NAACL 2021

Self-Training with Weak Supervision

Giannis Karamanolakis, Subhabrata Mukherjee,
Guoqing Zheng, Ahmed Hassan Awadallah

Columbia University
gkaraman@cs.columbia.edu

Microsoft Research
{submukhe, zheng, hassanam}@microsoft.com

Dominant Supervised Learning Paradigm: A Labeled Data Bottleneck

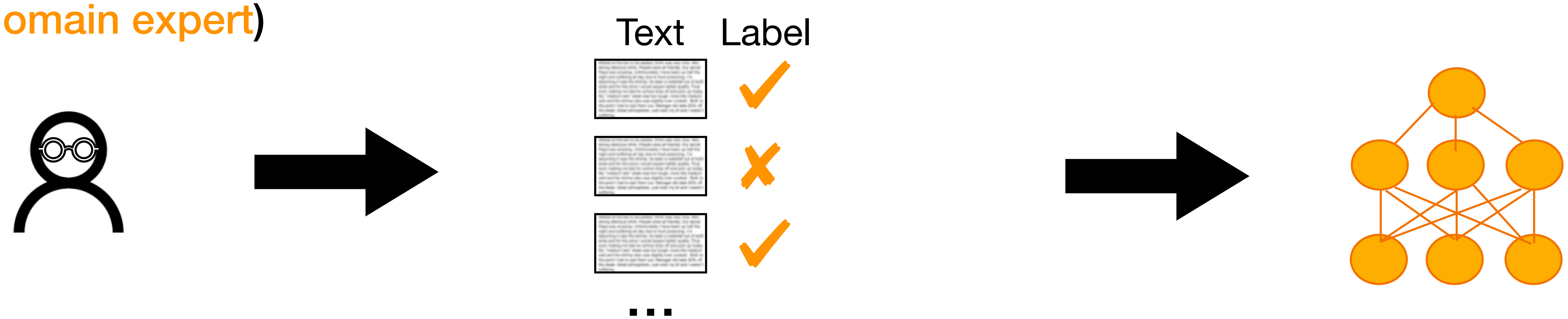
Task Specification

(e.g., document-level, binary sentiment classification)

Human
(e.g., domain expert)

Labeled Data

Classifier



Dominant Supervised Learning Paradigm: A Labeled Data Bottleneck

Task Specification
(e.g., document-level, binary sentiment classification)

Standard Benchmarks

- Fixed task specifications
- Large-scale labeled data

Dominant Supervised Learning Paradigm: A Labeled Data Bottleneck

Task Specification

(e.g., document-level, binary sentiment classification)

Standard Benchmarks

- Fixed task specifications
- Large-scale labeled data

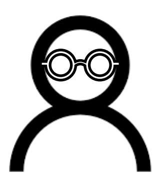
Real-World Applications

- Dynamic task specifications
- Limited or no labeled data

Dominant Supervised Learning Paradigm: A Labeled Data Bottleneck

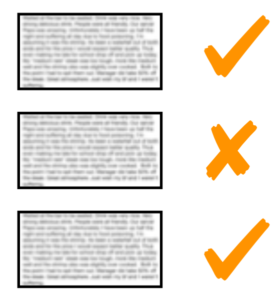
Task Specification
(e.g., document-level, binary sentiment classification)

Human
(e.g., domain expert)



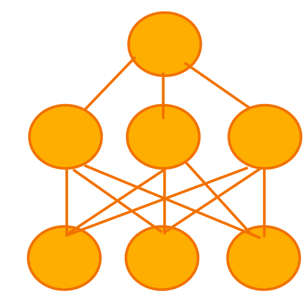
Labeled Data

Text Label



...

Classifier



(-) expensive
(-) time-consuming

(-) static

"labeled data bottleneck"

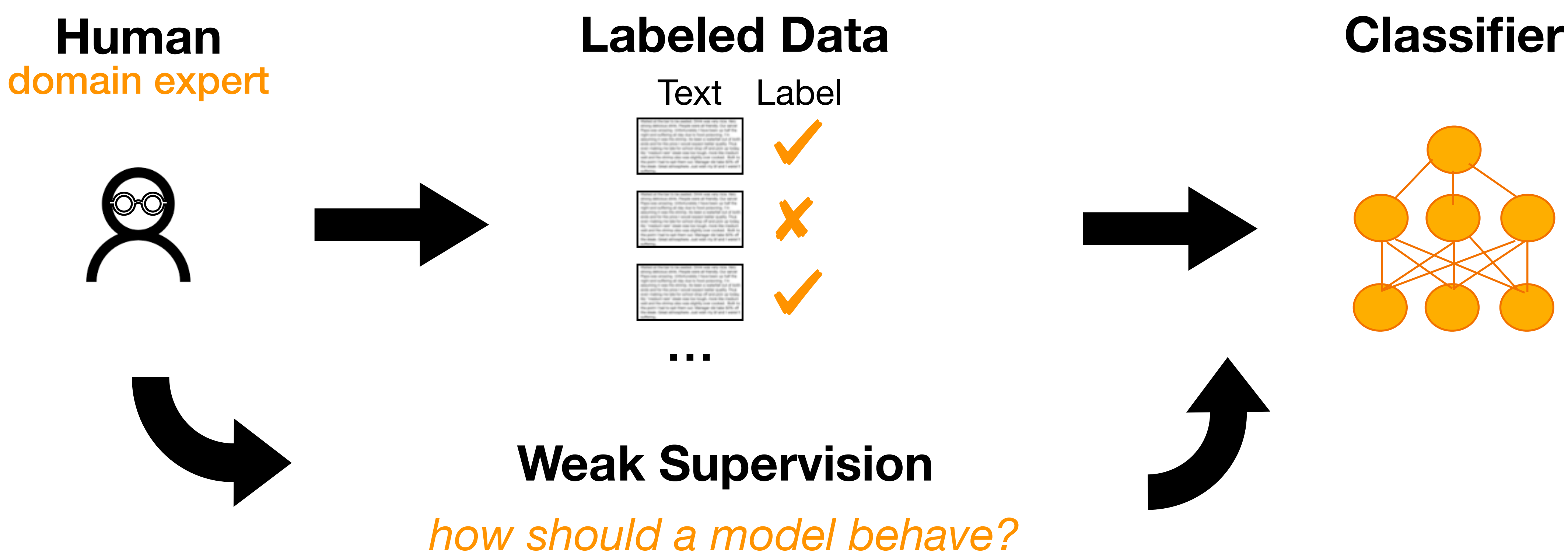
Standard Benchmarks

- Fixed task specifications
- Large-scale labeled data

Real-World Applications

- Dynamic task specifications
- Limited or no labeled data

Addressing the Labeled Data Bottleneck with Weak Supervision



Our Goal: *leverage **domain expertise** in absence of large-scale labeled data* **+** *leverage generalization power of **deep neural networks***

