Neural Inference of API Functions from Input-Output Examples

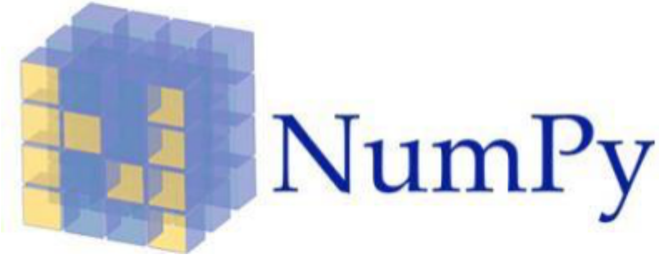
Rohan Bavishi, **Caroline Lemieux**, Neel Kant,
Roy Fox, Koushik Sen, Ion Stoica

Workshop on ML for Systems @ NeurIPS 2018

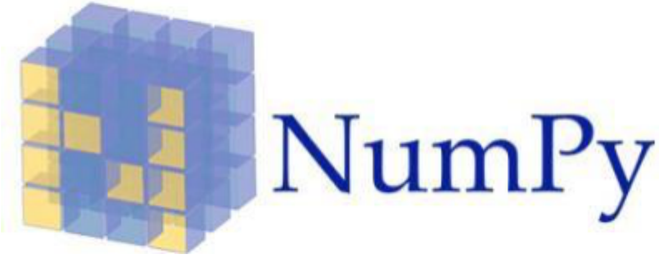
riselab
UC Berkeley

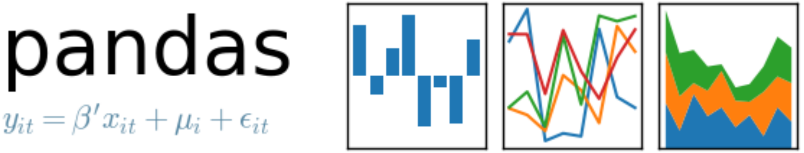
API Explosion!

API Explosion!

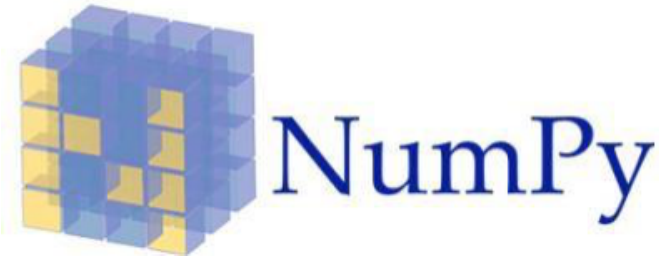


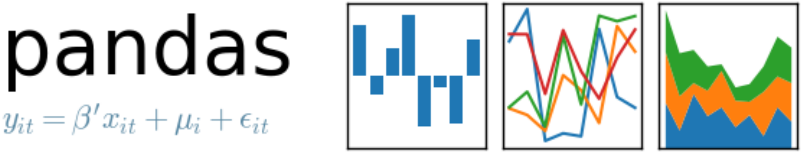
API Explosion!

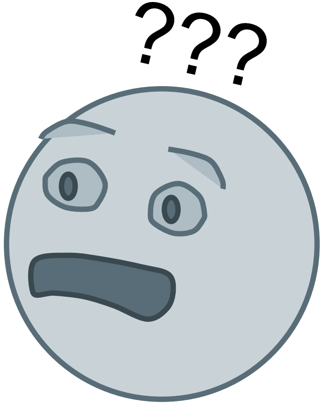




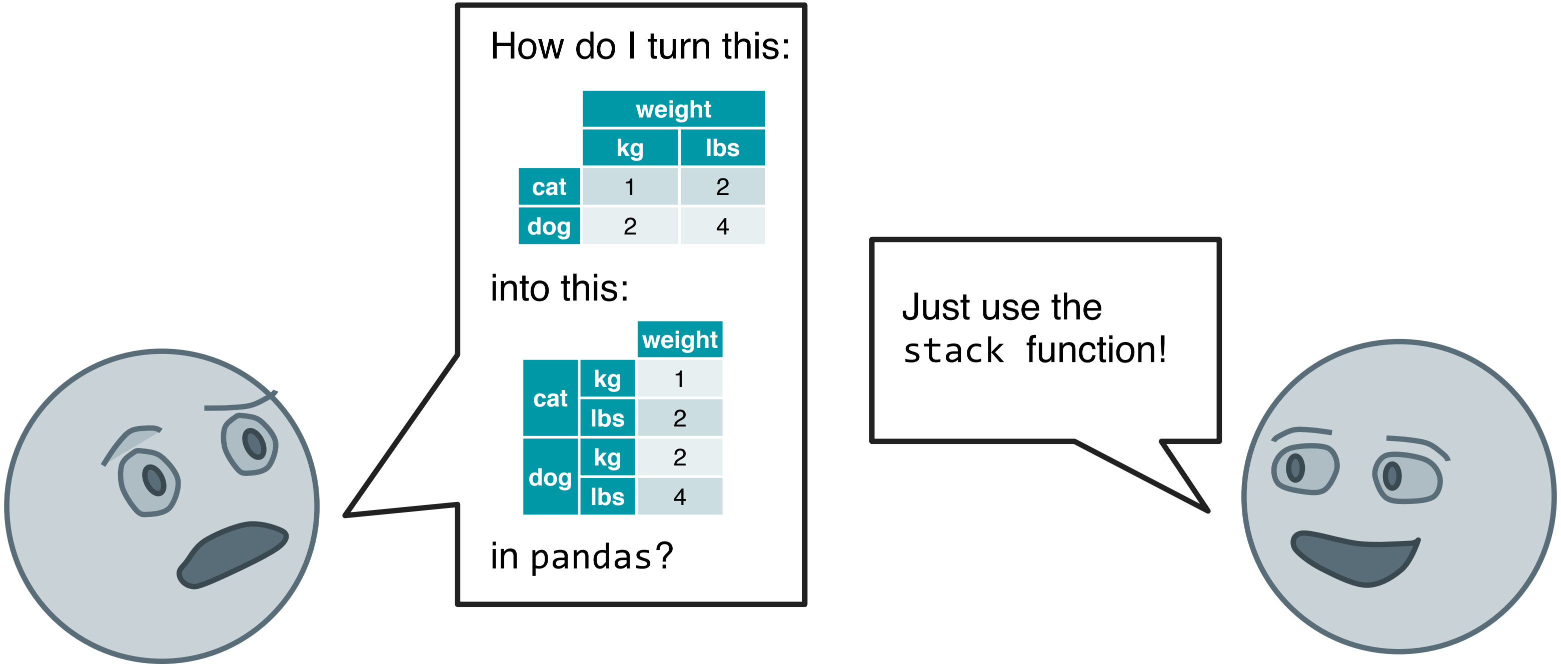
API Explosion!







How to cope? *StackOverflow*



How do I turn this:

	weight	
	kg	lbs
cat	1	2
dog	2	4

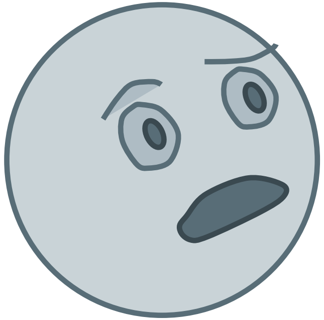
into this:

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in `pandas`?

Just use the `stack` function!

StackOverflow problems: Inefficient Solutions



How do I turn this:

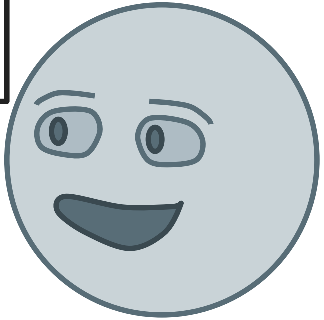
	weight	
	kg	lbs
cat	1	2
dog	2	4

into this:

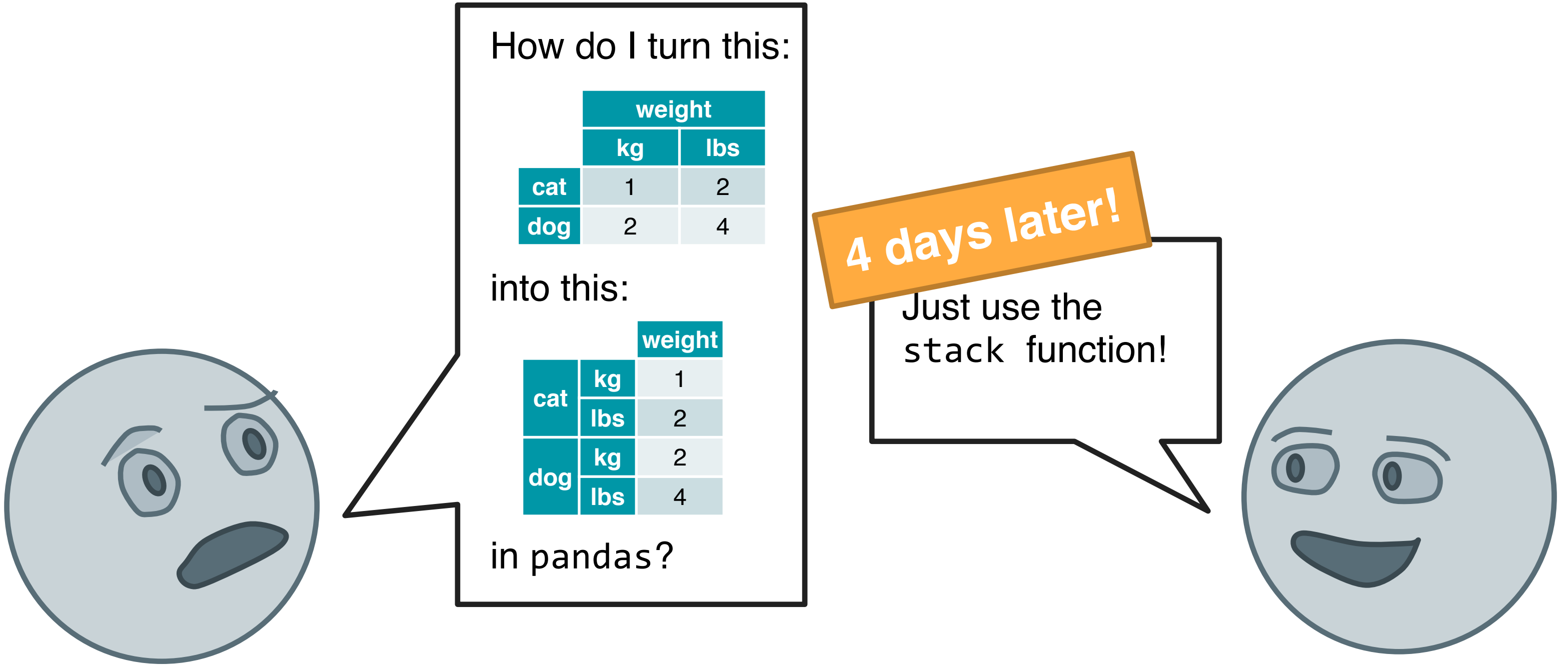
		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in pandas?

Well, you need to start by building the index `pd.MultiIndex(...`



StackOverflow problems: Slow Response…



How do I turn this:

	weight	
	kg	lbs
cat	1	2
dog	2	4

into this:

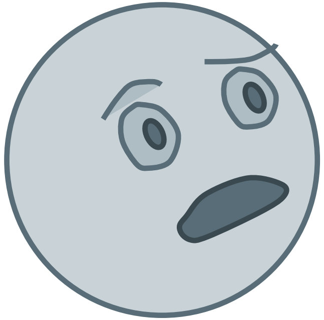
		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in pandas?

4 days later!

Just use the `stack` function!

StackOverflow problems: No Response



How do I turn this:

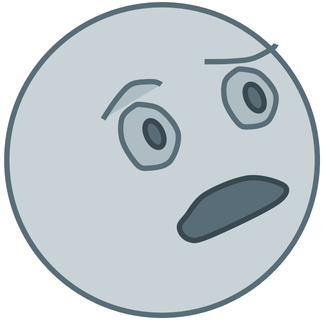
	weight	
	kg	lbs
cat	1	2
dog	2	4

into this:

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in pandas?

StackOverflow problems: No Response



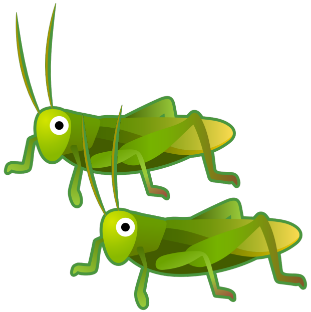
How do I turn this:

	weight	
	kg	lbs
cat	1	2
dog	2	4

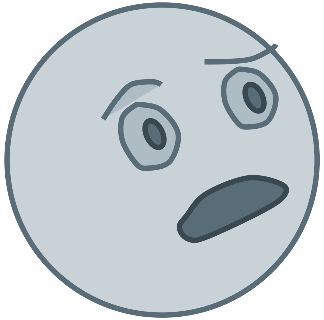
into this:

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in pandas?



