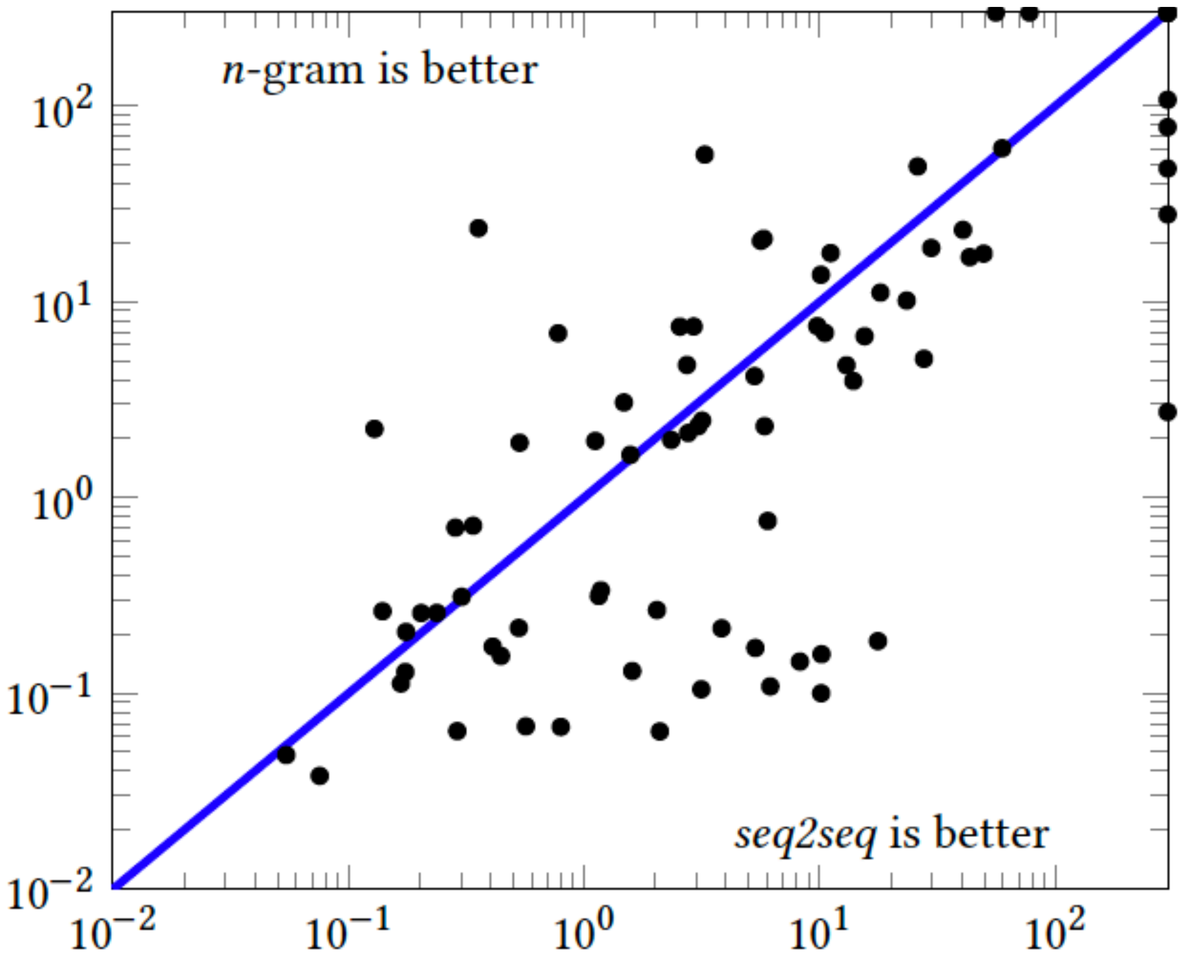


Quality of Suggested Candidates

- How often is the correct symbolic program among the first three? (larger is better)