Weak Supervision Via Domain-Specific Rules

- Rules: heuristic labeling functions written by **domain experts**
- Rules are used to automatically annotate **unlabeled** data

---

Example: regular expression patterns

Spam classification

```
def regex_check_out(x):
    return SPAM if re.search("check.*out", x) else ABSTAIN
```

Question type classification

```
def numeric_question(x):
    return NUMERIC if x.startswith("when") else ABSTAIN
```

Weak Supervision Via Domain-Specific Rules

- Rules: heuristic labeling functions written by **domain experts**
- Rules are used to automatically annotate **unlabeled** data

---

Example: regular expression patterns

Spam classification

```
def regex_check_out(x):
    return SPAM if re.search("check.*out", x) else ABSTAIN
```

Question type classification

```
def numeric_question(x):
    return NUMERIC if x.startswith("when") else ABSTAIN
```

Example: heuristic functions based on lexicons / models / knowledge bases

Sentiment classification

```
def sentiment_lexicon_score(x, sentiwordnet):
    if sentiwordnet(x) > 0.8:
        return POSITIVE
    elif sentiwordnet(x) < 0.2:
        return NEGATIVE
    else:
        ABSTAIN
```

Challenges in Learning with Weak Rules

(1) Noise

$\text{rule}(x_i) \rightarrow \text{SPAM}$ ✘

True label: HAM

---

(2) Coverage

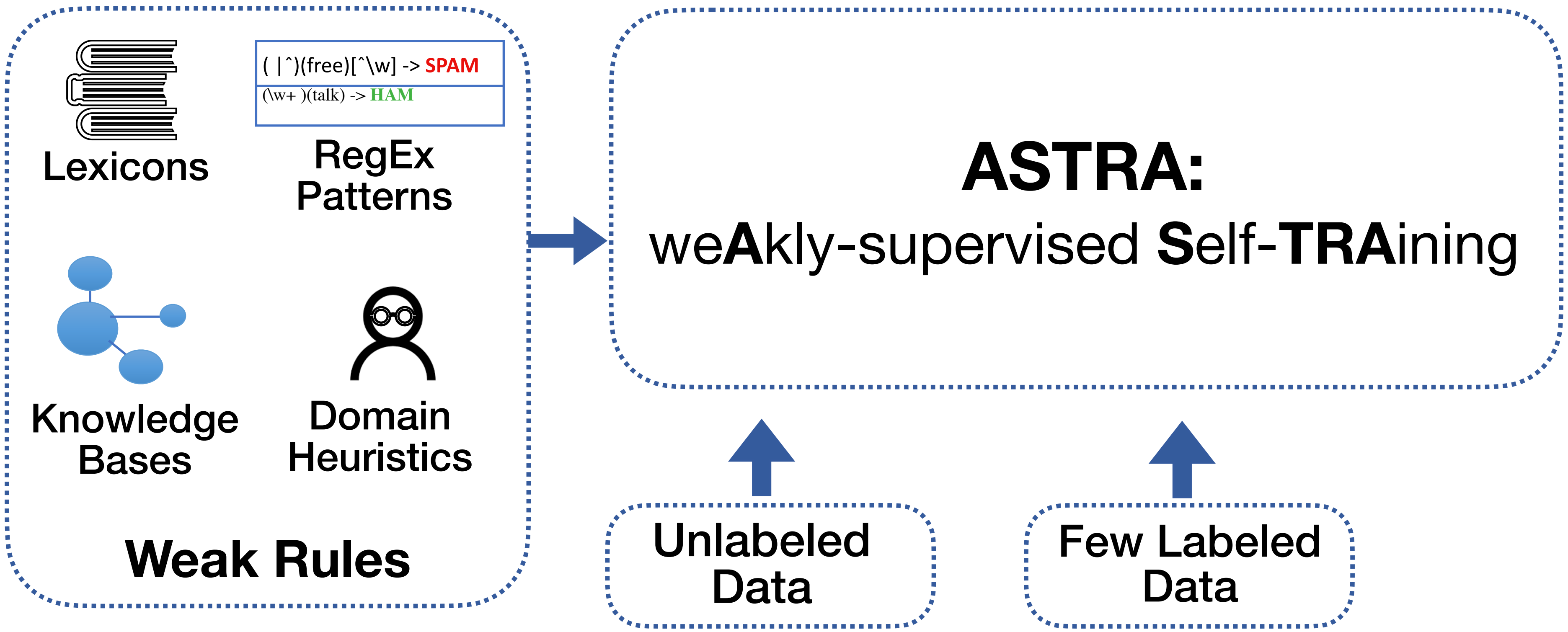
$\text{rule}(x_i) \rightarrow \text{ABSTAIN}$

