Babble Labble:

Training Classifiers with Natural Language Explanations

Braden Hancock, Paroma Varma, Stephanie Wang, Martin Bringmann, Percy Liang, Chris Ré

ACL

17 July 2018

Melbourne, Australia

Machine learning can help you!***

***If you have enough training data

Traditional Labeling

Example

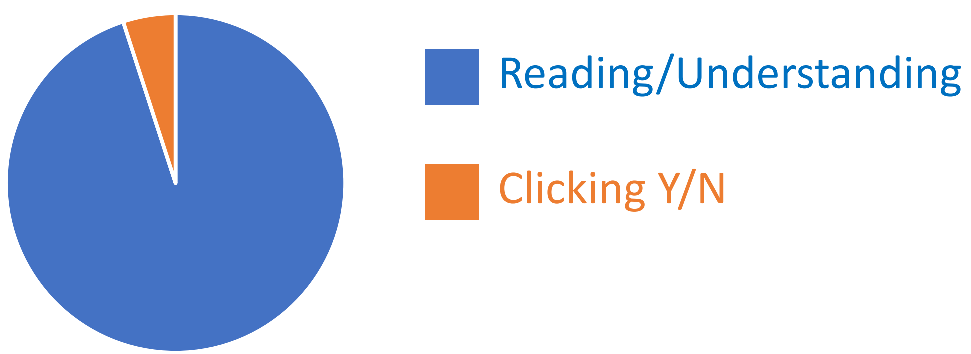
Tom Brady was spotted in New York City on Monday with his wife Gisele Bündchen amid rumors of Brady's alleged role in Deflategate.

Label

Is person 1 married to person 2?

Y N

Time Spent



Higher Bandwidth Supervision

Example

Tom Brady was spotted in New York City on Monday with his wife Gisele Bündchen amid rumors of Brady's alleged role in Deflategate.

Label

Is person 1 married to person 2?

Y N

Explanation

Why do you think so?

Because the words "his wife" are right before person 2.

Explanations Encode Labeling Heuristics

Explanation

Why did you label True?

Because the words "his wife" are right before person 2.

Label	Example
True	"Barack batted back tears as he thanked **his wife**, Michelle, for all her help."
True	"Both Bill and **his wife** Hillary smiled and waved at reporters as they rode by."
True	"George attended the event with **his wife**, Laura, and their two daughters."

Big Idea: Instead of collecting labels, collect labeling heuristics (in the form of explanations) that can be used to label more examples for free.

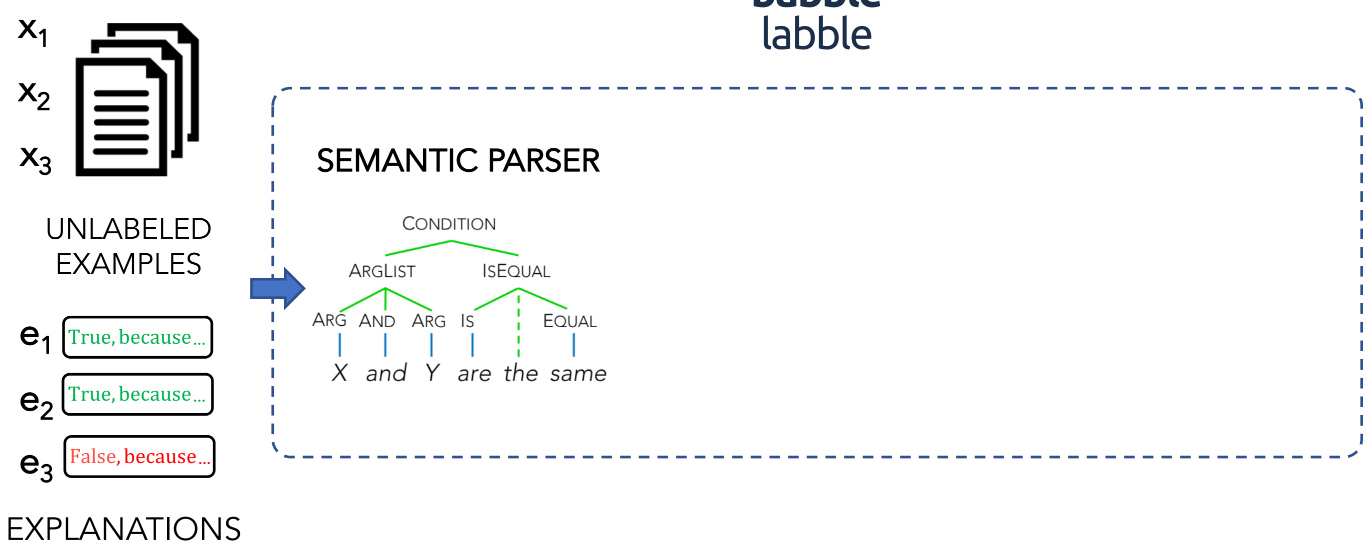
babble labble

A framework for generating large training sets from natural language explanations and unlabeled data

Result: classifiers trained with Babble Labble and explanations achieved the same F1 score as ones trained with traditional labels while requiring **5–100x** fewer user inputs

Babble Labble Framework

babble
labble



Explanations Encode Heuristics

Explanation

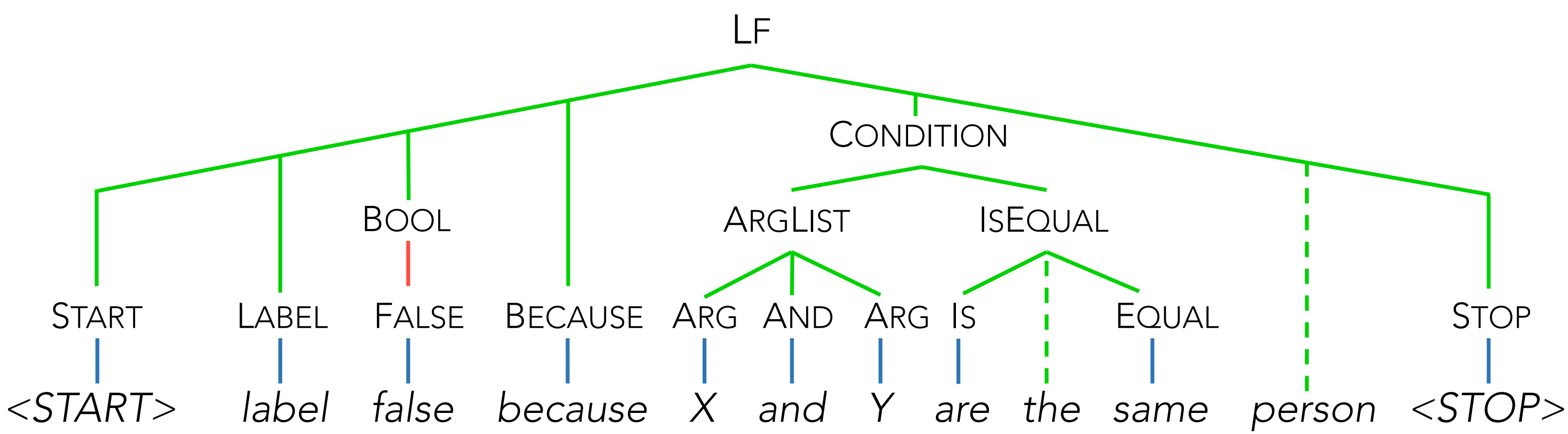
Why did you label True?