Our Goal: Automate *StackOverflow* for APIs



How do I turn this:

	weight	
	kg	lbs
cat	1	2
dog	2	4

into this:

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in pandas?

```
output = input.stack(
             level=[1],
             dropna=True
         )
```

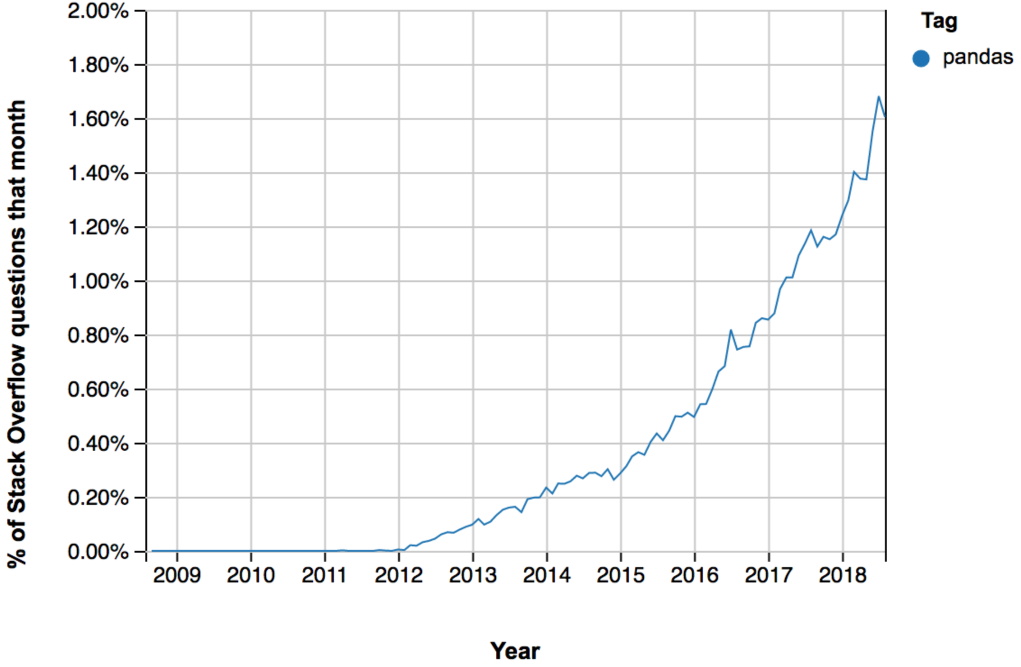
Technique Goals

- Program synthesis engine in *realistic, wide* API (vs. narrow DSL)
- Scale to complex data structures
- Scale to 100s of functions, 1000s of arguments

First Target API: *pandas* library

pandas

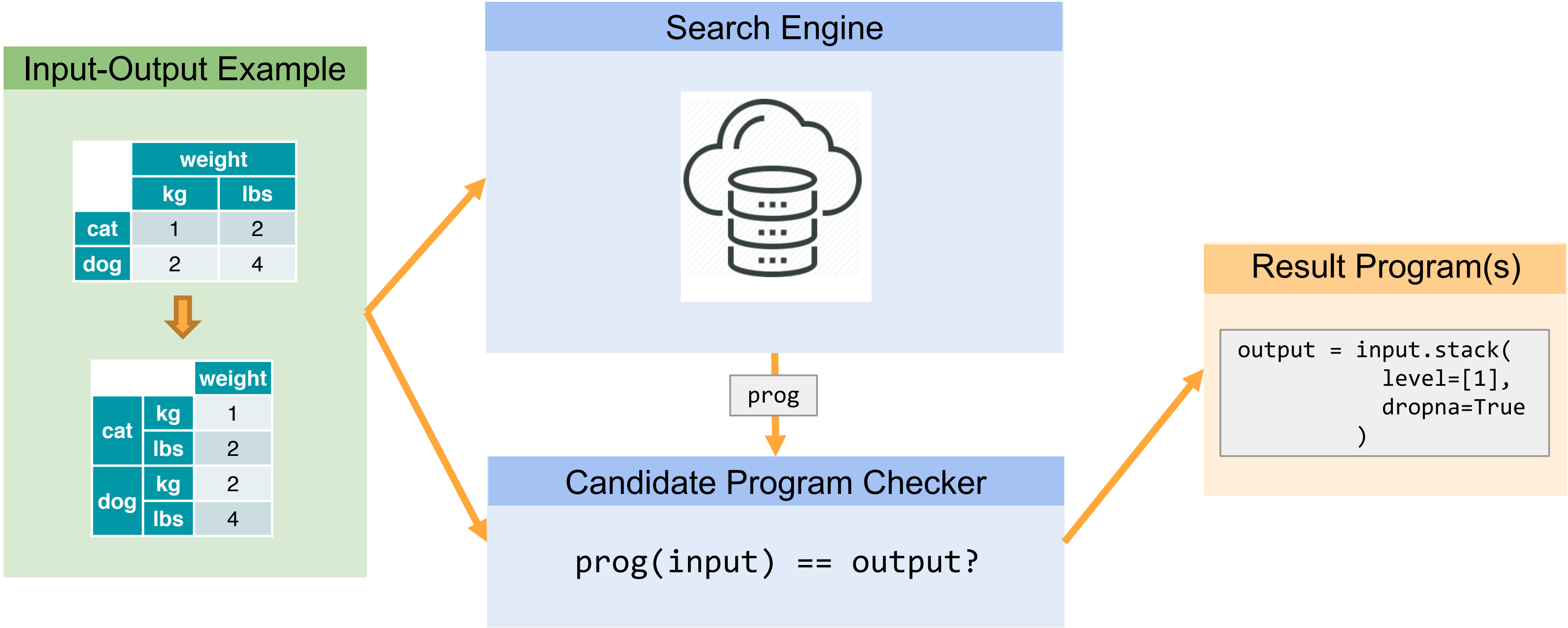
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



Technique Goals + *pandas*

- Program synthesis engine in *realistic* API
  - *Library of choice for data scientists*
- Scale to complex data structures
  - *DataFrames*
- Scale to 100s of functions, 1000s of arguments
  - ***10^{17}*** *branching factor for* ***depth 1***!

Synthesis Technique



Input-Output Example

	weight	
	kg	lbs
cat	1	2
dog	2	4

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

Search Engine

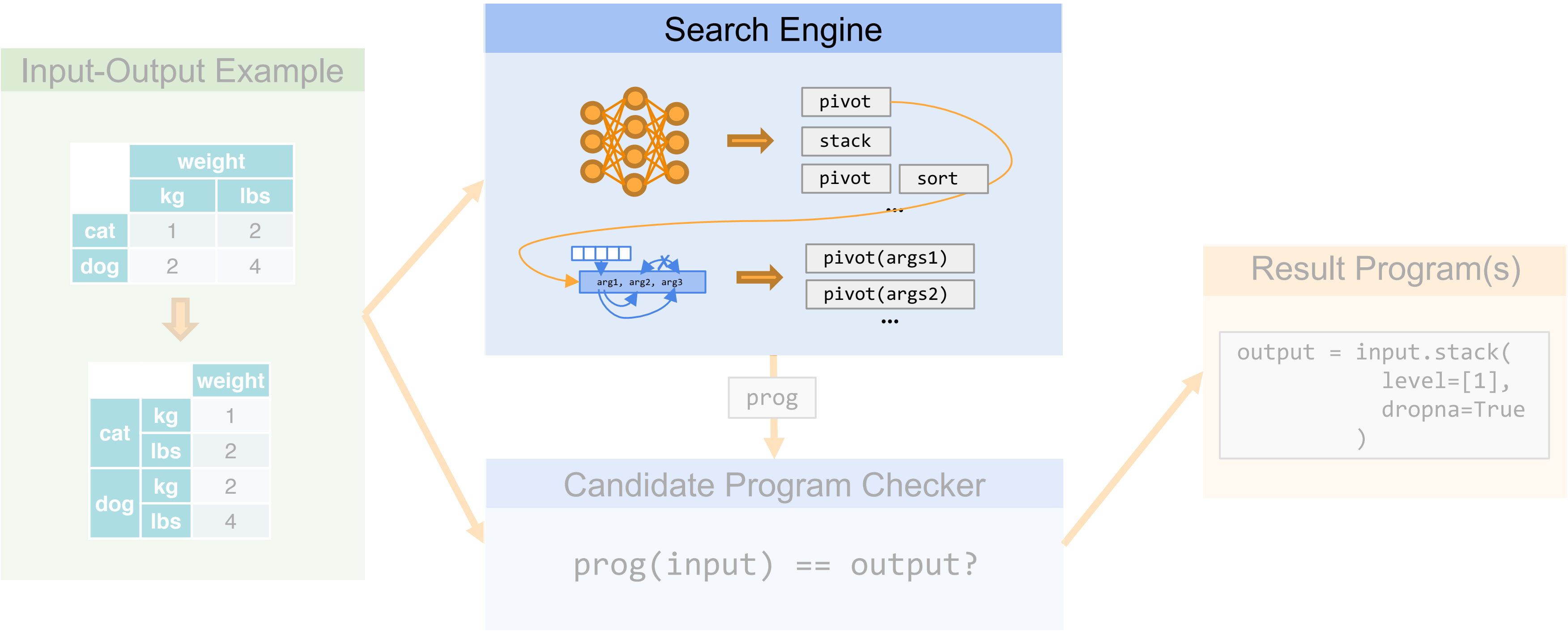
Candidate Program Checker

```
prog(input) == output?
```

Result Program(s)

```
output = input.stack(
           level=[1],
           dropna=True
         )
```

Search Technique



Input-Output Example

	weight	
	kg	lbs
cat	1	2
dog	2	4

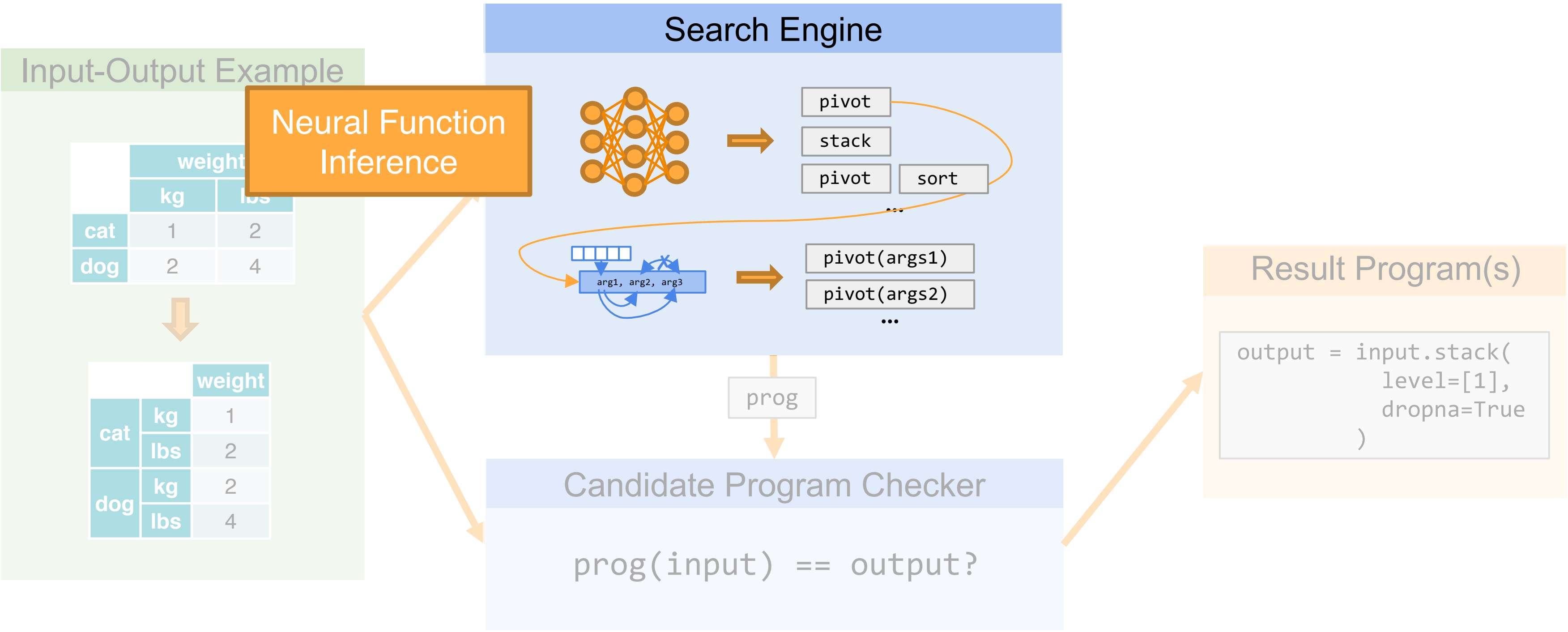
		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

```
prog(input) == output?
```

Result Program(s)

```
output = input.stack(
            level=[1],
            dropna=True
         )
```

Search Technique Step 1



Input-Output Example

	weight	
	kg	lbs
cat	1	2
dog	2	4

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

Neural Function Inference

Search Engine

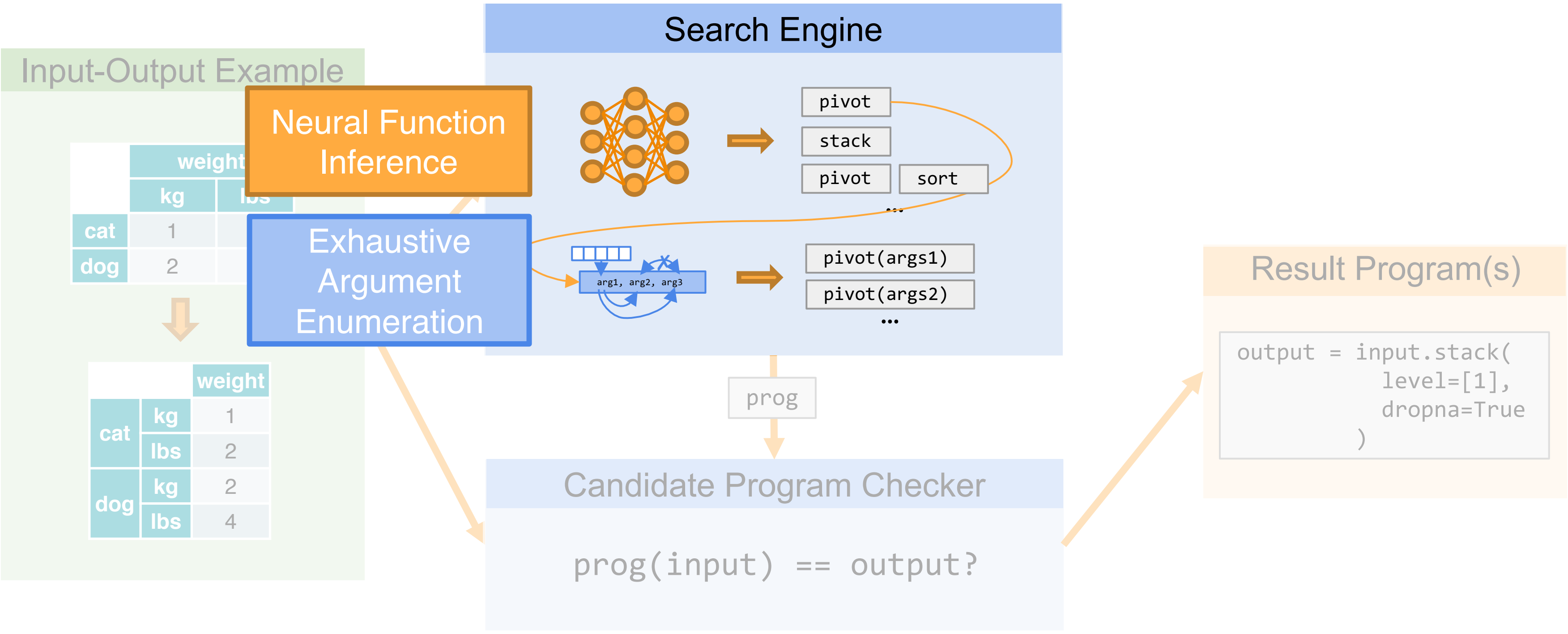
Candidate Program Checker

```
prog(input) == output?
```