---

(3) Conflicts

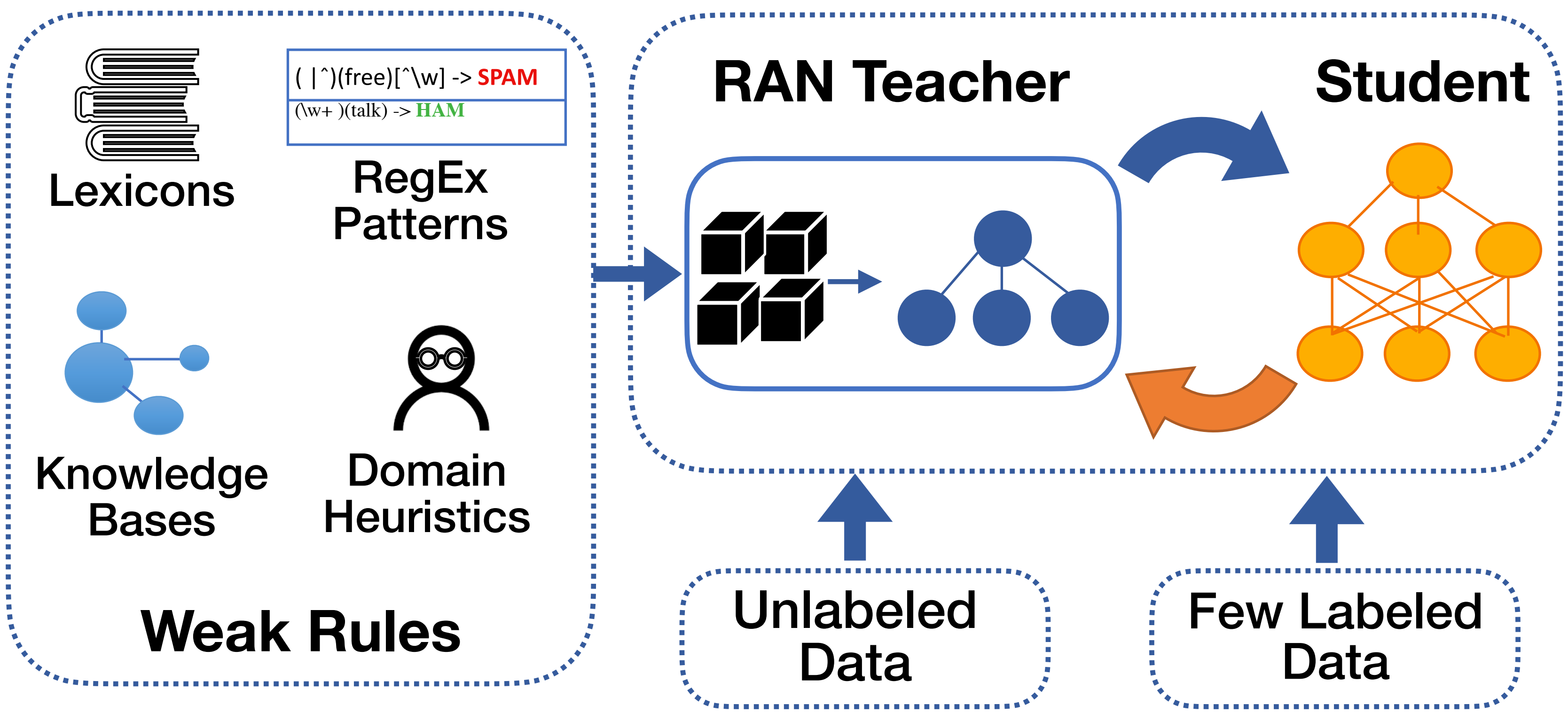
$$x_i \begin{cases} \text{rule1} \rightarrow \text{SPAM} \\ \text{rule2} \rightarrow \text{HAM} \\ \text{rule3} \rightarrow \text{SPAM} \\ \text{rule4} \rightarrow \text{HAM} \end{cases} \Rightarrow \text{???}$$

Our ASTRA Framework for Weak Supervision



https://github.com/microsoft/ASTRA

Our ASTRA Framework for Weak Supervision



Our Contributions:

1. Present an **iterative self-training** mechanism for training deep neural networks (Student) with weak supervision
2. Present a **rule attention network** (RAN Teacher) for aggregating multiple weak sources with instance-specific weights and construct an **SSL objective**
3. Show the effectiveness of ASTRA on **six benchmarks** for text classification

https://github.com/microsoft/ASTRA

Outline

1. Learning with Domain-Specific Rules

2. ASTRA: we**A**kly-supervised **S**elf-**TRA**ining

3. Experiments

4. Conclusions

How to Train Robust Classifiers with Weak Rules?

(1) Noise **(2) Low Coverage** **(3) Conflicts**

Limitation of Previous Methods for Weak Supervision

- Previous work **ignore unlabeled instances** that are **not** covered by rules

[Ratner et al., 2017; Bach et al., 2019; Awasthi et al., 2020]

Limitation of Previous Methods for Weak Supervision

- Previous work **ignore unlabeled instances** that are **not** covered by rules

[Ratner et al., 2017; Bach et al., 2019; Awasthi et al., 2020]

All Instances → Covered by Rules → **Weak Supervision**

All Instances → **Not** Covered by Rules → **Filtered Out**

- Expert-defined rules are usually **sparse**:

6 real-world datasets
45 rules / dataset
→
- **just 33%** of instances covered by > 1 **rule**
- **40%** of instances are **not** covered by **any rules**

Limitation of Previous Methods for Weak Supervision

- Previous work **ignore unlabeled instances** that are **not** covered by rules

[Ratner et al., 2017; Bach et al., 2019; Awasthi et al., 2020]

All Instances → Covered by Rules → **Weak Supervision**

All Instances → **Not** Covered by Rules ✗ **Filtered Out**

- Expert-defined rules are usually **sparse**:

6 real-world datasets
45 rules / dataset

→

- **just 33%** of instances covered by > 1 **rule**
- **40%** of instances are **not** covered by **any rules**

Filtered-out

Limitation of Previous Methods for Weak Supervision

- Previous work **ignore unlabeled instances** that are **not** covered by rules

[Ratner et al., 2017; Bach et al., 2019; Awasthi et al., 2020]

- Expert-defined rules are usually **sparse**:

6 real-world datasets
45 rules / dataset

- **just 33%** of instances covered by > 1 **rule**
- **40%** of instances are **not** covered by **any rules**

Filtered-out

Don’t throw them away!