Because the words “his wife” are right before person 2.

Labeling Function

```
def f(x):
  return 1 if (“his wife” in left(x.person2, dist==1))
     else 0 #abstain
```

Semantic Parser



Lexical Rules		Unary Rules		Compositional Rules		Ignored token
<START>	→ START	FALSE	→ BOOL	START LABEL BOOL BECAUSE CONDITION STOP	→ LF	
label	→ LABEL	TRUE	→ BOOL	ARGLIST ISEQUAL	→ CONDITION	
false	→ FALSE	INT	→ NUM	ARG AND ARG	→ ARGLIST	

Labeling Function Template:

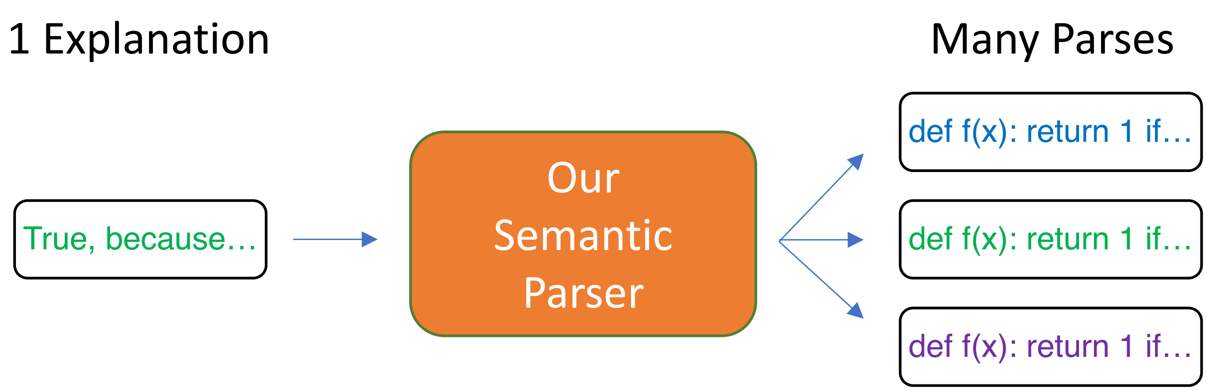
```
def LF(x):
  return [label] if [condition] else [abstain]
```

Predicates

	Predicate	Description
Logic & Comparison	`bool, string, int, float, tuple, list, set`	Standard primitive data types
	`and, or, not, any, all, none`	Standard logic operators
	$=, \neq, <, \leq, >, \geq$	Standard comparison operators
String Matching	`lower, upper, capital, all_caps`	Return True for strings of the corresponding case
	`starts_with, ends_with, substring`	Return True if the first string starts/ends with or contains the second
NER Tags	`person, location, date, number, organization`	Return True if a string has the corresponding NER tag
Sets & Mapping	`alias`	A frequently used list of words may be predefined and referred to with an alias
	`count, contains, intersection`	Operators for checking size, membership, or common elements of a `list/set`
	`map, filter`	Apply a functional primitive to each member of `list/set` to transform or filter the elements
Relative Positioning	`word_distance, character_distance`	Return the distance between two strings by words or characters
	`left, right, between, within`	Return as a string the text that is left/right/within some distance of a string or between two designated strings