Result Program(s)

```
output = input.stack(
            level=[1],
            dropna=True
         )
```

Search Technique Step 2



Input-Output Example

	weight	
	kg	lbs
cat	1	
dog	2	

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

Neural Function Inference

Exhaustive Argument Enumeration

Search Engine

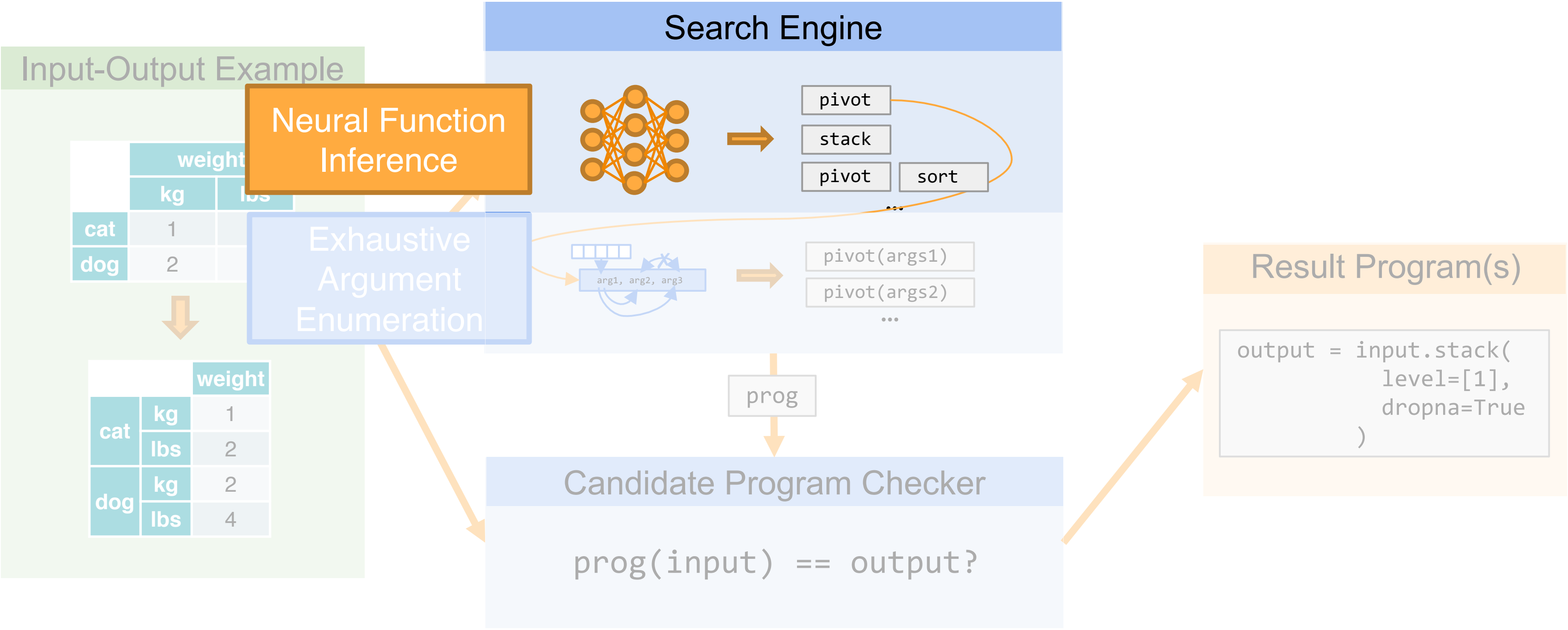
Candidate Program Checker

```
prog(input) == output?
```

Result Program(s)

```
output = input.stack(
            level=[1],
            dropna=True
         )
```

Focus: Neural Prediction Problem



Input-Output Example

	weight	
	kg	lbs
cat	1	
dog	2	

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

Search Engine

Neural Function Inference

```
pivot
stack
pivot  sort
...
```

Exhaustive Argument Enumeration

```
pivot(args1)
pivot(args2)
...
```

```
prog
```

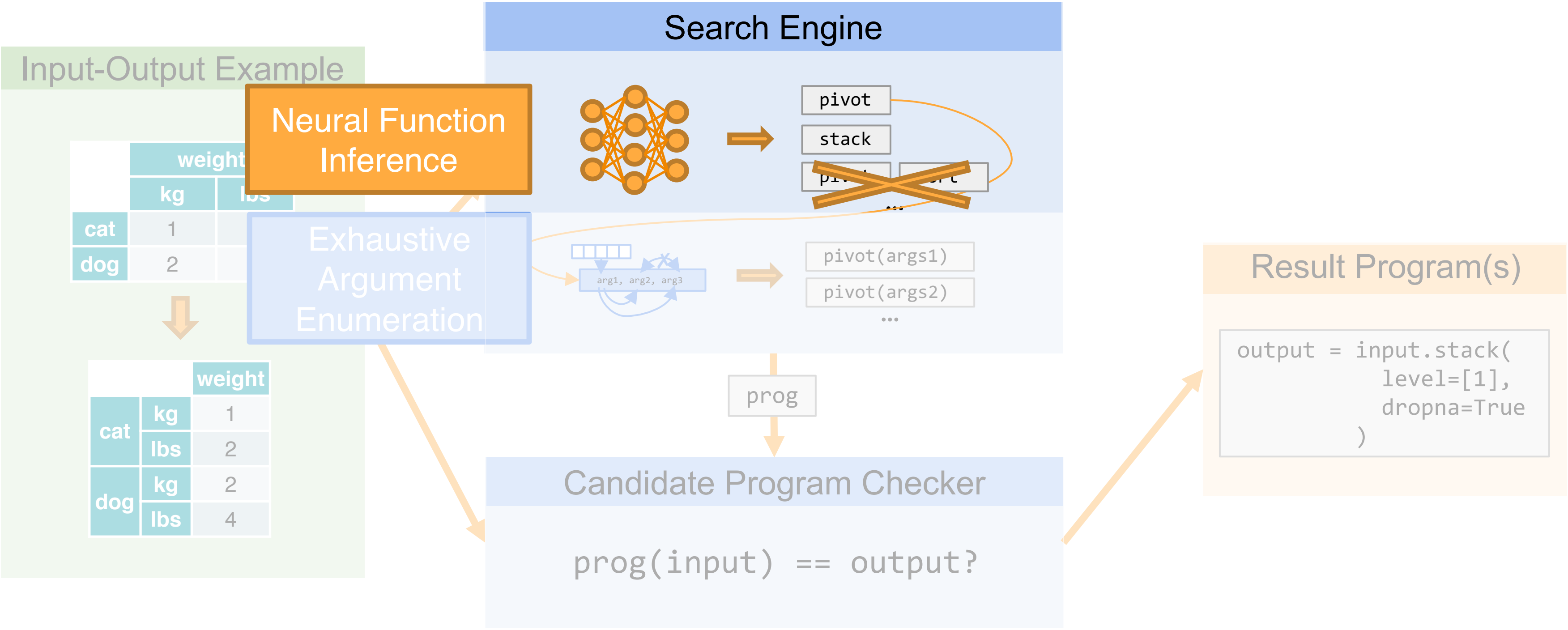
Candidate Program Checker

```
prog(input) == output?
```

Result Program(s)

```
output = input.stack(
           level=[1],
           dropna=True
         )
```

Focus: Neural Prediction Problem (Depth 1)



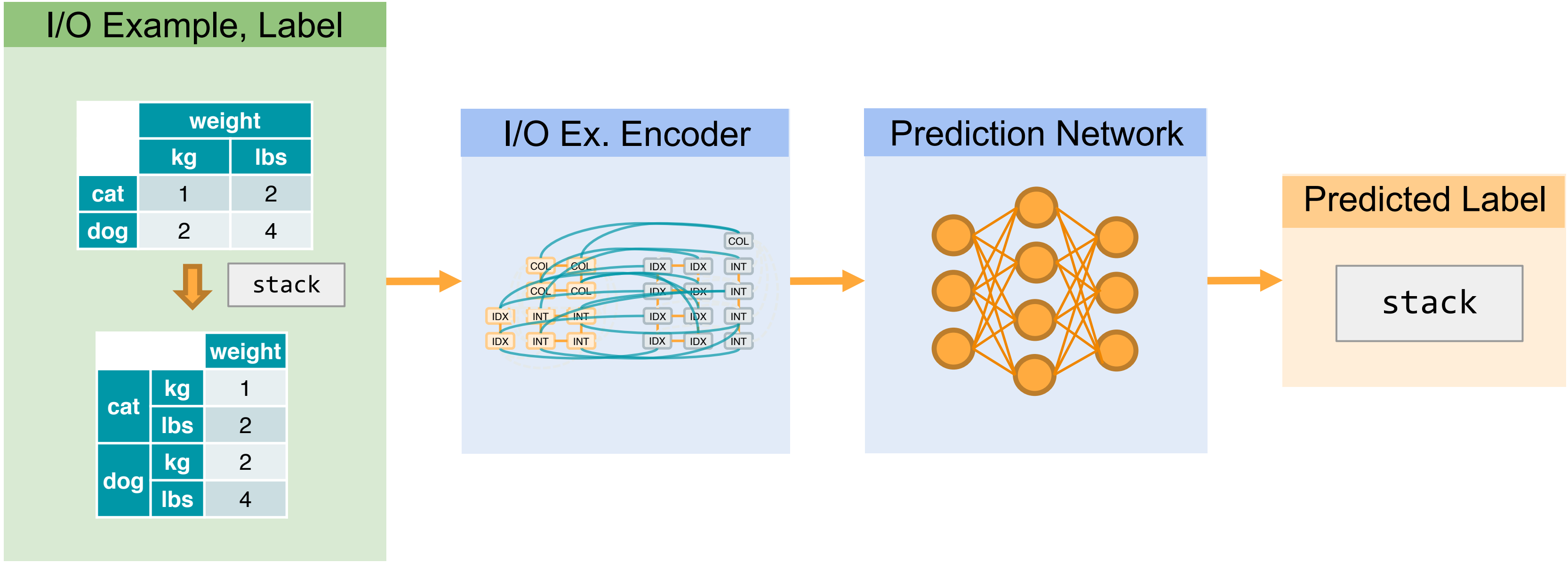
	weight	
	kg	lbs
cat	1	
dog	2	

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

```
output = input.stack(
             level=[1],
             dropna=True
         )
```

©2018 RISELab

Zoom in: Neural Prediction Problem (Depth 1)