Limitation of Previous Methods for Weak Supervision

- Previous work **ignore unlabeled instances** that are **not** covered by rules

[Ratner et al., 2017; Bach et al., 2019; Awasthi et al., 2020]

All Instances → Covered by Rules → **Weak Supervision**

All Instances → **Not** Covered by Rules ✗ **Filtered Out**

- Expert-defined rules are usually **sparse**:

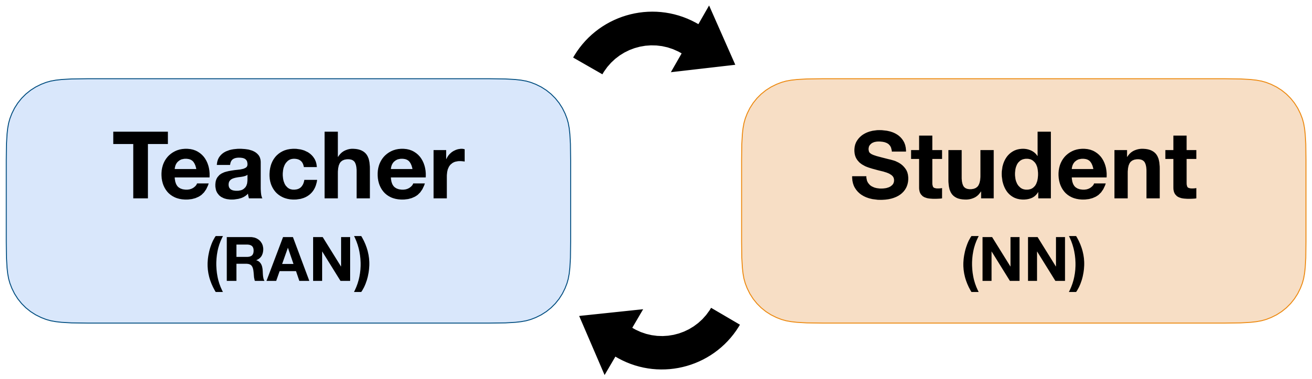
6 real-world datasets
45 rules / dataset
→
- **just 33%** of instances covered by > 1 **rule**
- **40%** of instances are **not** covered by **any rules**

Filtered-out Don't throw them away!

- We leverage **all unlabeled instances** for weak supervision via **self-training**