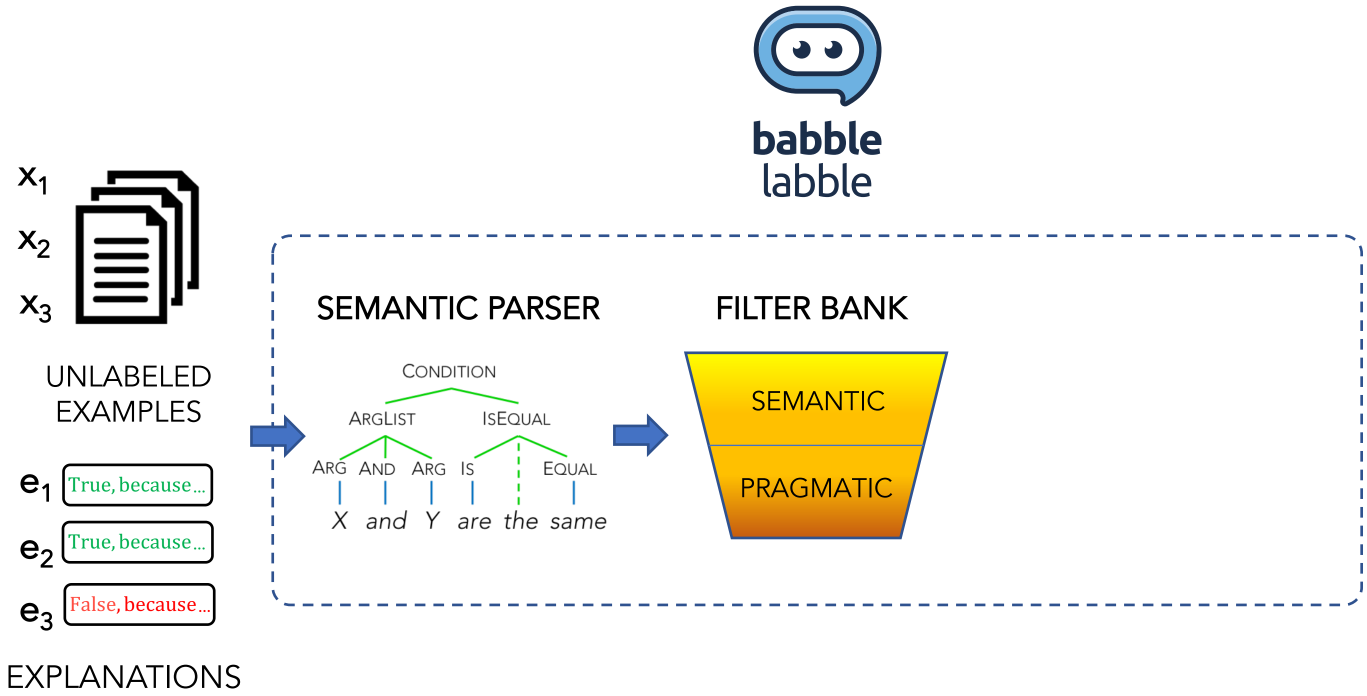
Semantic Parser I/O

Goal: produce the *correct* parse

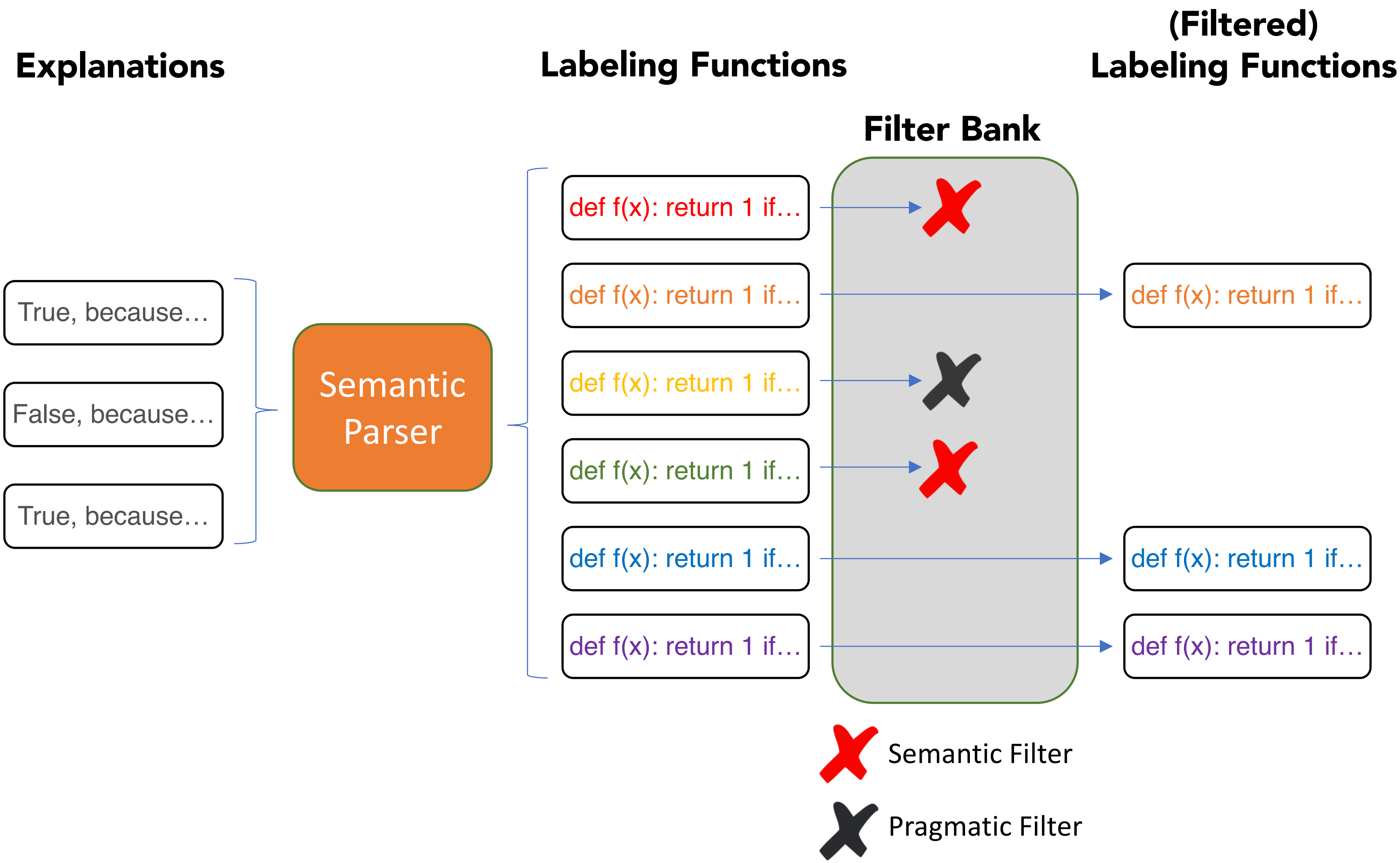


Goal: produce *useful* parses
(whether they're correct or not)

Babble Labble Framework



Filter Bank


Explanations
Labeling Functions
(Filtered) Labeling Functions
Filter Bank
True, because…
False, because…
True, because…
Semantic Parser
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
def f(x): return 1 if…
Semantic Filter
Pragmatic Filter

Semantic Filter

Example

x1: Tom Brady was spotted in New York City on Monday with his wife Gisele Bündchen amid rumors of Brady's alleged role in Deflategate.

Explanation

```
True, because the words “his wife”
are right before person 2.
```

Candidate Labeling Functions

“right before” = “to the right of”

```
def LF_1b(x):
  return (1 if “his wife” in
    right(x.person2) else 0
```

`LF_1b(x1) == 0` ✗

(“his wife” is not to the right of person 2)

“right before” = “immediately before”

```
def LF_1a(x):
  return (1 if “his wife” in
  left(x.person2, dist==1) else 0)
```

`LF_1a(x1) == 1` ✓

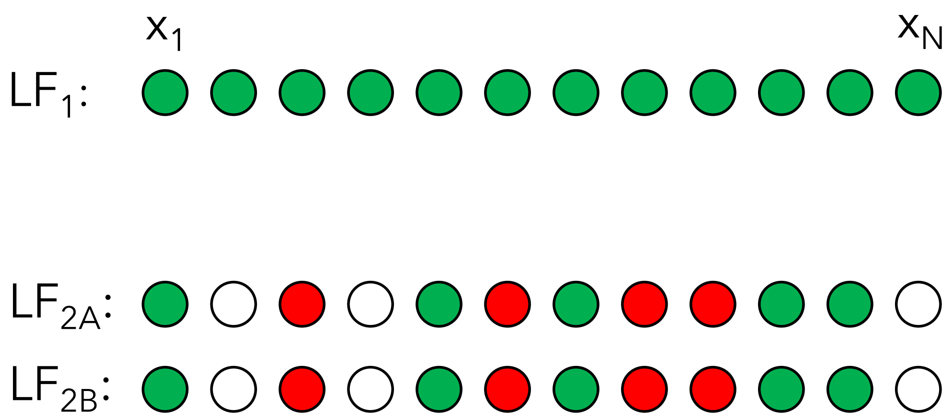
(“his wife” is, in fact, 1 word to the left of person 2)