I/O Example, Label

	weight	
	kg	lbs
cat	1	2
dog	2	4

stack

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

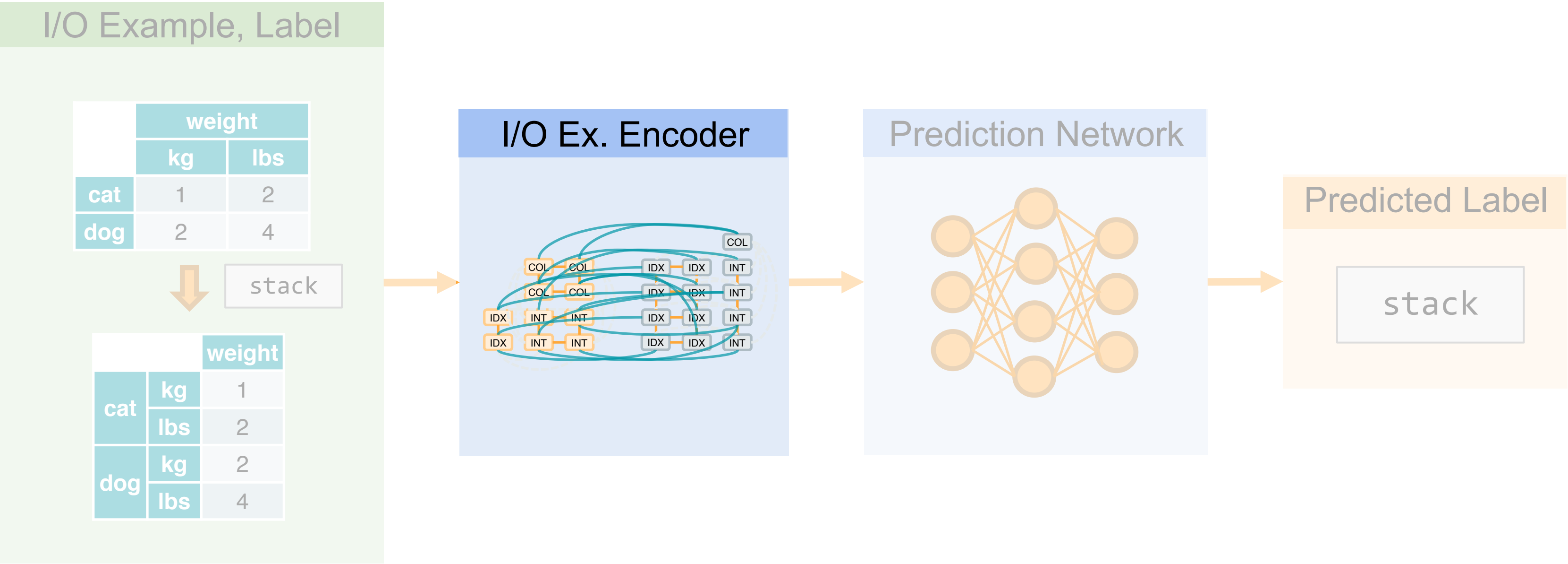
I/O Ex. Encoder

Prediction Network

Predicted Label

stack

Step 1: Encoding I/O Example



I/O Example, Label

	weight	
	kg	lbs
cat	1	2
dog	2	4

stack

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

I/O Ex. Encoder

Prediction Network

Predicted Label

stack

Step 1: Encoding I/O Example

Challenge 1: DataFrames of ***arbitrary size, shape***

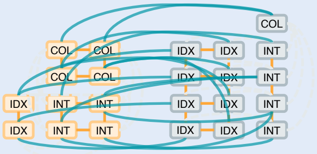
I/O Example, Label

	weight	
	kg	lbs
cat	1	2
dog	2	4

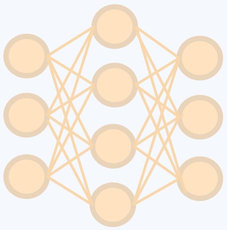
⇩ `stack`

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

I/O Ex. Encoder



Prediction Network



Predicted Label

`stack`

Step 1: Encoding I/O Example

Challenge 1: DataFrames of ***arbitrary size, shape***

Challenge 2: ***Infinite*** space of primitive values

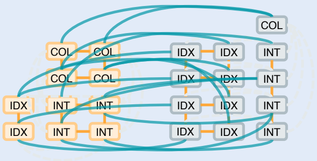
I/O Example, Label

	weight	
	kg	lbs
cat	1	2
dog	2	4

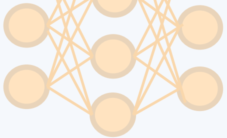
stack

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

I/O Ex. Encoder

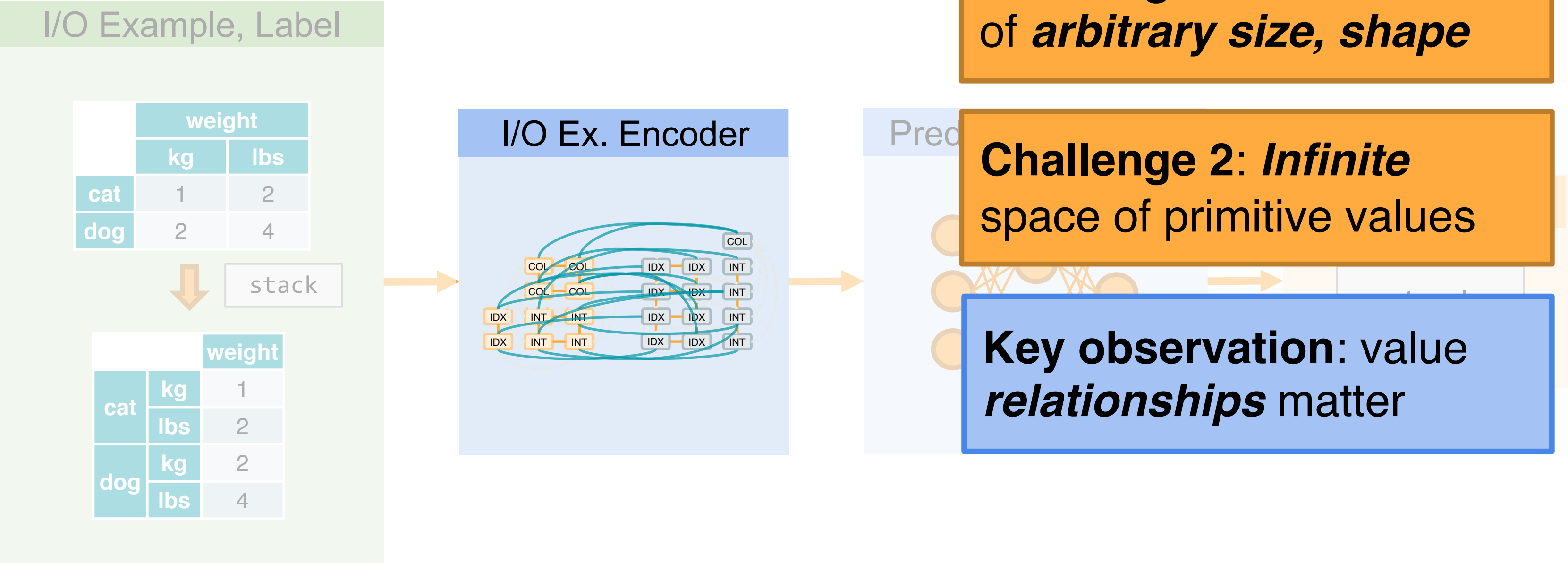


Pred



stack

Step 1: Encoding I/O Example



	weight	
	kg	lbs
cat	1	2
dog	2	4

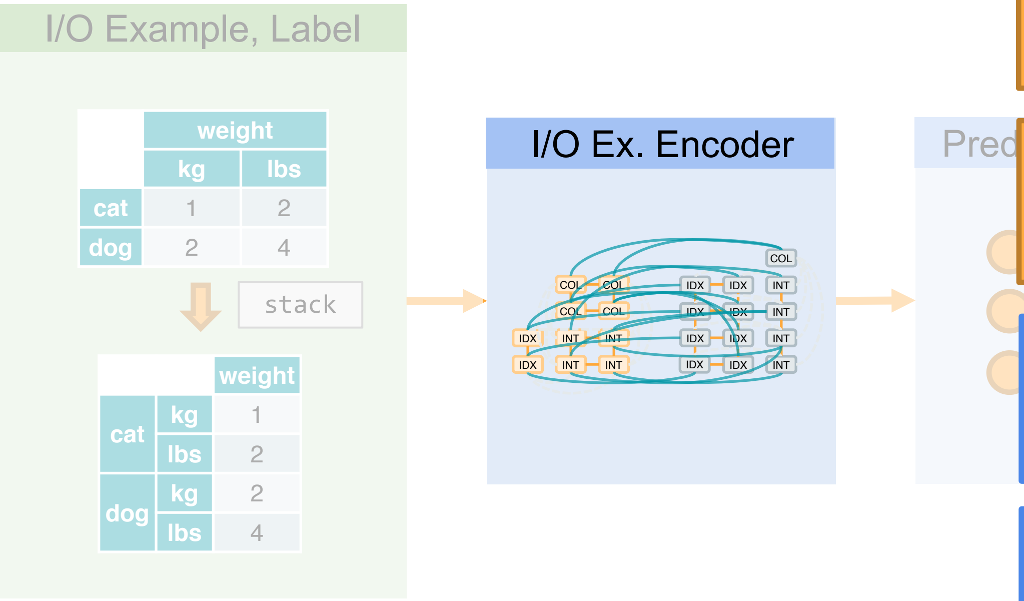
		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