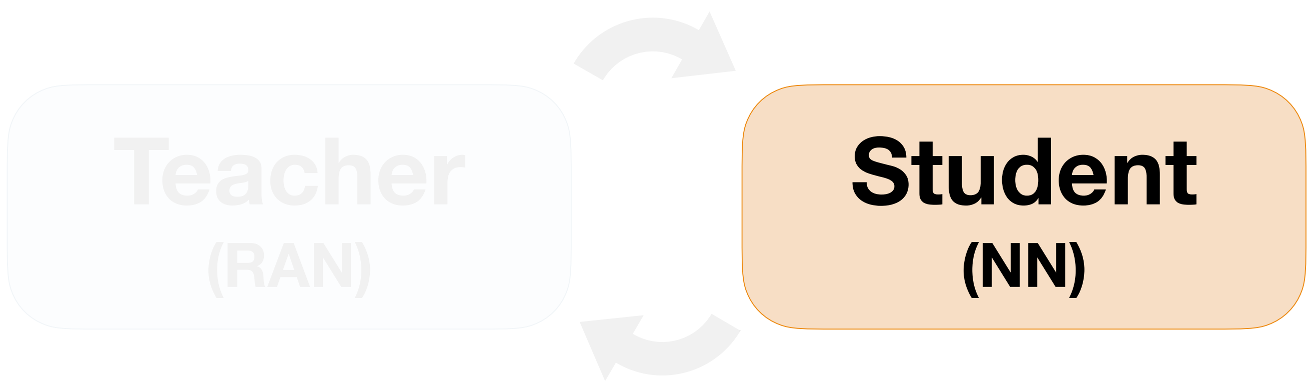
ASTRA: Weakly-Supervised Self-Training

1. Student
2. Teacher



ASTRA: Weakly-Supervised Self-Training

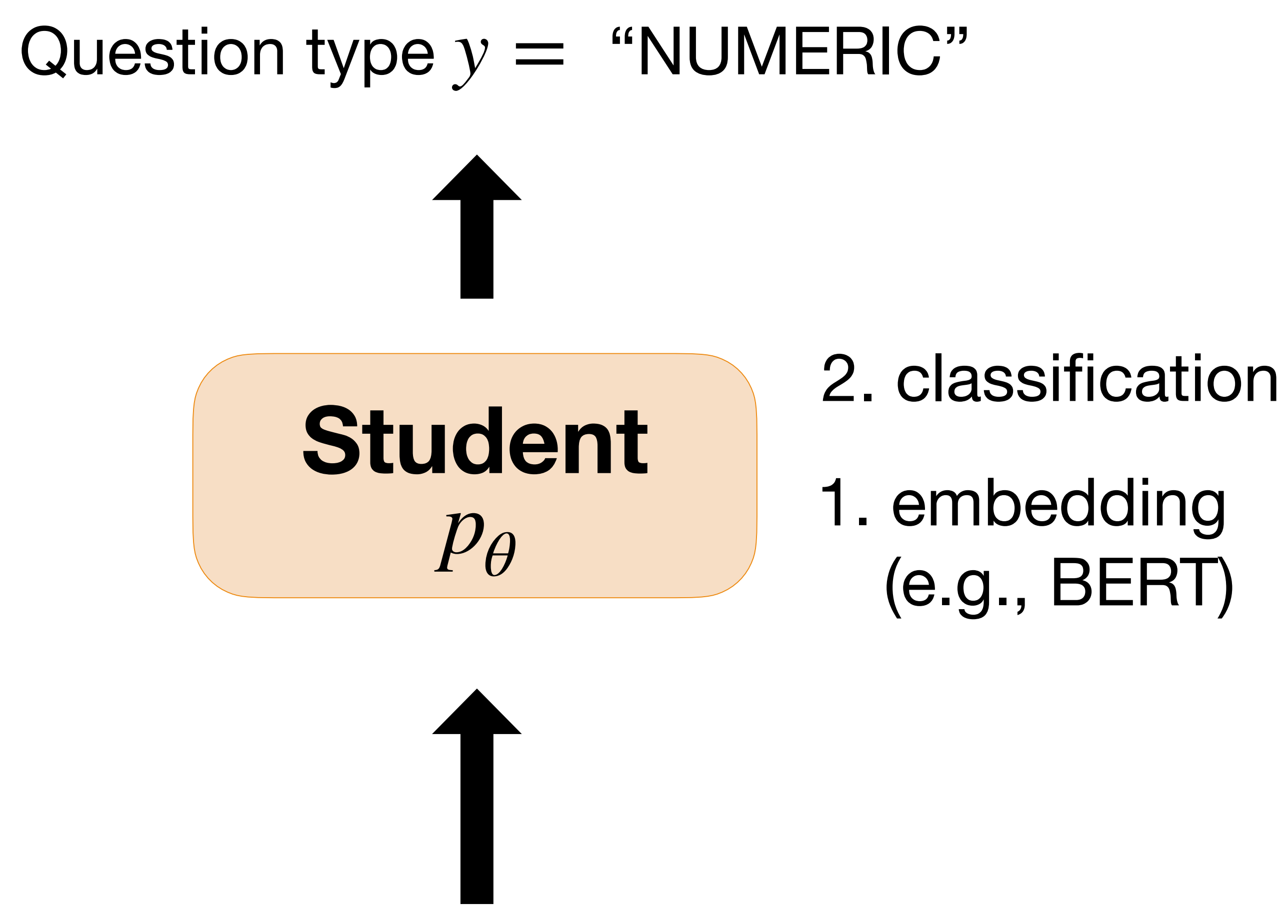
1. **Student**
2. Teacher



Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations

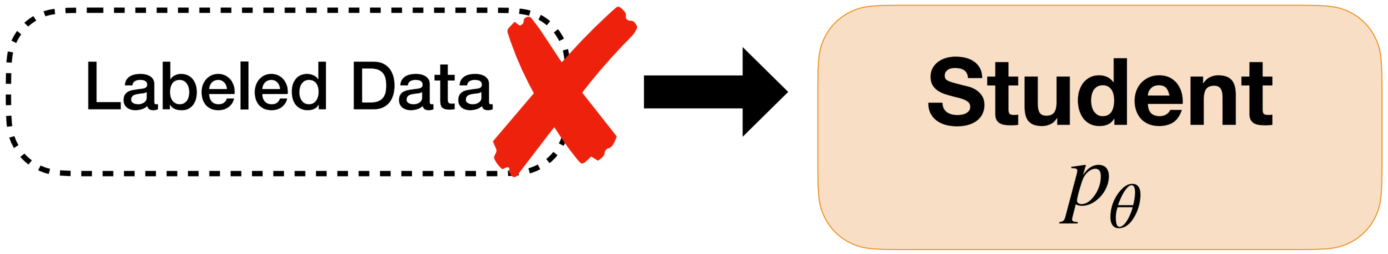
<u>Example: Question Type Classification (in TREC)</u>



input x: “What is the percentage of water content in the human body?”

Student: An Embedding-Based Neural Network

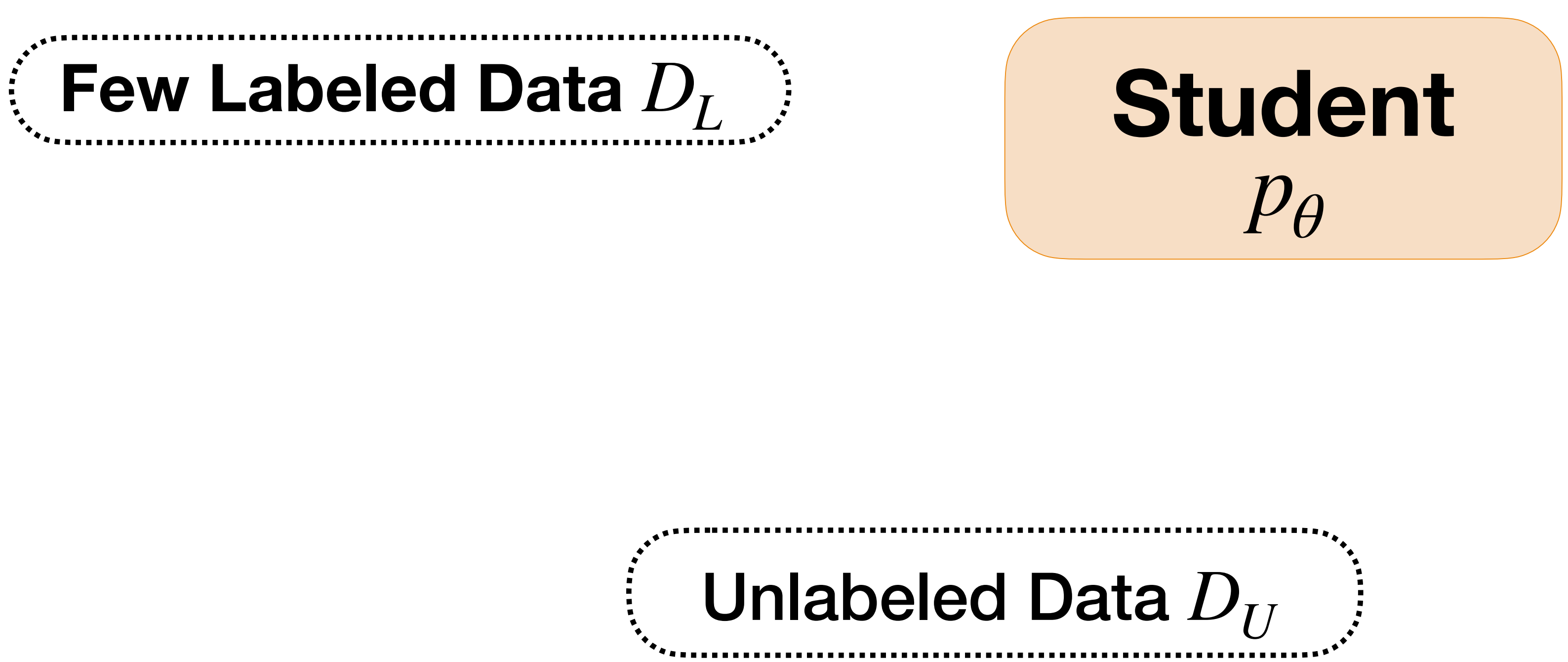
- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain



Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain

Self-Training Paradigm



Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain

Self-Training Paradigm

$$\min_{\theta} \mathbb{E}_{x,y \in D_L} - \log \; p_\theta(y \mid x)$$

Few Labeled Data D_L $\xrightarrow{\textbf{(1)}}$ **Student** p_θ

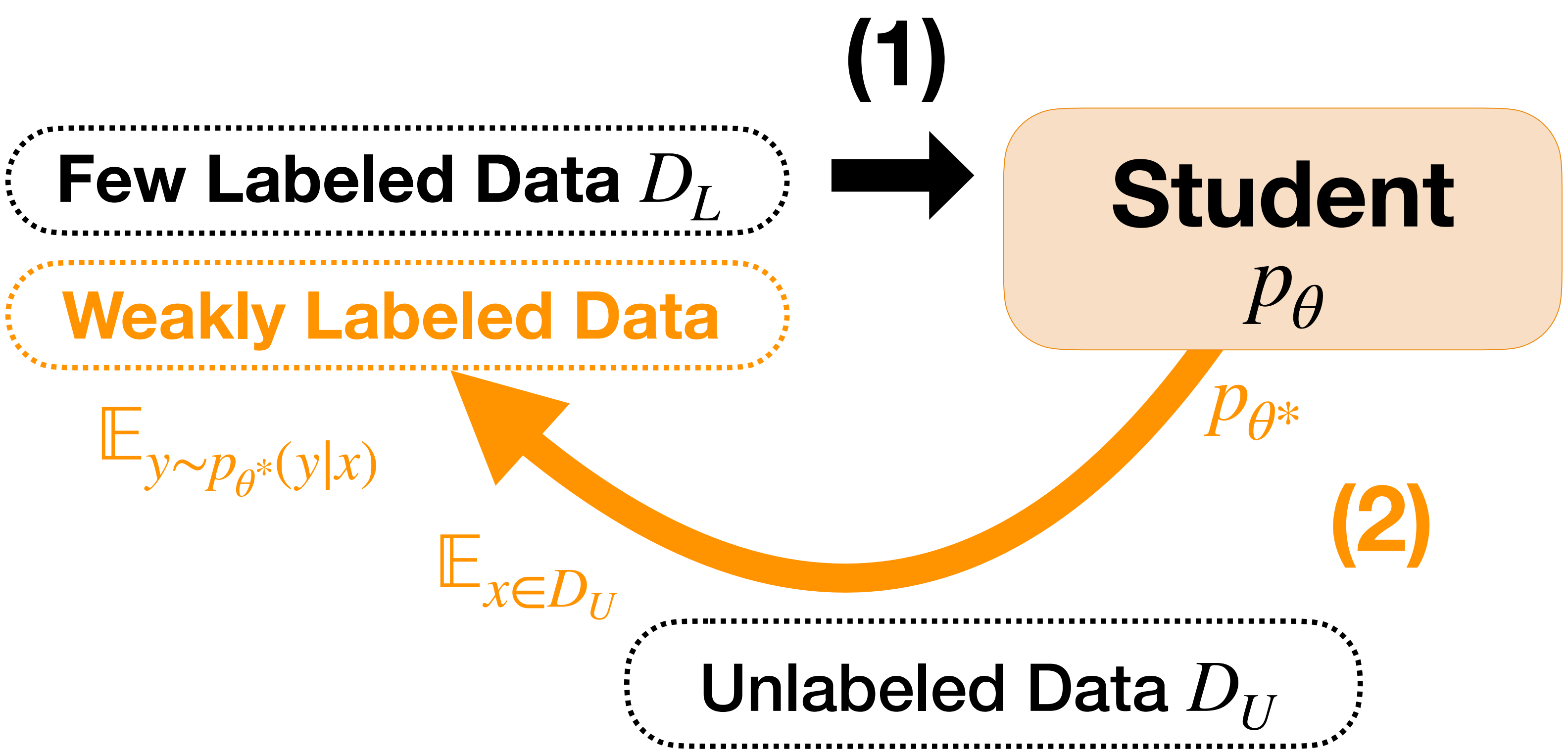
Unlabeled Data D_U

Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain

Self-Training Paradigm

$$\min_{\theta} \; \mathbb{E}_{x,y \in D_L} - \log \; p_{\theta}(y \mid x) \qquad \mathbb{E}_{x \in D_U} \; \mathbb{E}_{y \sim p_{\theta^*}(y|x)}$$