Pragmatic Filters

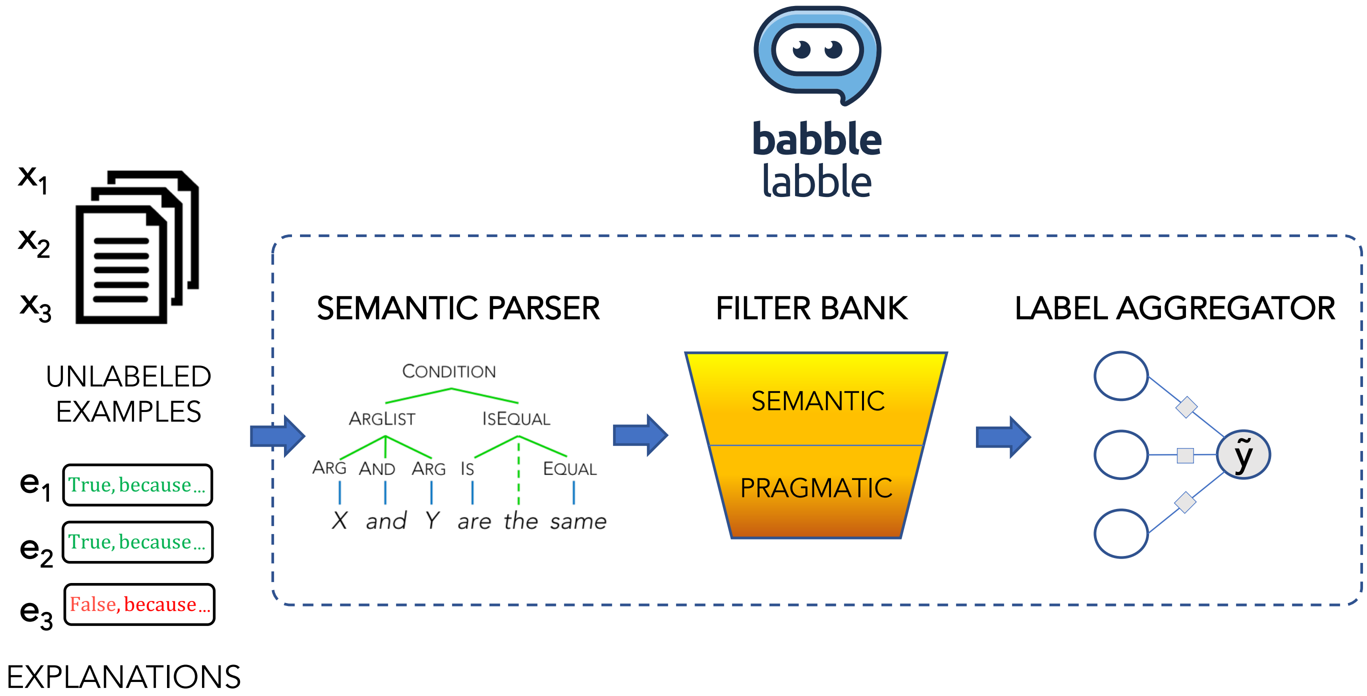
How does the LF label our unlabeled data?