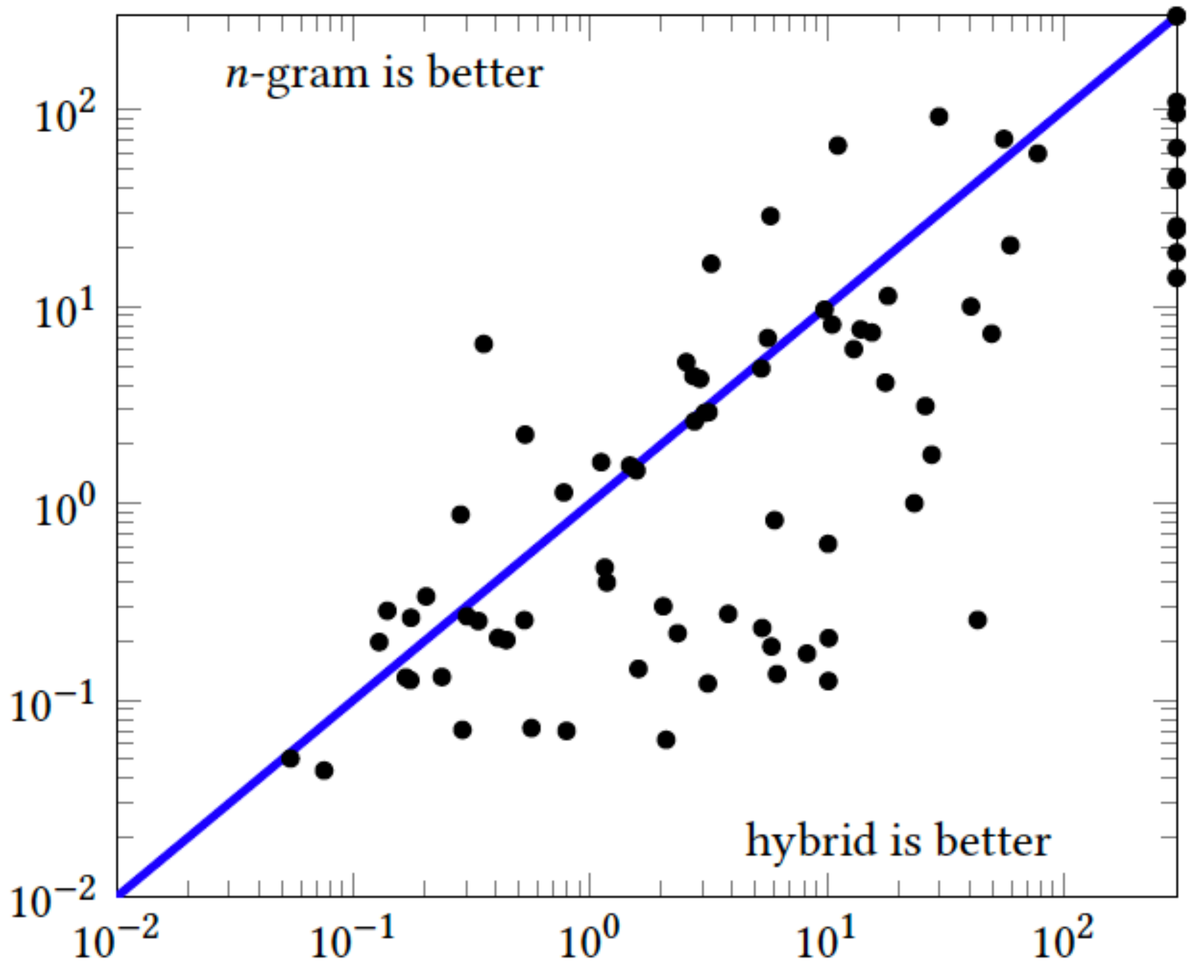
Performance Improvement



- Morpheus (n-gram) vs Mars (seq2seq)

	Avg. Speedup	**#timeouts**
Morpheus (n-gram)	1x	11
Mars (seq2seq)	6x	8

Performance Improvement

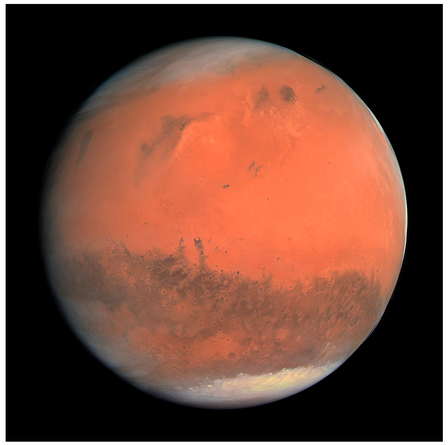


- Morpheus (n-gram) vs Mars (hybrid)

	Avg. Speedup	**#timeouts**
Morpheus (n-gram)	1x	11
Mars (hybrid)	15x	2

Takeaway

Mars



- Multiple specifications accurately capture the user intent
- seq2seq requires a lot of data:
  - Association rules can find hidden connections between keywords and functions
  - Hybrid neural network model achieves better accuracy
- We can encode multiple specifications as a Maximum Satisfiability Modulo Theory problem (Max-SMT)