Challenge 1: DataFrames of ***arbitrary size, shape***

Challenge 2: ***Infinite*** space of primitive values

Key observation: value ***relationships*** matter

Step 1: Encoding I/O Example



	weight	
	kg	lbs
cat	1	2
dog	2	4

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

Challenge 1: DataFrames of ***arbitrary size, shape***

Challenge 2: ***Infinite*** space of primitive values

Key observation: value ***relationships*** matter

Solution: ***graph-based*** encoding of I/O Example

Encoding an I/O Example as Graph

	weight	
	kg	lbs
cat	1	2
dog	2	4

input

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

output

Encoding: Cells, Indices → Nodes

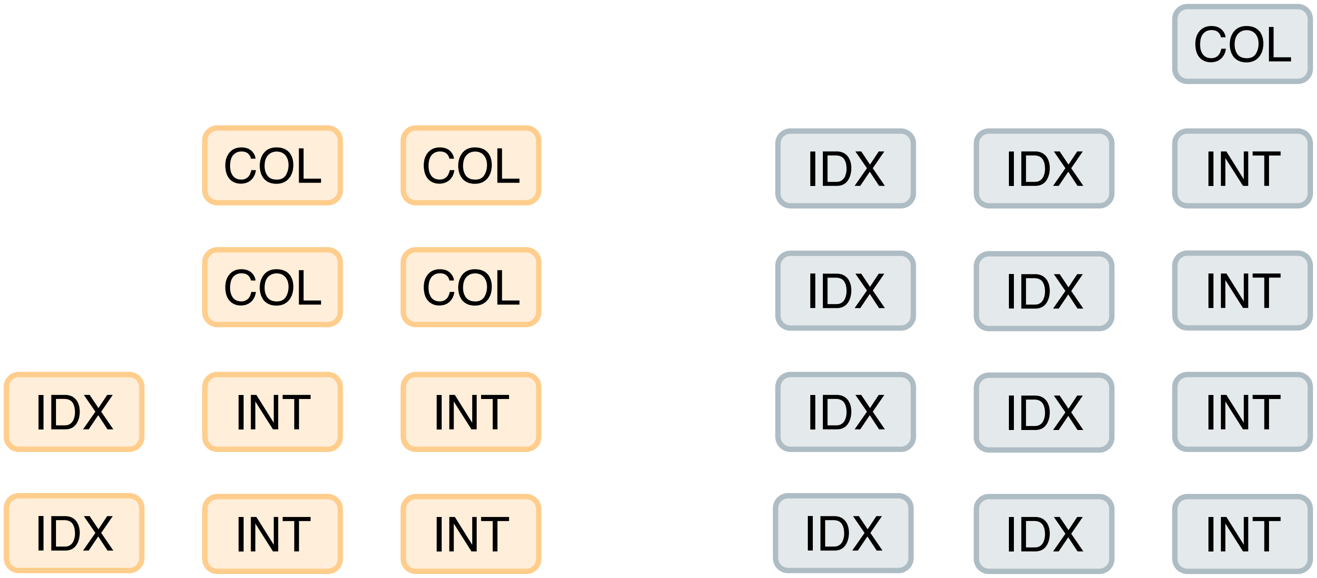
	weight	weight
	kg	lbs
cat	1	2
dog	2	4

input

		weight
cat	kg	1
cat	lbs	2
dog	kg	2
dog	lbs	4

output

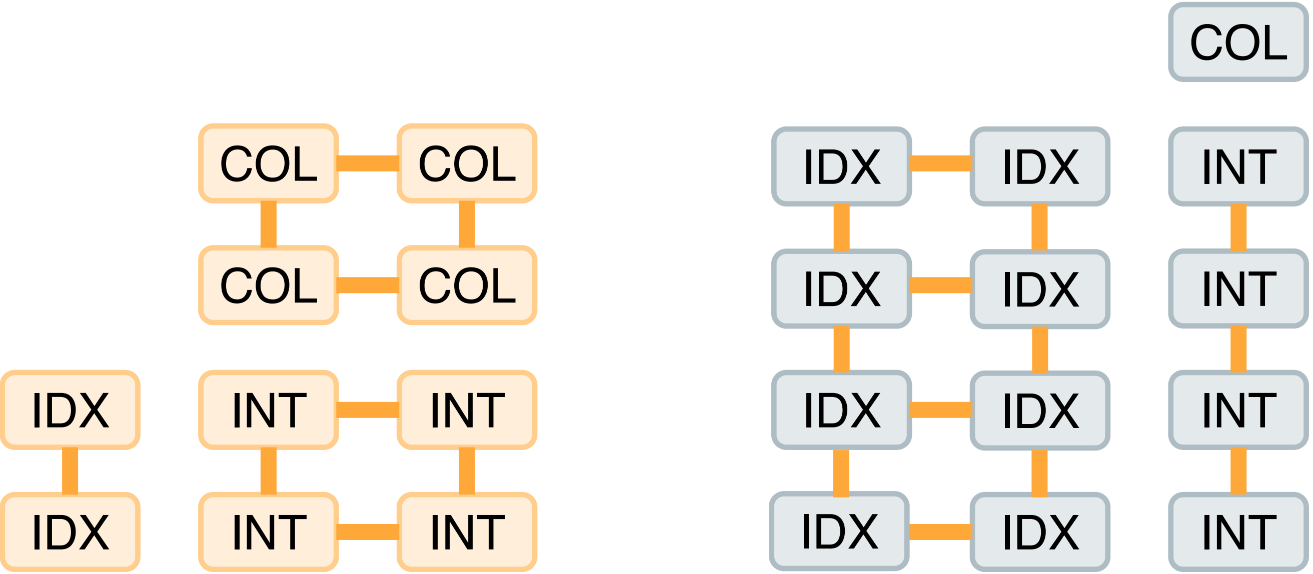
Encoding: Primitive Values → Types