Uniform labeling signature

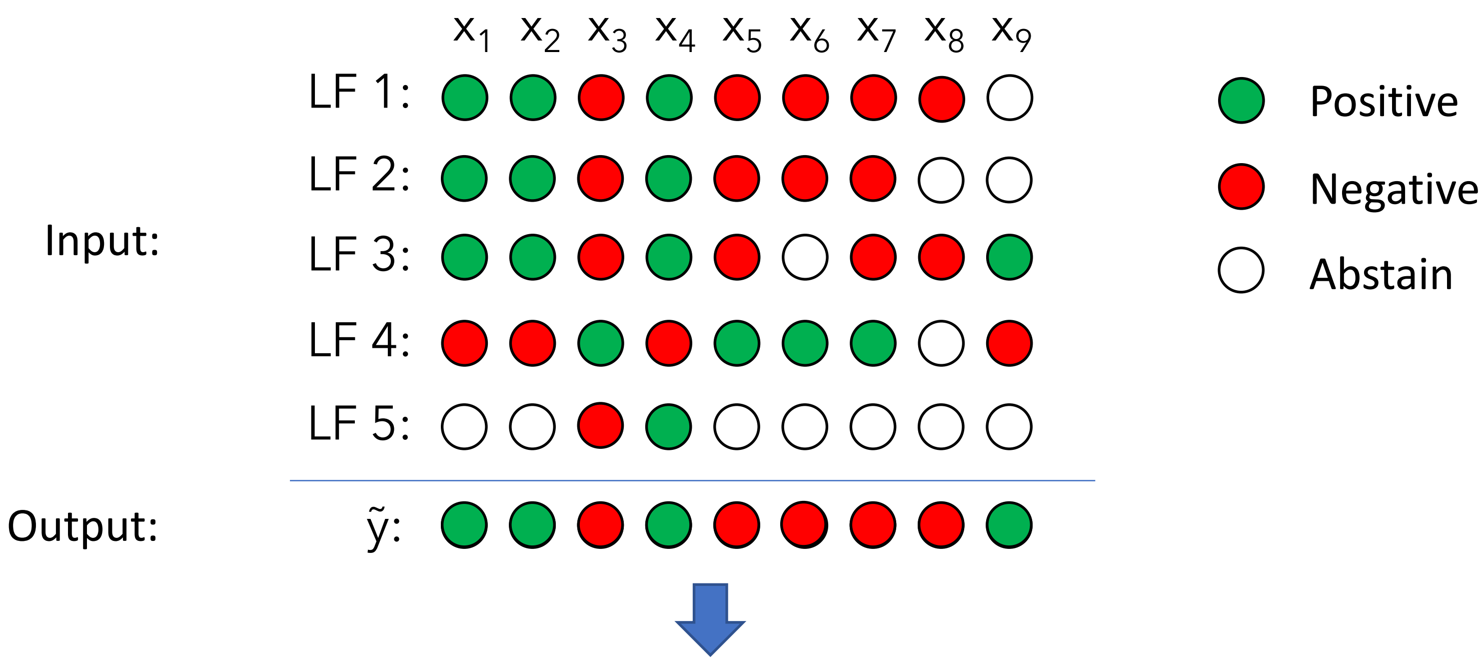
Duplicate labeling signature



Babble Labble Framework



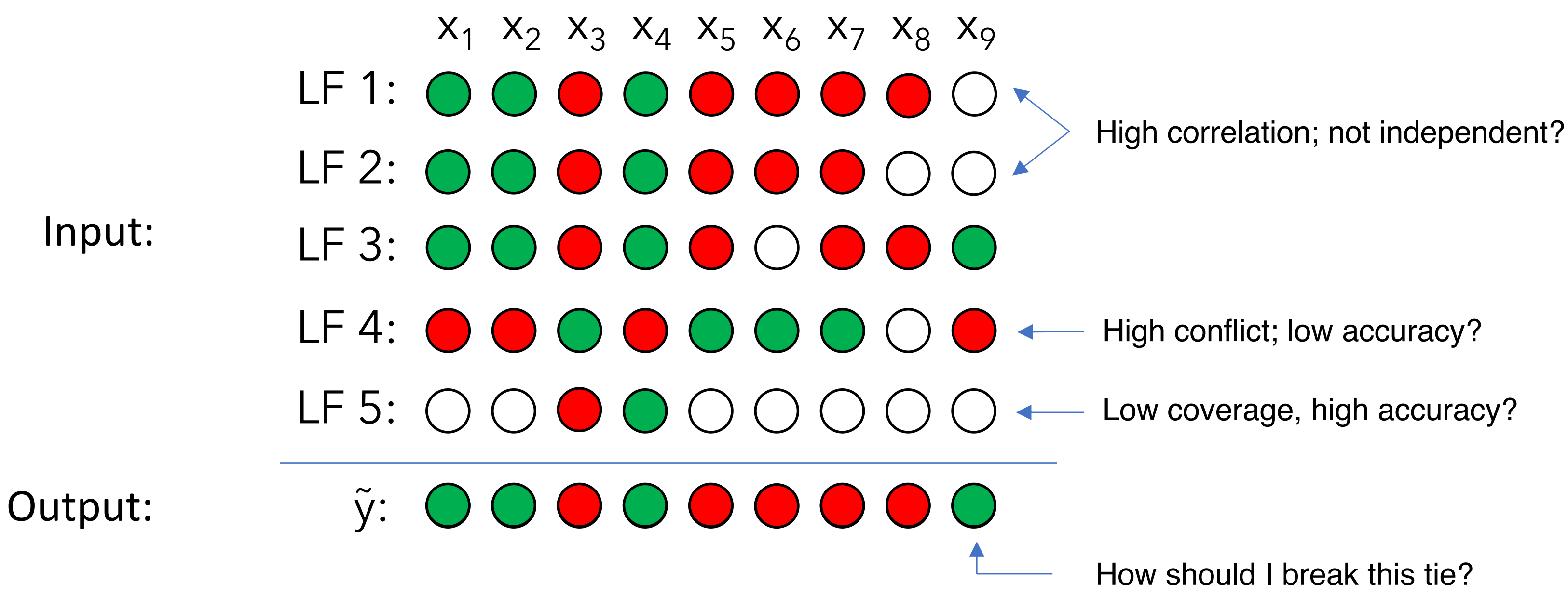
Label Aggregator



Training Data

$(x_1, \tilde{y}_1)$
$(x_2, \tilde{y}_2)$
$(x_3, \tilde{y}_3)$
$(x_4, \tilde{y}_4)$

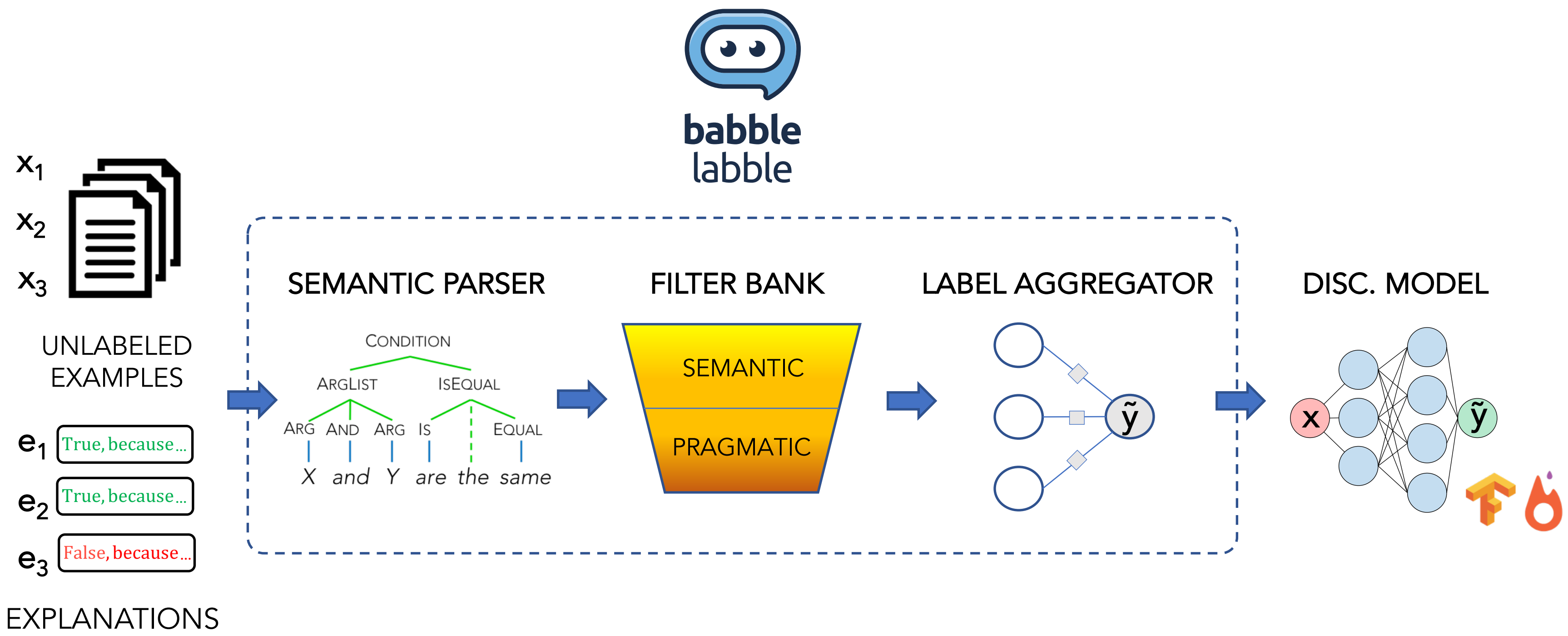
Label Aggregator



Data Programming:
(Ratner, et al. NIPS 2016)

As implemented in:
`snorkel.stanford.edu`

Babble Labble Framework

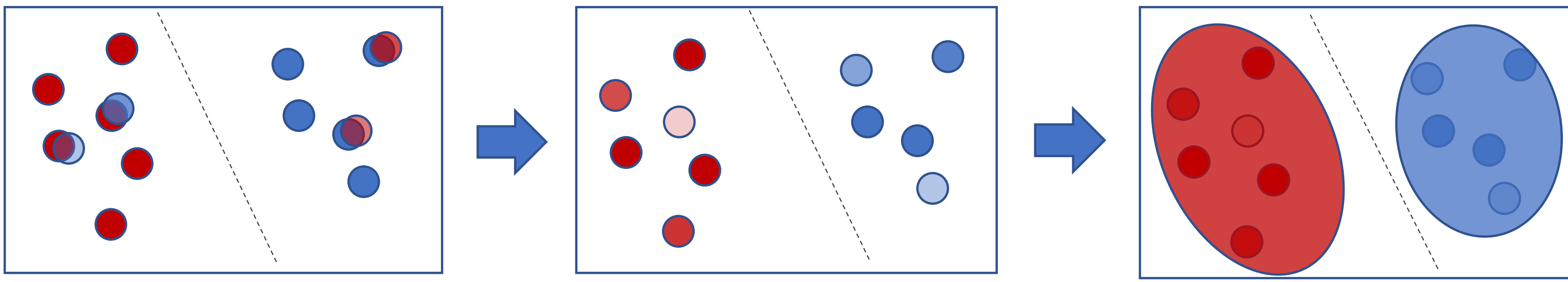


Discriminative Classifier

Input: Labeling Functions, *Unlabeled data*

Label Aggregator

Discriminative Model



Labeling functions generate noisy, conflicting votes

Resolve conflicts, re-weight & combine

Generalize beyond the labeling functions

Generalization

Task: identify disease-causing chemicals

Keywords mentioned in LFs:

"treats", "causes", "induces", "prevents", …

Highly relevant features learned by discriminative model:

"could produce a", "support diagnosis of", …

Training a discriminative model that can take advantage of additional useful features not specified in labeling functions boosted performance by **4.3 F1** points on average (10%).

Datasets

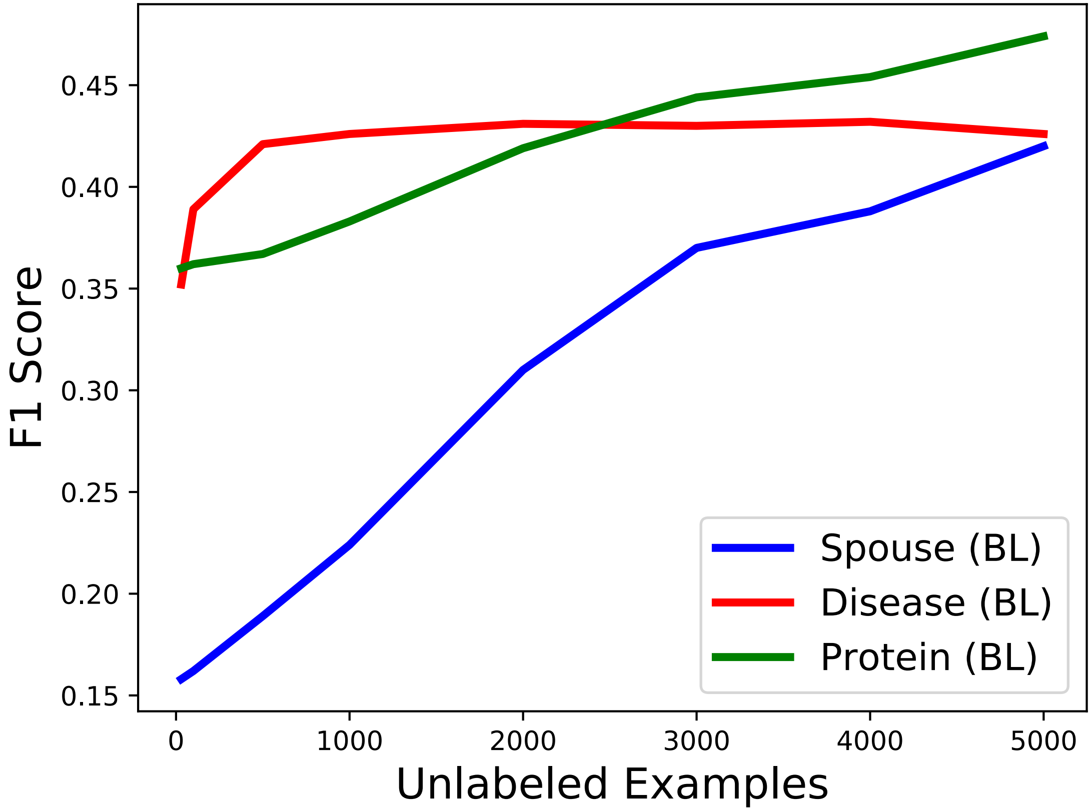
Name	# Unlabeled	Sample Explanations
Spouse	22k	Label true because "and" occurs between X and Y and "marriage" occurs one word after person1.
Disease	6.7k	Label true because the disease is immediately after the chemical and "induc" or "assoc" is in the chemical name.
Protein	5.5k	Label true because "Ser" or "Tyr" are within 10 characters of the protein.

Results

Task	F1 Score	Babble Labble # Explanations	Traditional Labels # Labels	Reduction in User Inputs
`Spouse`	50.1	30	3000+	**100x**
`Disease`	42.3	30	1000+	**33x**
`Protein`	47.3	30	150+	**5x**

Classifiers trained with Babble Labble and explanations achieved the same F1 score as ones trained with traditional labels while requiring **5–100x** fewer user inputs

Utilizing Unlabeled Data



With labeling functions, training set size (and often performance) scales with the amount of *unlabeled* data we have.

Filter Bank Effectiveness

Task	Babble Labble (No Filters)	Babble Labble	% Incorrected Parses Filtered
Spouse	15.7	50.1	97.8%
Disease	39.8	42.3	96.0%
Protein	38.2	47.3	97.0%
AVERAGE	**31.2**	**46.6**	**96.9%**

The filters removed almost **97%** of incorrect parses.
Without the filters removing bad parses, F1 drops by **15 F1** points on average.

Perfect Parsers Need Not Apply

Task	Babble Labble	Babble Labble (Perfect Parses)
Spouse	50.1	49.8
Disease	42.3	43.2
Protein	47.3	46.8
AVERAGE	**46.6**	**46.8**

Using perfect parses yielded negligible improvements.
In this framework, for this task, a naïve semantic parser is good enough!

Limitations

"Alice beat Bob in the annual office pool tournament."

Prefers
High-level
(e.g., "it says so")

Do you think person 1 is the spouse of person 2? Why?

No, because it sounds like they're just co-workers.

What's a co-worker?

Prefers
Low-level
(e.g., keywords, word distance, capitalization, etc.)