	COL	COL
	COL	COL
IDX	INT	INT
IDX	INT	INT

		COL
IDX	IDX	INT
IDX	IDX	INT
IDX	IDX	INT
IDX	IDX	INT

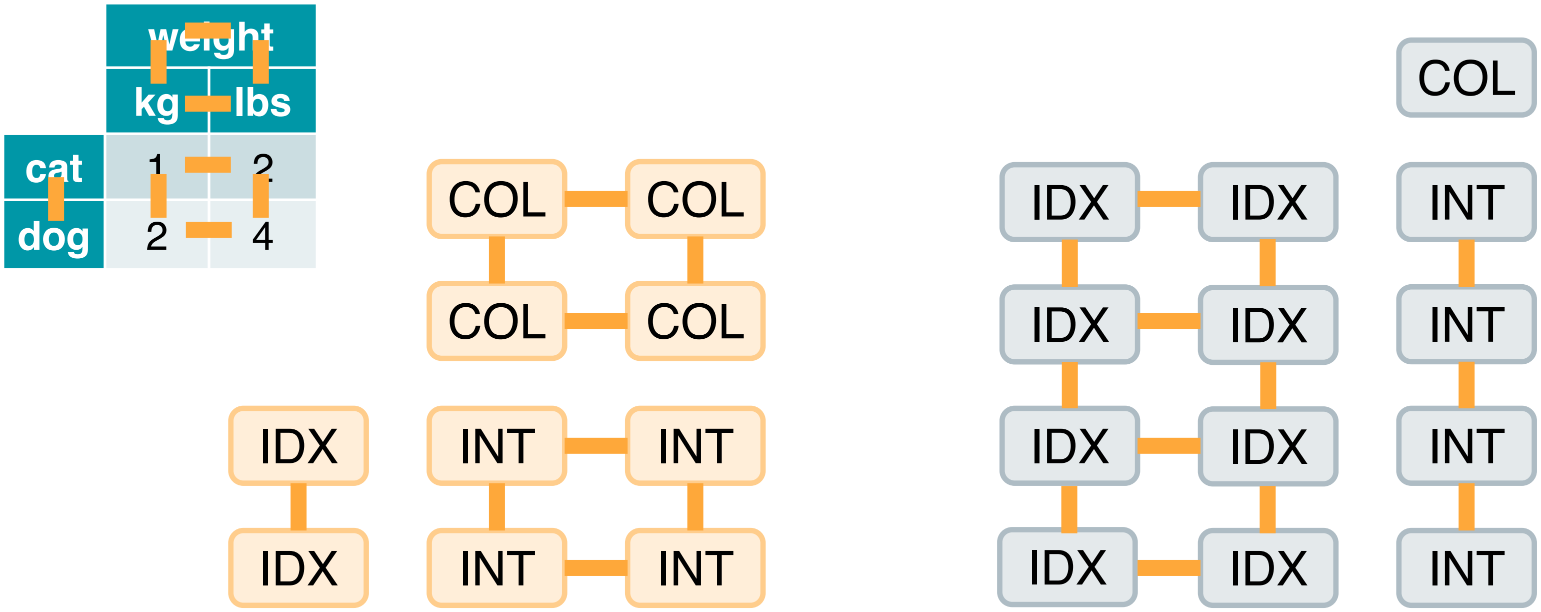
Encoding: *Adjacency* Edges



COL COL
COL COL
IDX INT INT
IDX INT INT

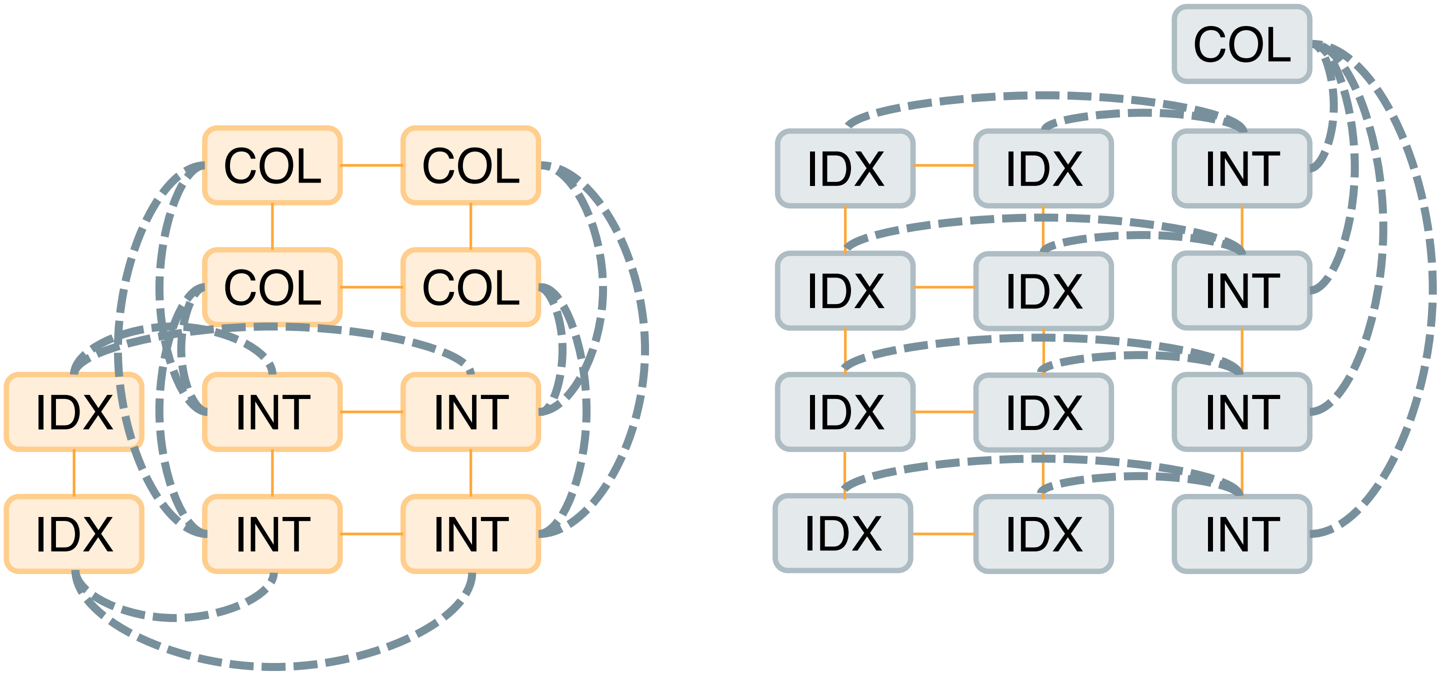
COL
IDX IDX INT
IDX IDX INT
IDX IDX INT
IDX IDX INT

Encoding: *Adjacency* Edges



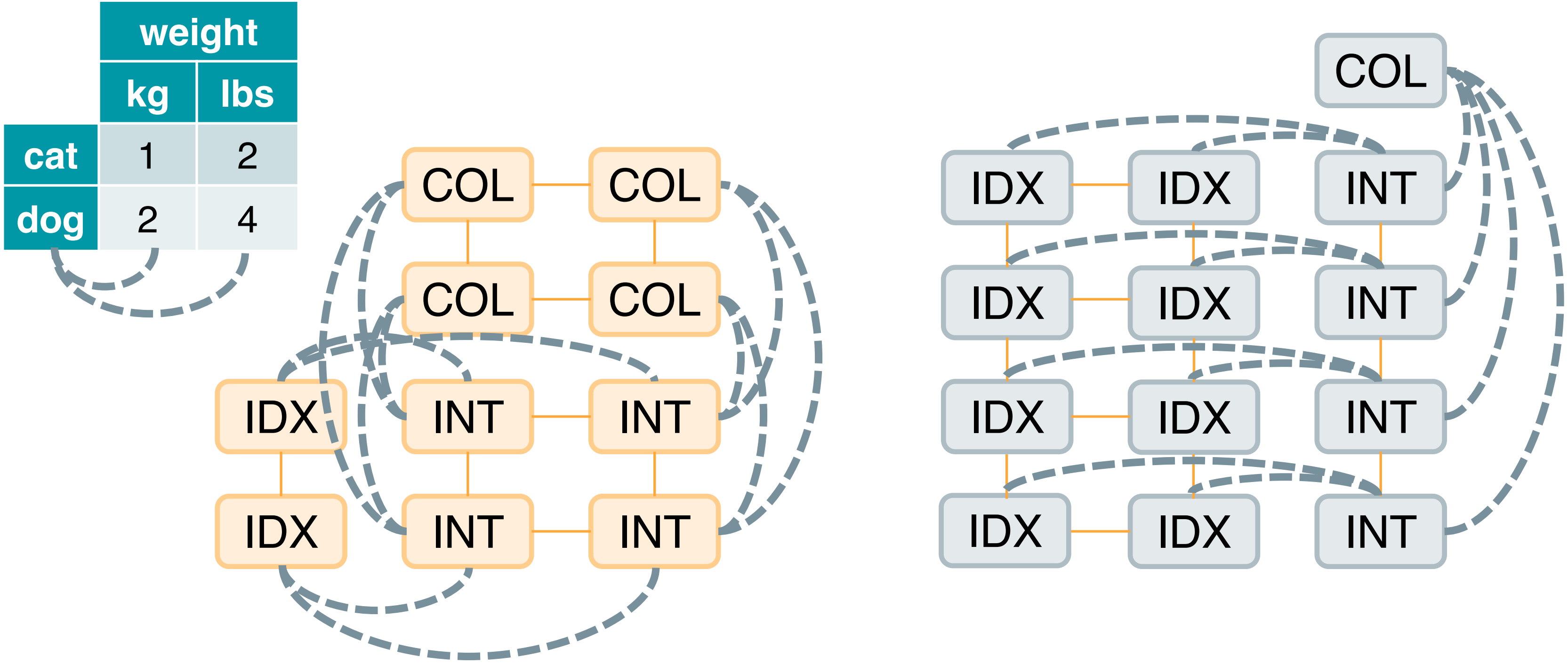
	weight	
	kg	lbs
cat	1	2
dog	2	4

Encoding: *Indexing* Edges



©2018 RISELab

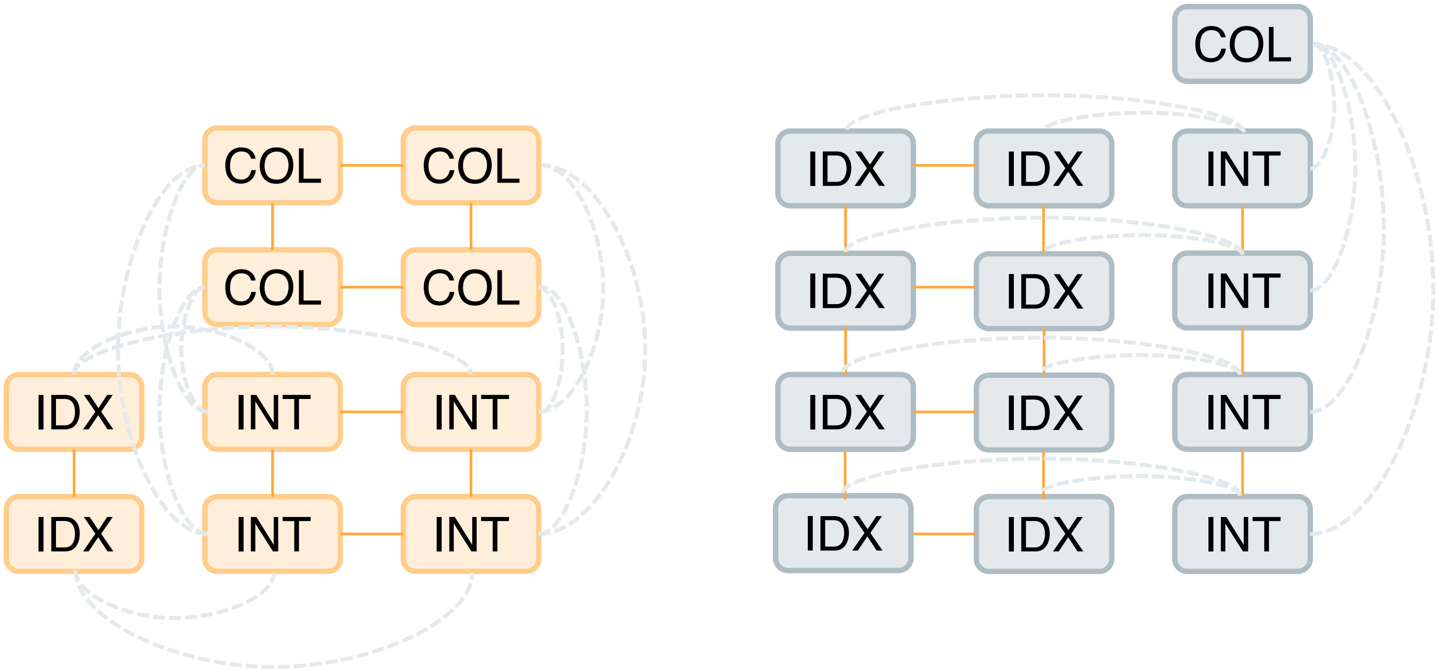
Encoding: *Indexing* Edges



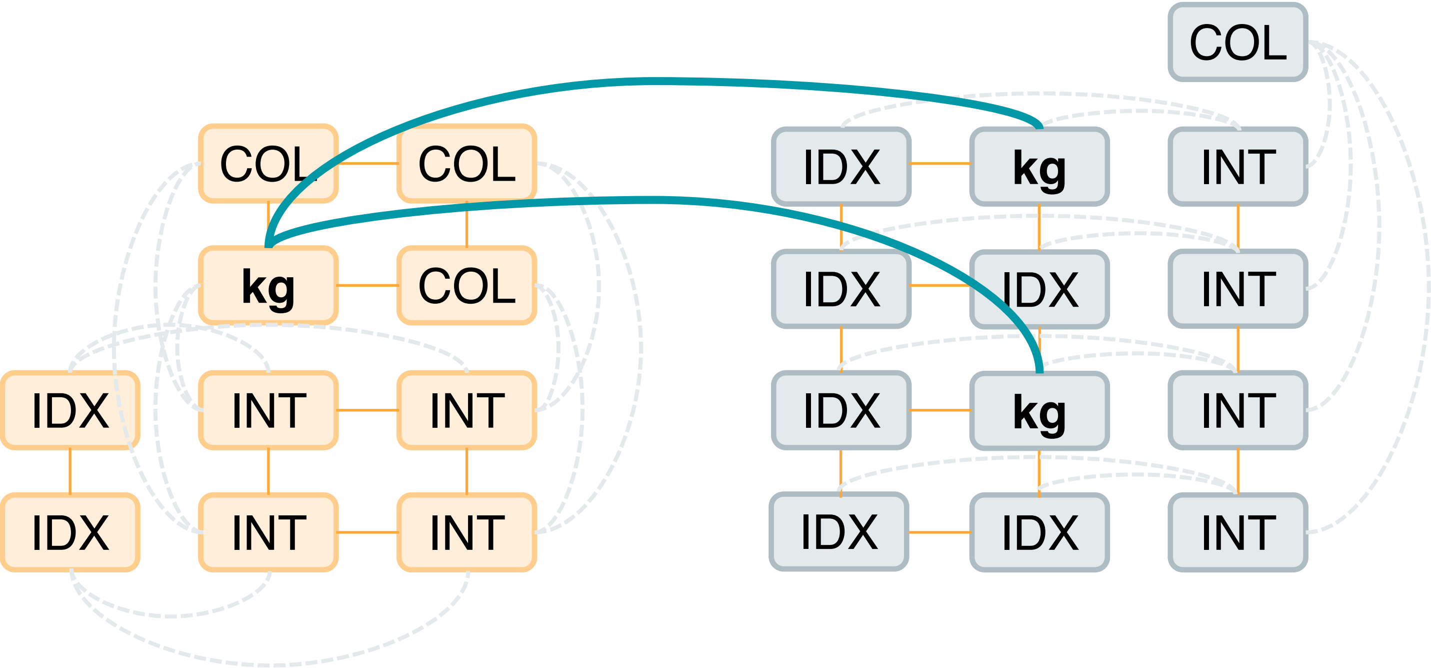
	weight	
	kg	lbs
cat	1	2
dog	2	4

©2018 RISELab

Encoding: *Equality* Edges

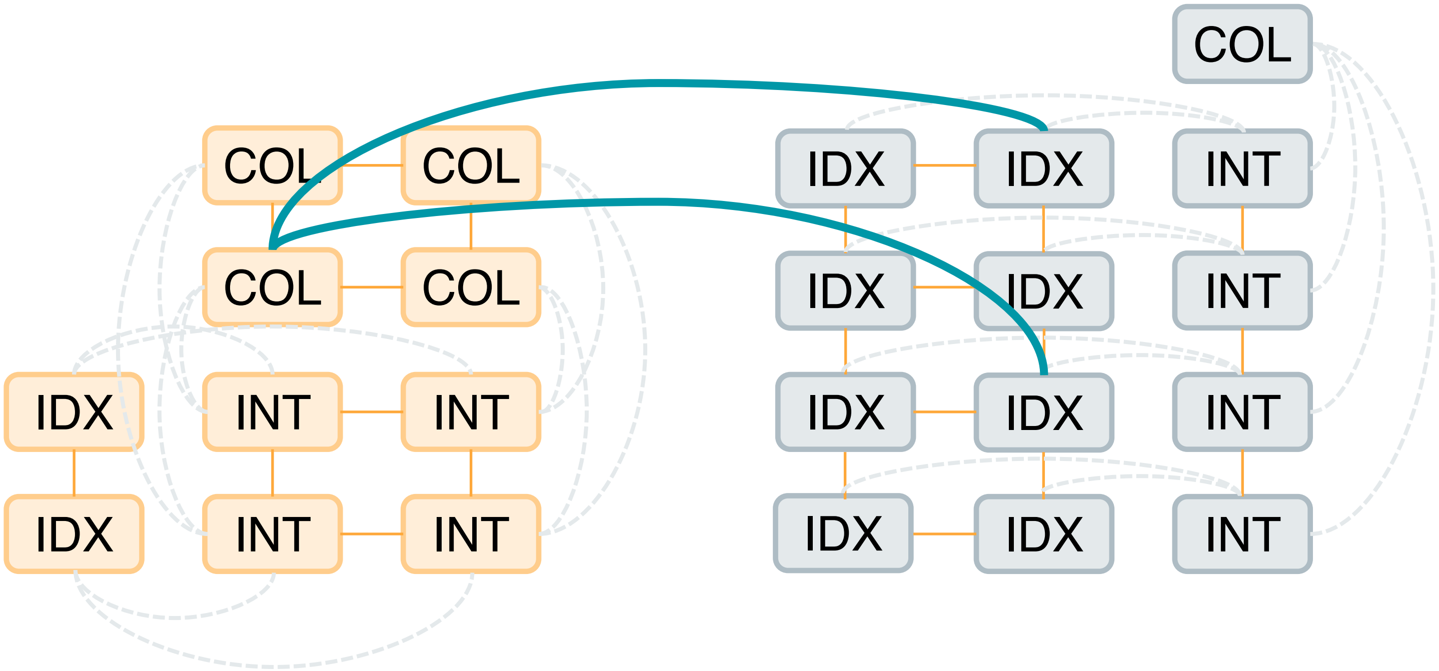


Encoding: *Equality* Edges

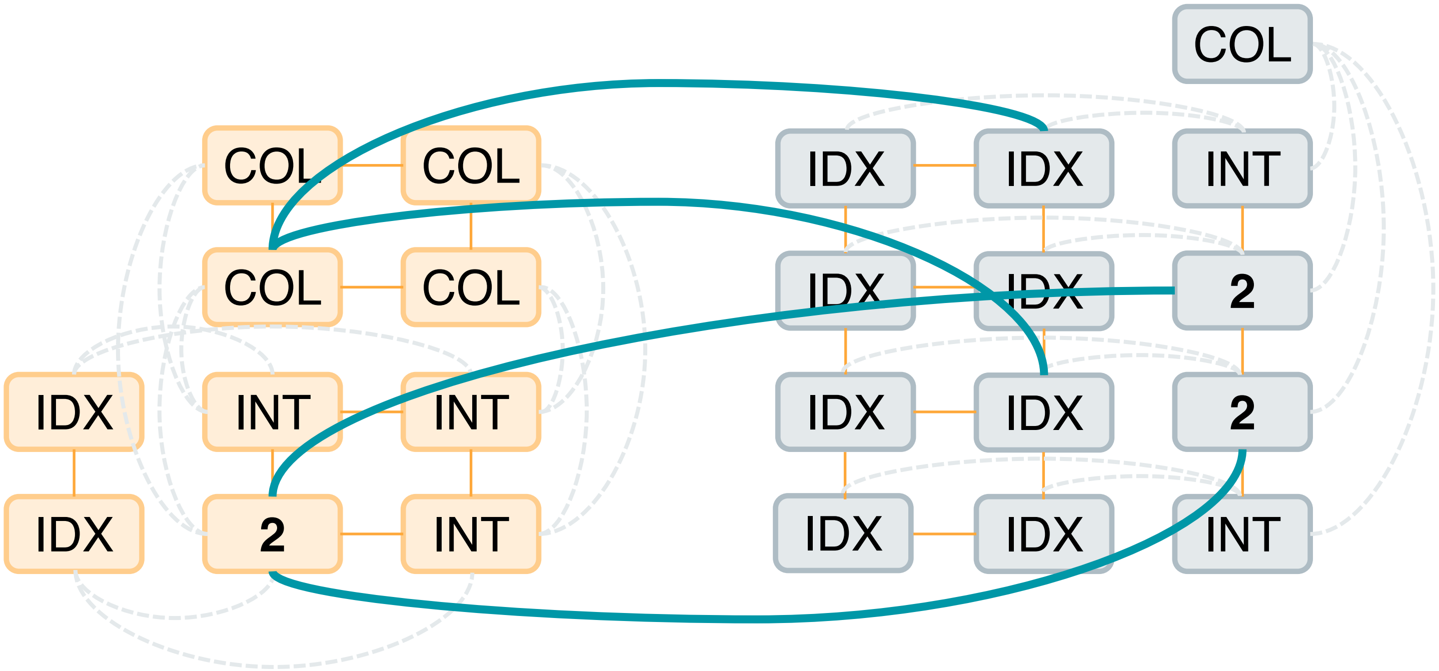


©2018 RISELab

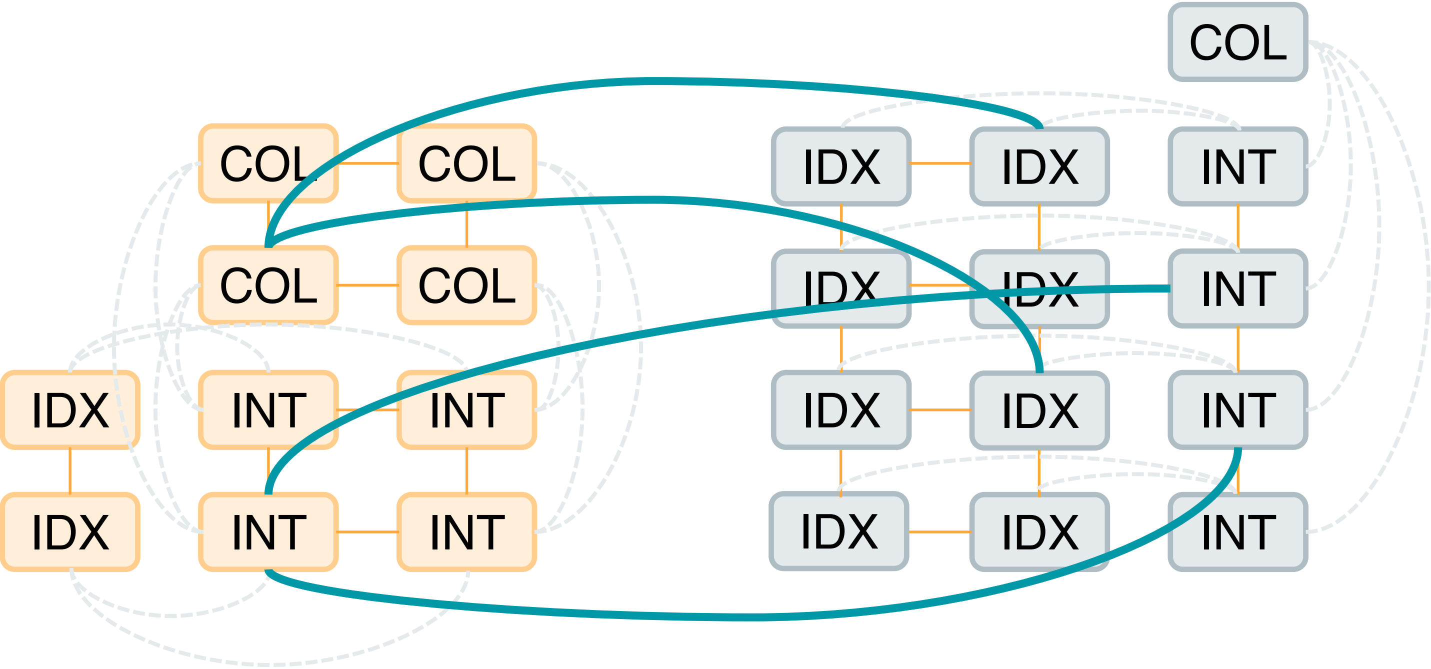
Encoding: *Equality* Edges



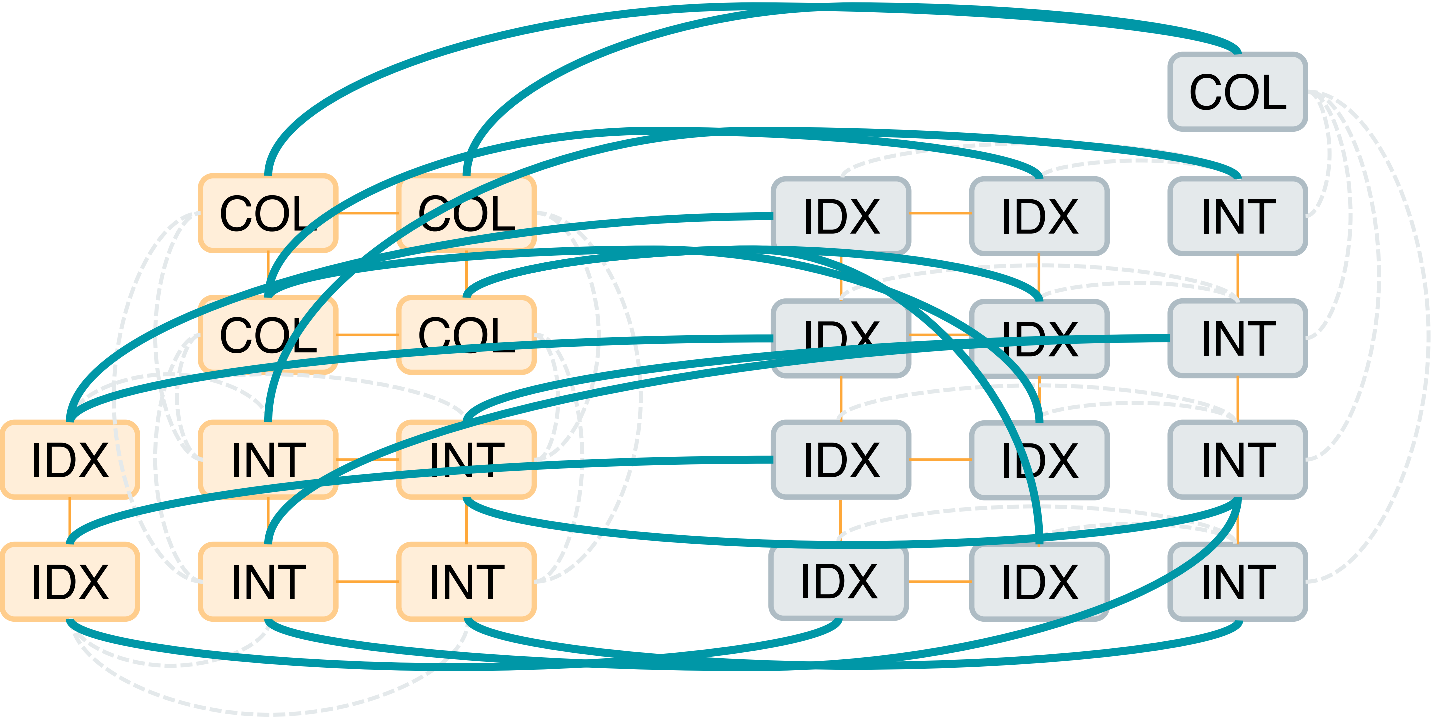
Encoding: *Equality* Edges



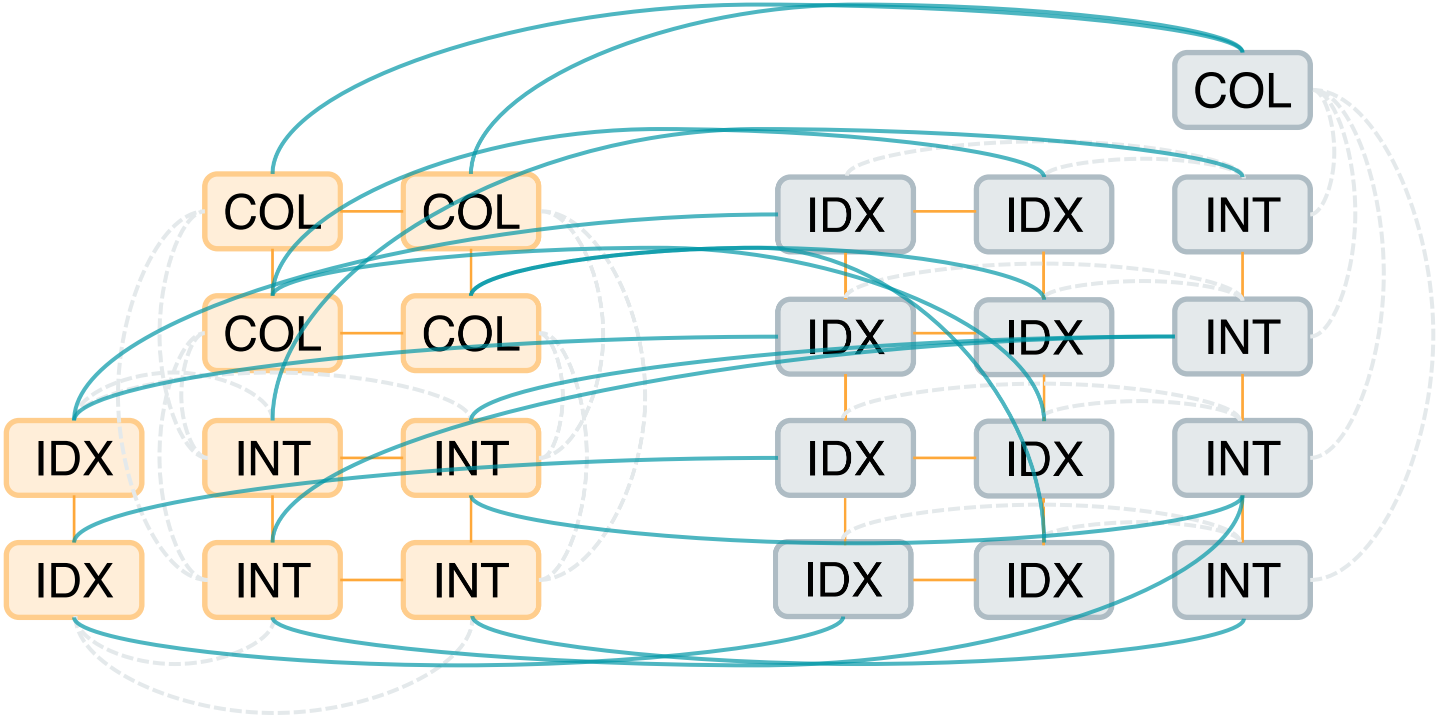
Encoding: *Equality* Edges



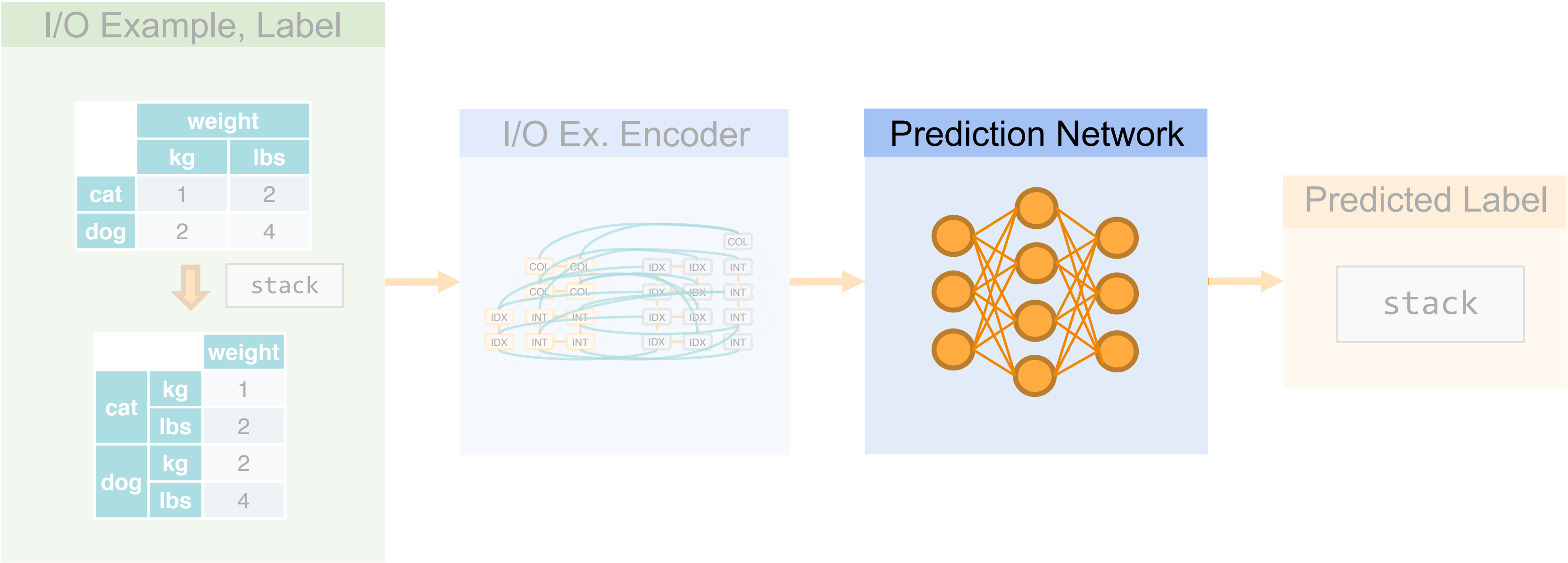
Encoding: *Equality* Edges



Final Encoding



Step 2: Predicting Function Label



I/O Example, Label

	weight	
	kg	lbs
cat	1	2
dog	2	4

stack

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

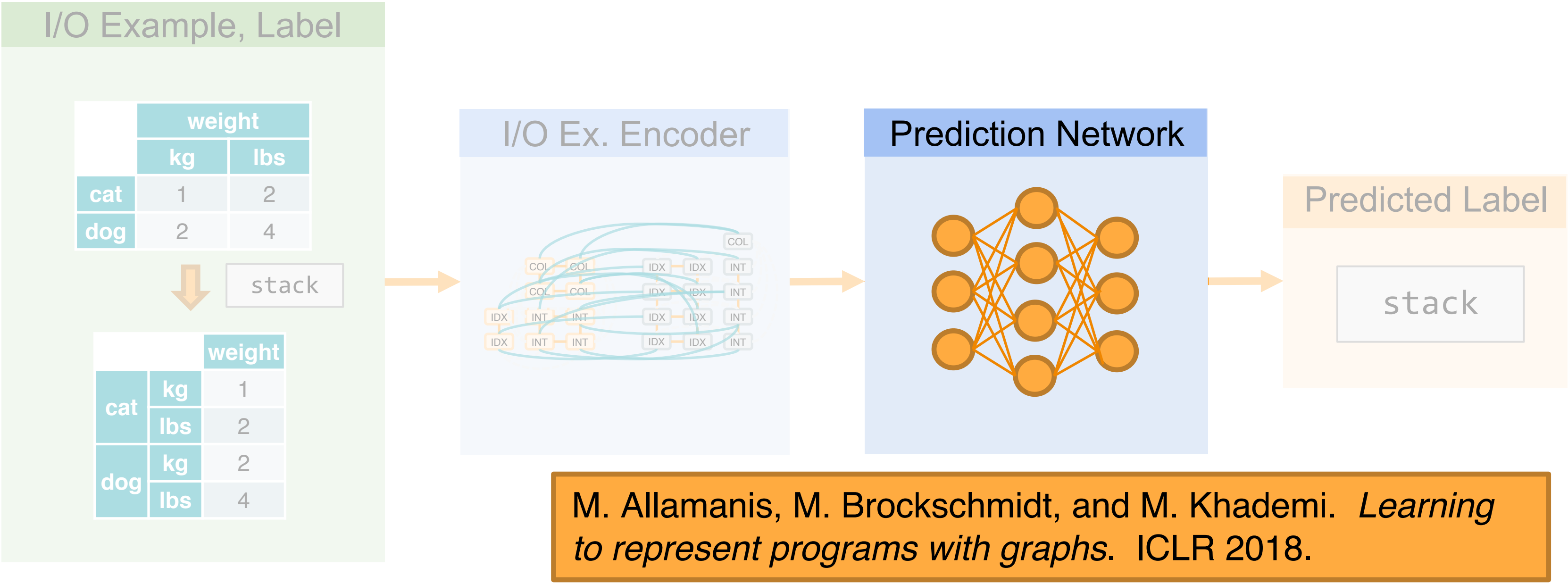
I/O Ex. Encoder

Prediction Network

Predicted Label

stack

Step 2: Predicting Function Label



I/O Example, Label

	weight	
	kg	lbs
cat	1	2
dog	2	4

```
stack
```

		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

I/O Ex. Encoder

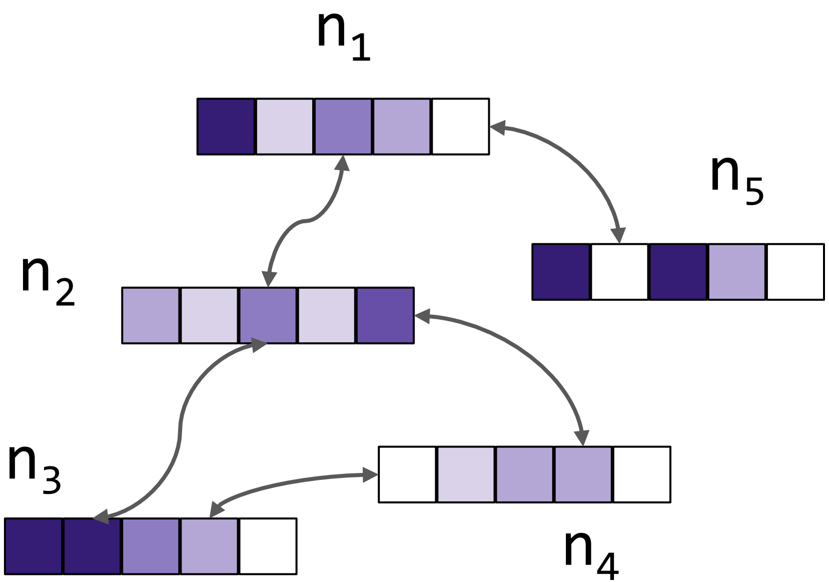
Prediction Network

Predicted Label

```
stack
```

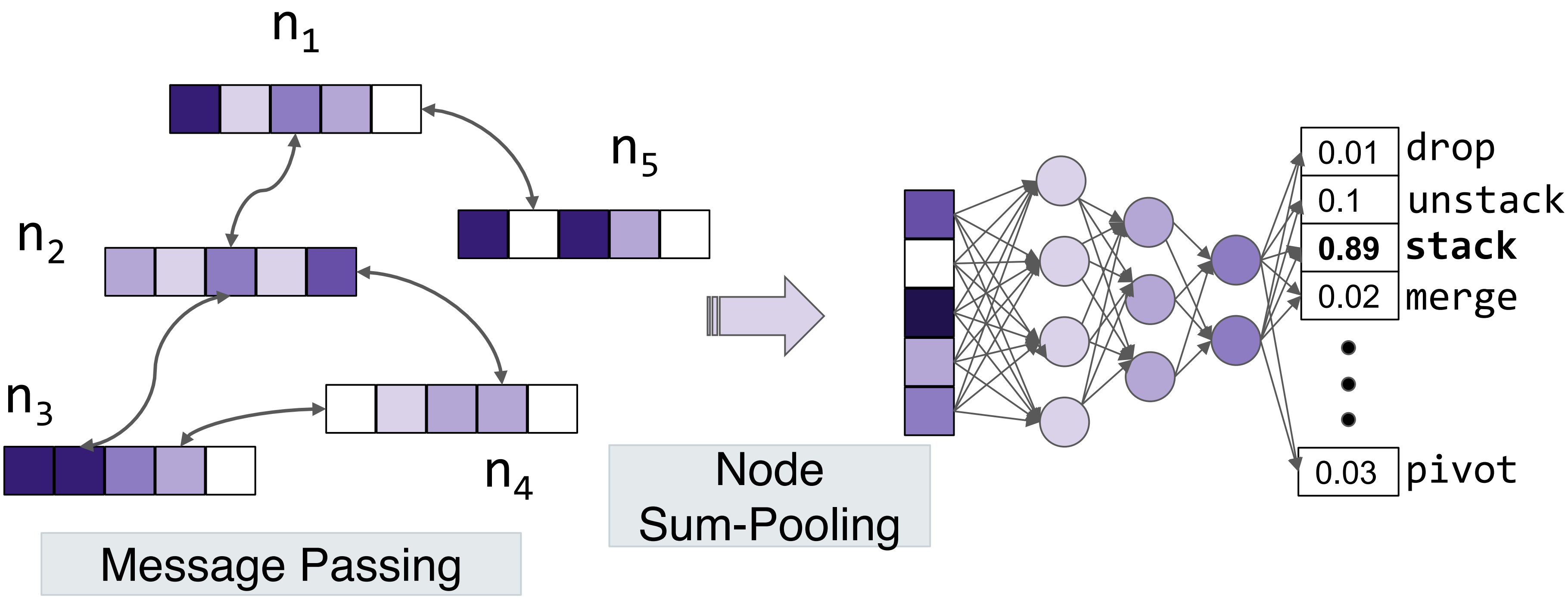
M. Allamanis, M. Brockschmidt, and M. Khademi. *Learning to represent programs with graphs*. ICLR 2018.