Users' reasons for labeling are sometimes high-level concepts that are hard to parse.

Related Work: Data Programming

Common theme:

Use weak supervision (e.g., labeling functions) to generate training sets

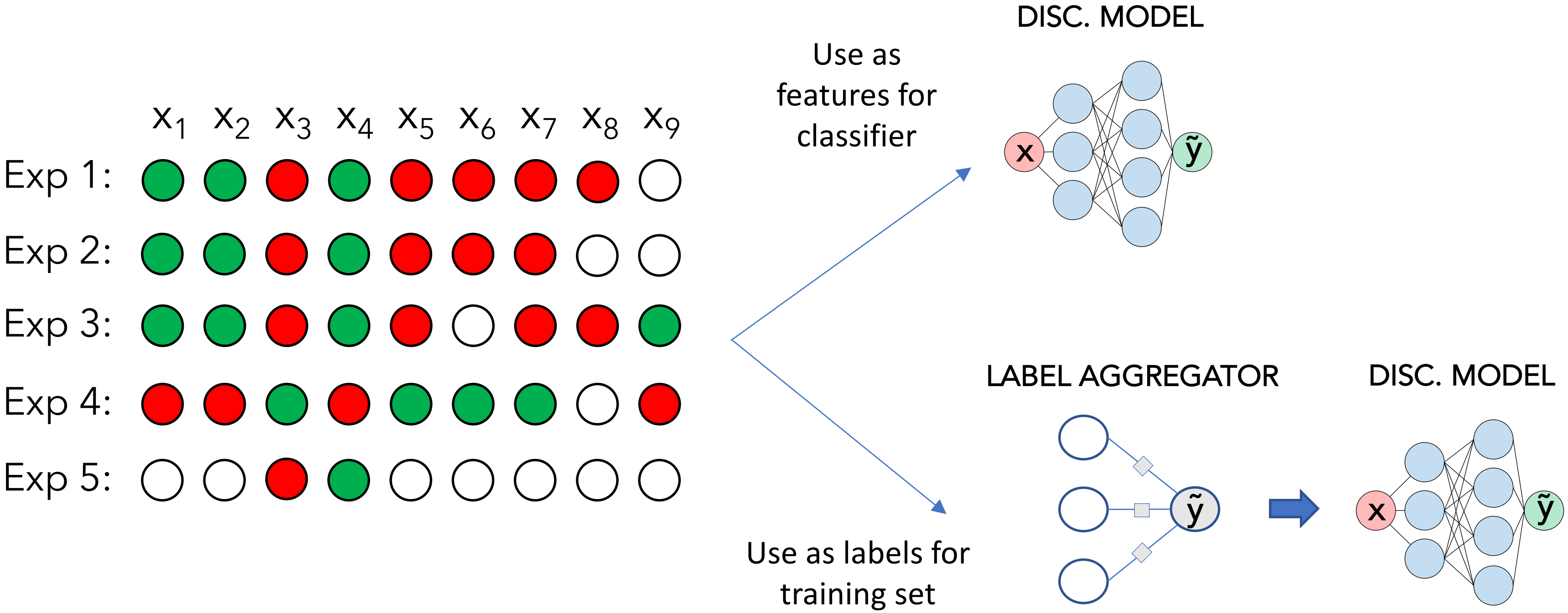
- **Snorkel** (Ratner et al., VLDB 2018)
  - Flagship platform for dataset creation from weak supervision
- **Structure Learning** (Bach et al., ICML 2017)
  - Learning dependencies between correlated labeling functions
- **Reef** (Varma and Ré, In Submission)
  - Auto-generating labeling functions from a small labeled set

`snorkel.stanford.edu`

Related Work: Explanations as Features

(Srivastava et al., 2017)

What if we use our explanations to make features instead of training labels?



Using the parses to label training data instead of as features boosts **4.5 F1** points.

Related Work: Highlighting

Highlight key phrases in text:

(Zaidan and Eisner, 2008), (Arora and Nyberg, 2009)

Mark key regions in images:

(Ahn et al., 2006)

Label key features directly:

(Druck et al., 2009), (Raghavan et al., 2005), (Liang et al., 2009)

Tom Brady was spotted in New York City on Monday with his wife **Gisele Bündchen** amid rumors of Brady's alleged role in Deflategate.

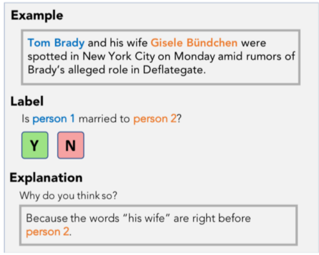
Benefits of natural language approach:

- more options: e.g., "X is **not** in the sentence", "X **or** Y is in the sentence"
- more direct credit assignment (compared to highlighting)
- no feature set required a priori

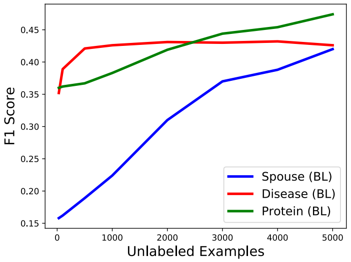
Summary

We need more efficient ways to collect supervision

We can collect labeling heuristics instead of labels



Using this approach, training set size grows with the amount of *unlabeled* data we have