Graph Neural Network



Message Passing

Graph Neural Network



Message Passing

Node Sum-Pooling

Training the Predictor

- Training set: 1,000,000 random *<input, output, function>* tuples
- Validation set: 100,000 random *<input, output, function>* tuples
- Test set: 29 real-world examples (StackOverflow, pandas book)

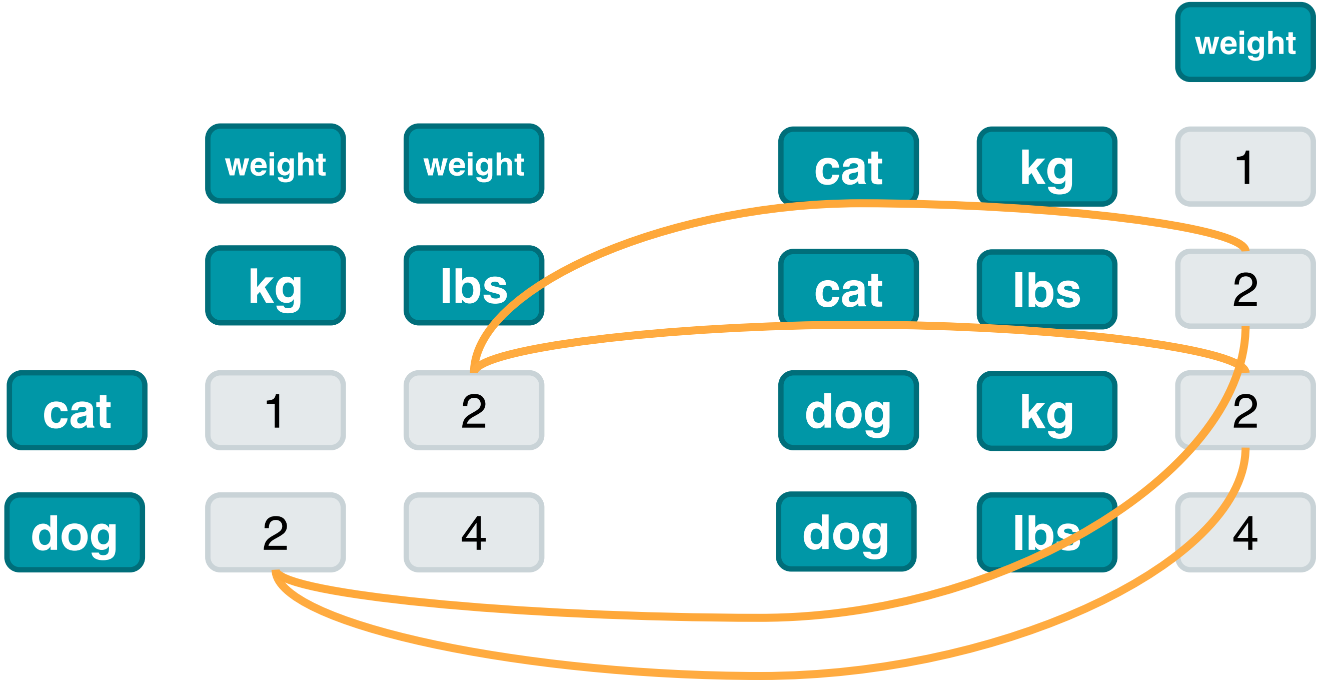
Accuracy Results

	Ground-Truth	
Suite	Top-1	Top-5
Validation	65%	94%
Test	59%	83%

Accuracy Results

	Ground-Truth		Success-Rate	
Suite	Top-1	Top-5	Top-1	Top-5
Validation	65%	94%	82%	97%
Test	59%	83%	69%	83%

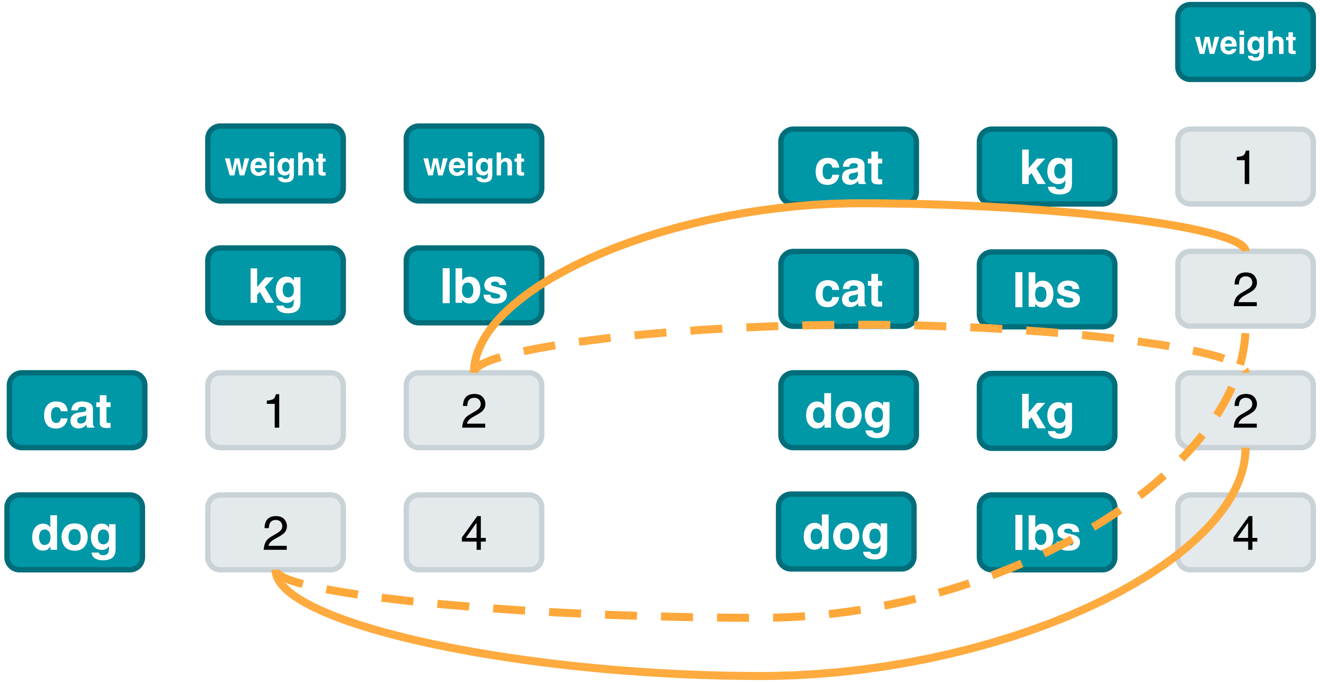
Cleaning: Removing Spurious Edges



	weight	weight
	kg	lbs
cat	1	2
dog	2	4

		weight
cat	kg	1
cat	lbs	2
dog	kg	2
dog	lbs	4

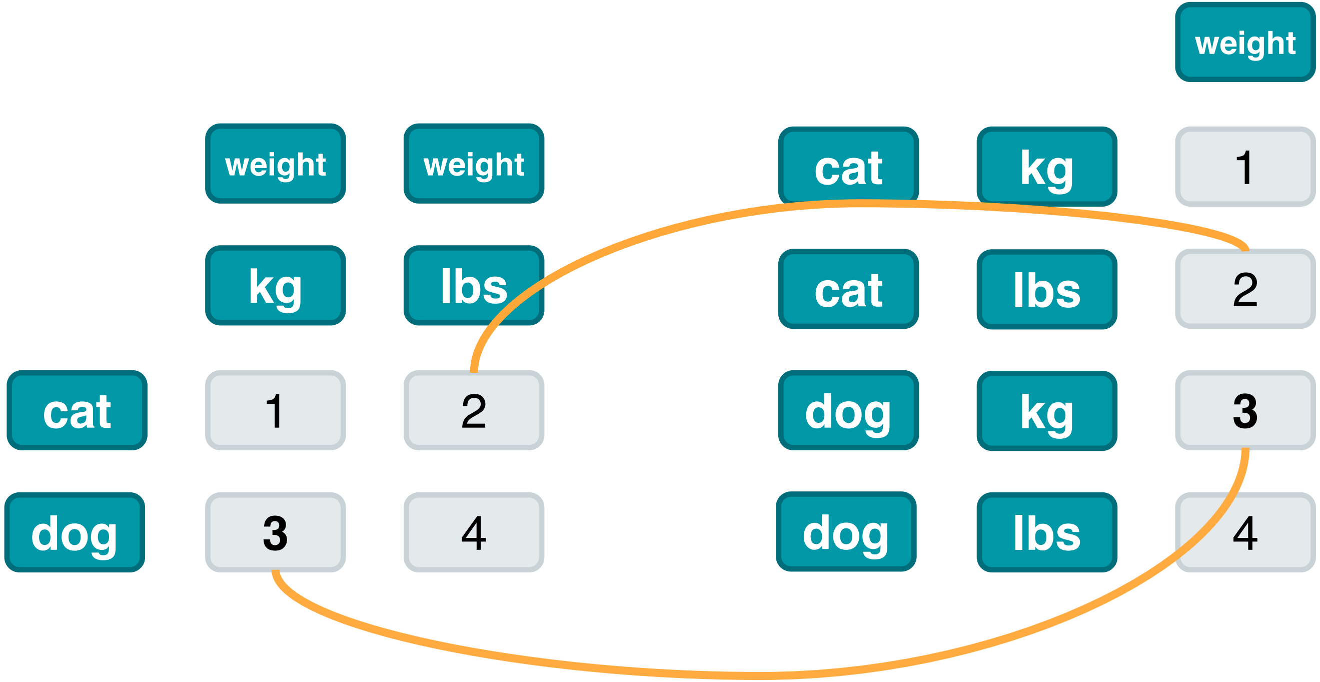
Cleaning: Removing Spurious Edges



	weight	weight
	kg	lbs
cat	1	2
dog	2	4

		weight
cat	kg	1
cat	lbs	2
dog	kg	2
dog	lbs	4

Cleaning: Removing Spurious Edges



	weight	weight
	kg	lbs
cat	1	2
dog	3	4

		weight
cat	kg	1
cat	lbs	2
dog	kg	3
dog	lbs	4

Accuracy Results

	Ground-Truth		Success-Rate	
Suite	Top-1	Top-5	Top-1	Top-5
Validation	65%	94%	82%	97%
Test	59%	83%	69%	83%
Cleaned Test	66%	97%	83%	97%

Accuracy Results

	Ground-Truth		Success-Rate	
Suite	Top-1	Top-5	Top-1	Top-5
Validation	65%	94%	82%	97%
Test	59%	83%	69%	83%
Cleaned Test	66%	97%	83%	97%

Moving Forward: Key Challenges

1. Accurately representing relationships (e.g. no spurious)
2. Semantically identical programs
3. Higher depths: sensible program generation

Moving Forward: Key Challenges

1. Accurately representing relationships (e.g. no spurious)
2. Semantically identical programs
3. Higher depths: sensible program generation

	Depth-2 Ground Truth		
Suite	Top-1	Top-5	Top-25
Validation	16.8%	43.5%	75.8%

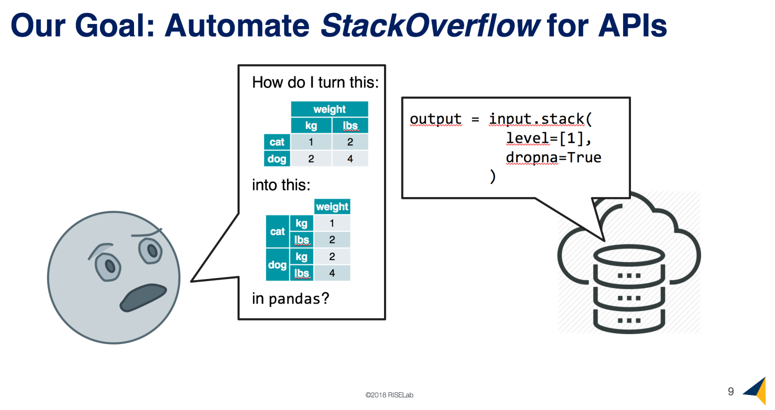
Moving Forward: Key Challenges

1. Accurately representing relationships (e.g. no spurious)
2. Semantically identical programs
3. Higher depths: sensible program generation

	Depth-2 Ground Truth		
Suite	Top-1	Top-5	Top-25
Validation	16.8%	43.5%	75.8%

```
v0 = input.stack()
v1 = input.eq(v0)
```

✗



Our Goal: Automate *StackOverflow* for APIs

How do I turn this:

	weight	
	kg	lbs
cat	1	2
dog	2	4

into this:

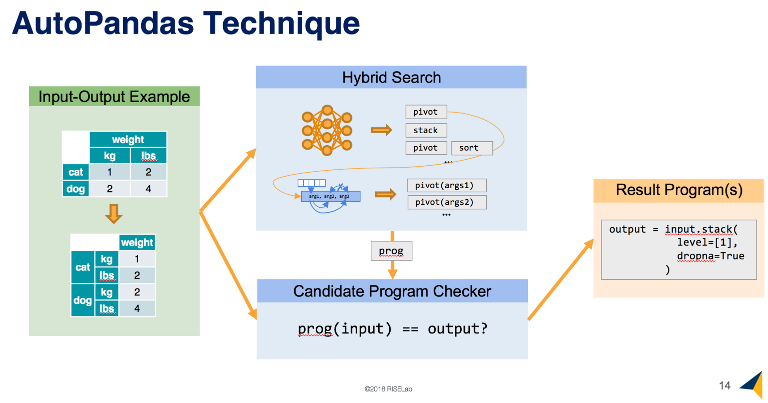
		weight
cat	kg	1
	lbs	2
dog	kg	2
	lbs	4

in pandas?

```
output = input.stack(
          level=[1],
          dropna=True
       )
```

©2018 RISELab

9



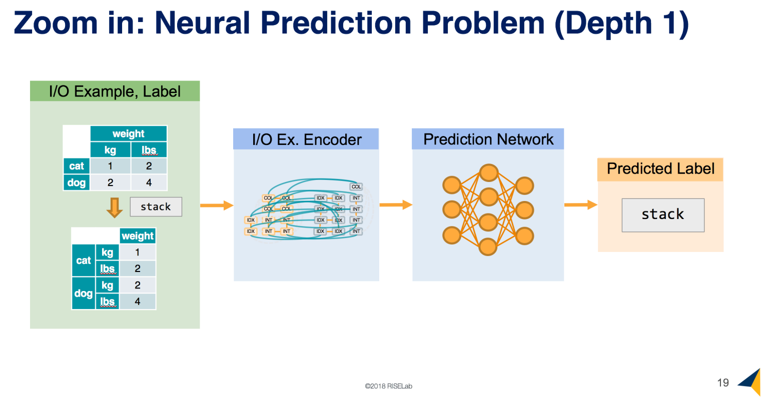
AutoPandas Technique

```
prog(input) == output?
```

```
output = input.stack(
          level=[1],
          dropna=True
       )
```

©2018 RISELab

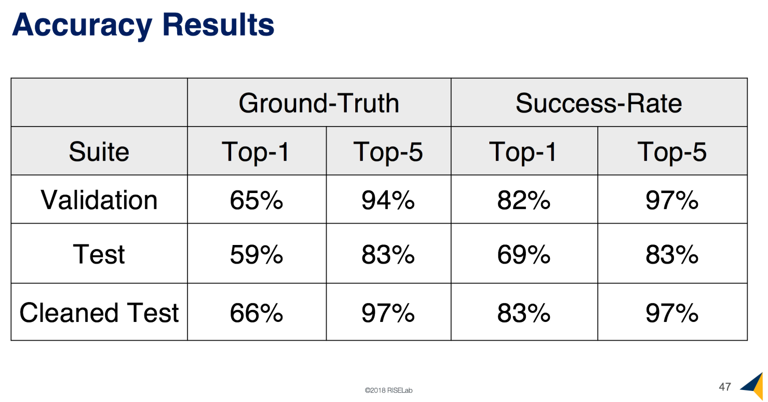
14



Zoom in: Neural Prediction Problem (Depth 1)

©2018 RISELab

19



Accuracy Results

	Ground-Truth		Success-Rate	
Suite	Top-1	Top-5	Top-1	Top-5
Validation	65%	94%	82%	97%
Test	59%	83%	69%	83%
Cleaned Test	66%	97%	83%	97%

©2018 RISELab

47

Caroline Lemieux, clemieux@berkeley.edu