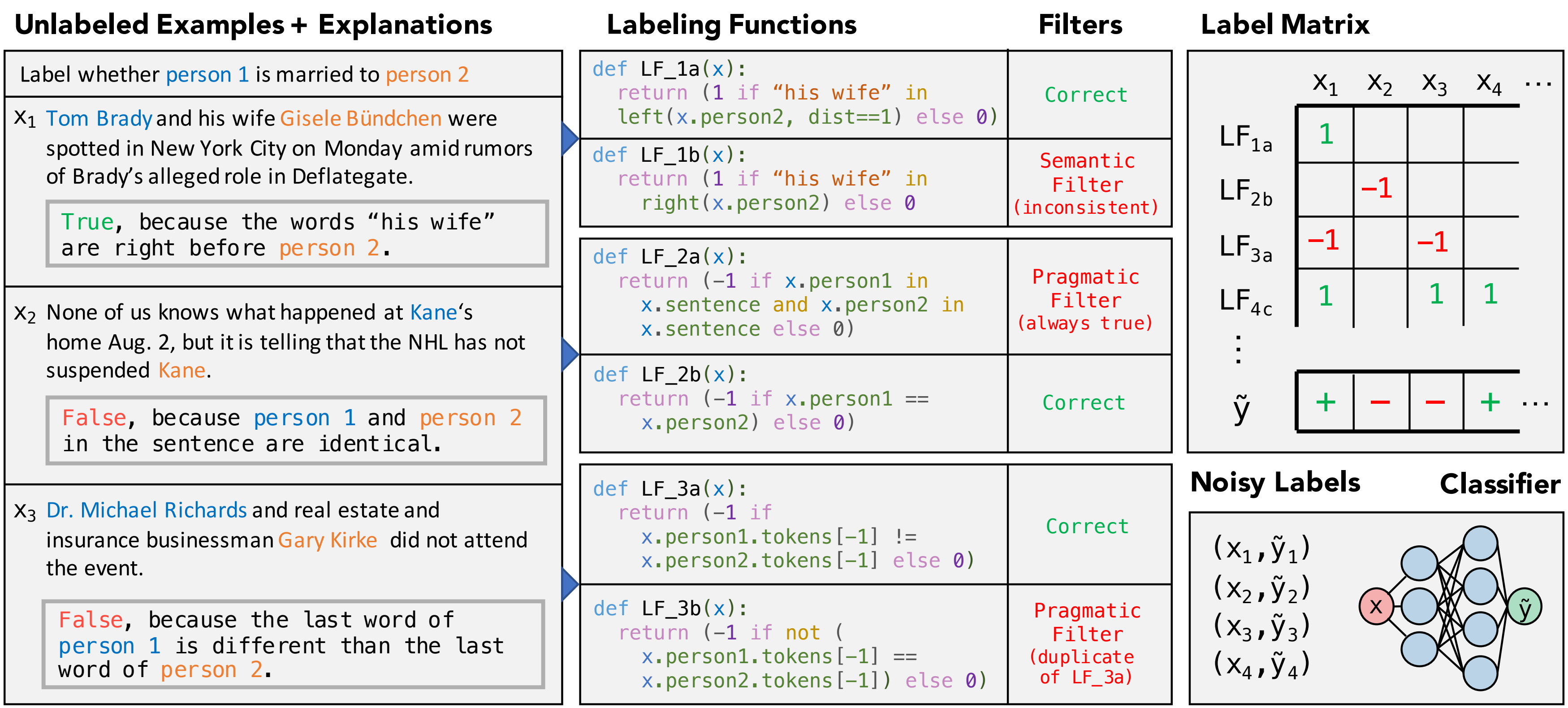
https://github.com/HazyResearch/babble

EXTRA SLIDES

Dataset Statistics

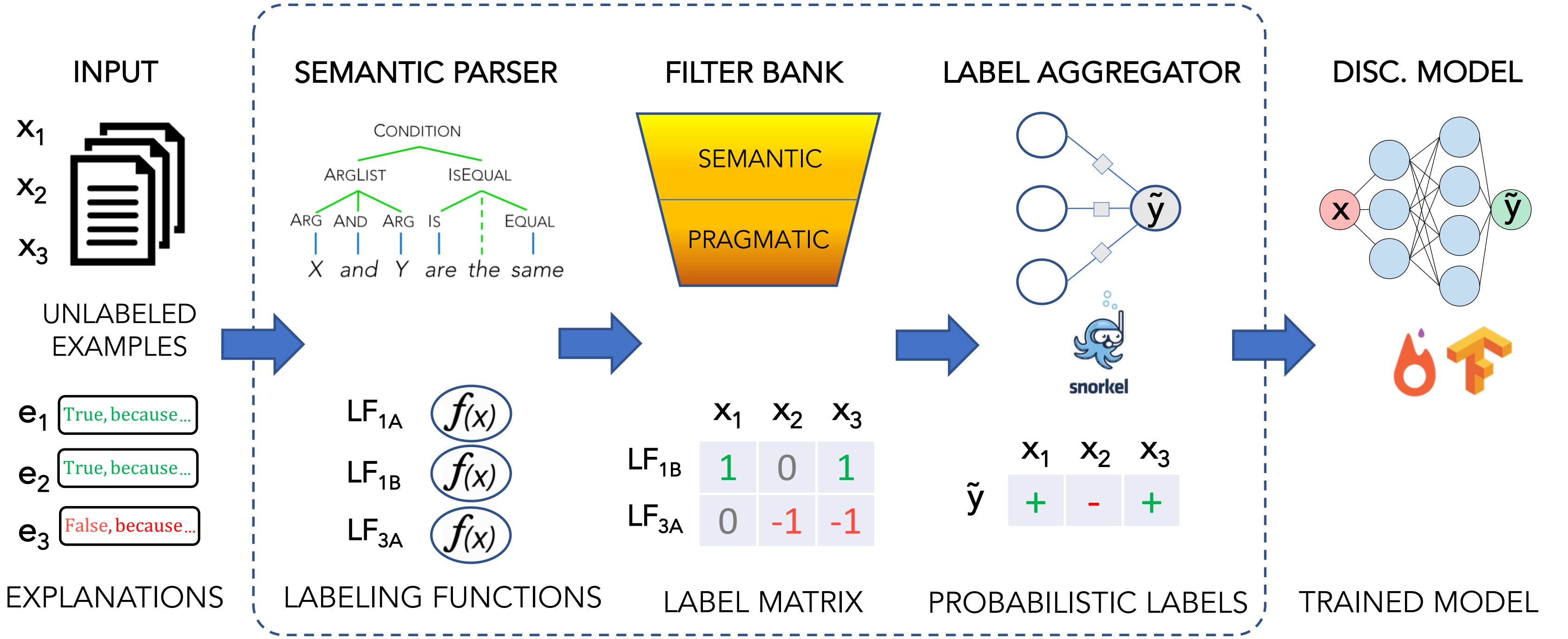
Task	Train	Dev	Test	% Pos.
Spouse	22195	2796	2697	8%
Disease	6667	773	4101	20%
Protein	5546	1011	1058	22%

Babble Labble Framework



Babble Labble Framework

babble labble



IMPORTANT: No Babble Labble components require no labeled training data!

Babble Labble

Example

Tom Brady was spotted in New York City on Monday with his wife Gisele Bündchen amid rumors of Brady's alleged role in Deflategate.

Label

Is person 1 married to person 2?

Y N

Explanation

Why do you think so?

Because the words "his wife" are right before person 2.

Labeling Function

```
def LF1(x):
  return (1 if "his wife" in left(x.person2, dist==1)
     else 0)
```

Label Matrix

	x_1	x_2	x_3	x_4	…
LF_1	1				
LF_2		−1			
LF_3	−1		−1		
LF_4	1		1	1	
⋮					

Aggregated Labels

$\tilde{y}$	+	−	−	+	…

Classifier

$(x_1, \tilde{y}_1)$
$(x_2, \tilde{y}_2)$
$(x_3, \tilde{y}_3)$
$(x_4, \tilde{y}_4)$