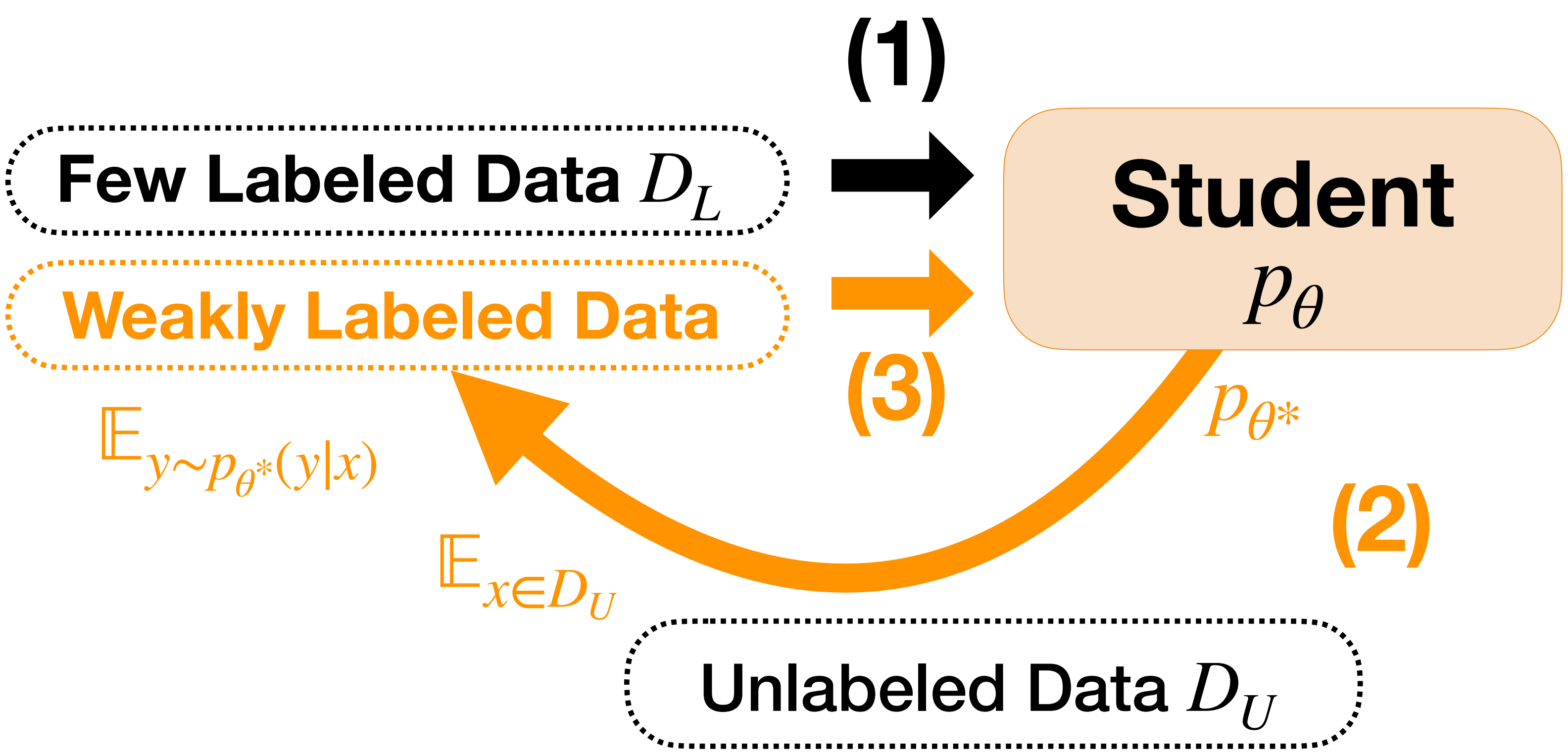


Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain

Self-Training Paradigm

$$\min_{\theta} \ \mathbb{E}_{x,y \in D_L} - \log \ p_\theta(y \mid x) \ + \ \lambda \mathbb{E}_{x \in D_U} \ \mathbb{E}_{y \sim p_{\theta^*}(y|x)} - \log \ p_\theta(y \mid x)$$



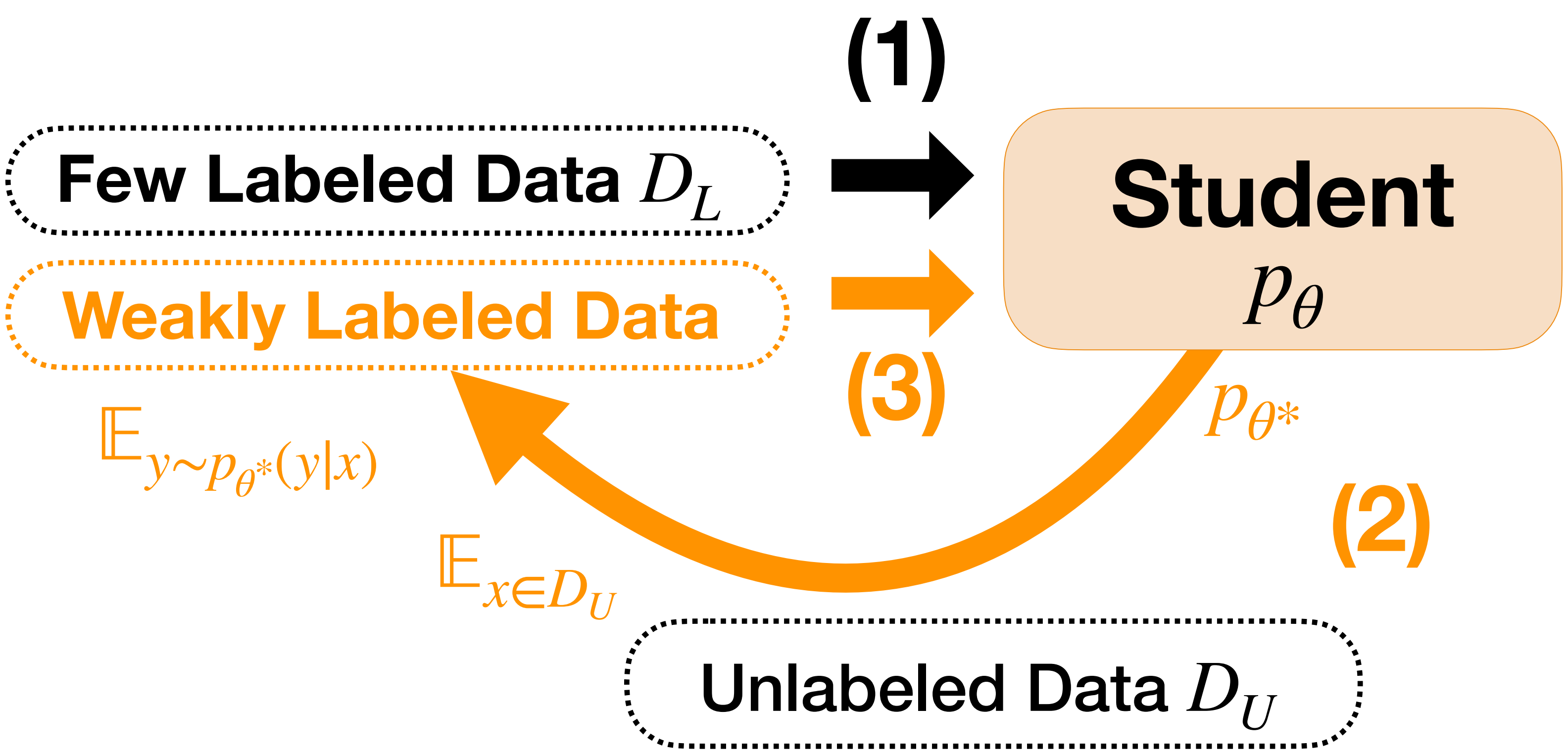
Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain

Self-Training Paradigm

$$\min_{\theta} \ \mathbb{E}_{x,y \in D_L} - \log \ p_\theta(y \mid x) \ + \ \lambda \mathbb{E}_{x \in D_U} \ \mathbb{E}_{y \sim p_{\theta^*}(y|x)} - \log \ p_\theta(y \mid x)$$

(-) Prone to error propagation

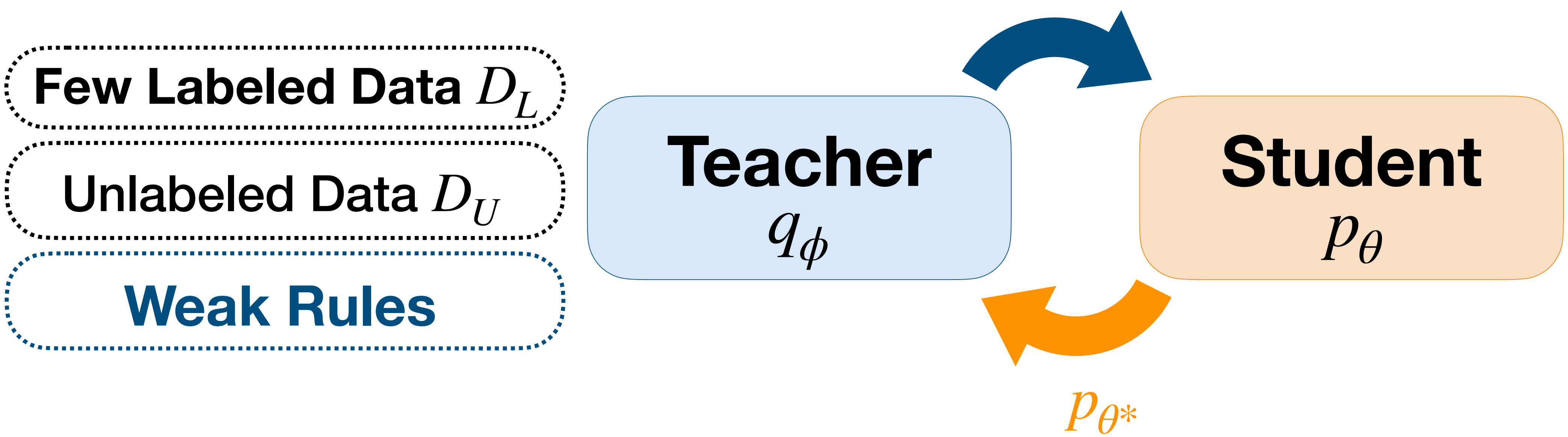


Student: An Embedding-Based Neural Network

- Represents input x using contextualized representations
- Large-scale labeled data is expensive to obtain
- We train Student using Teacher's labels

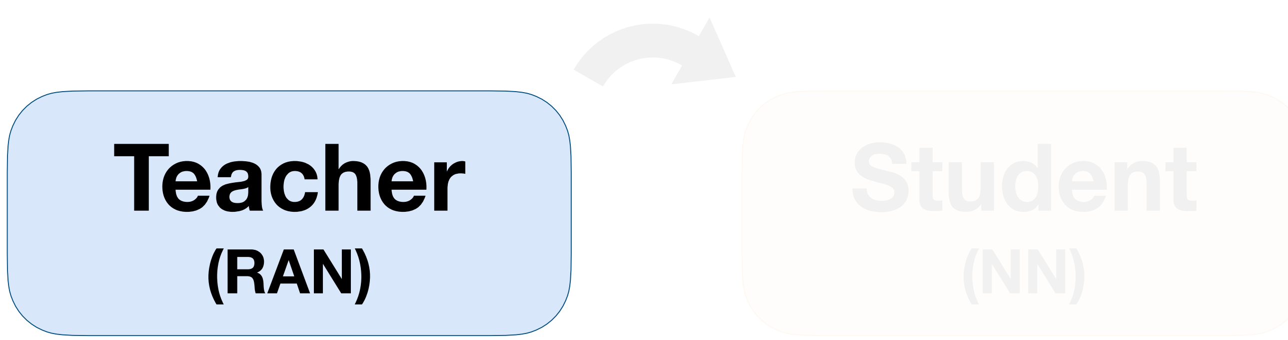
<u>Weakly-Supervised Self-Training</u>

$$\min_{\theta} \ \mathbb{E}_{x,y\in D_L} - \log \ p_\theta(y \mid x) \quad + \quad \lambda \mathbb{E}_{x\in D_U} \ \underset{\mathbb{E}_{y\sim q_{\phi^*}(y|x)}}{\cancel{\mathbb{E}_{y\sim p_{\theta^*}(y|x)}}} - \log \ p_\theta(y \mid x)$$



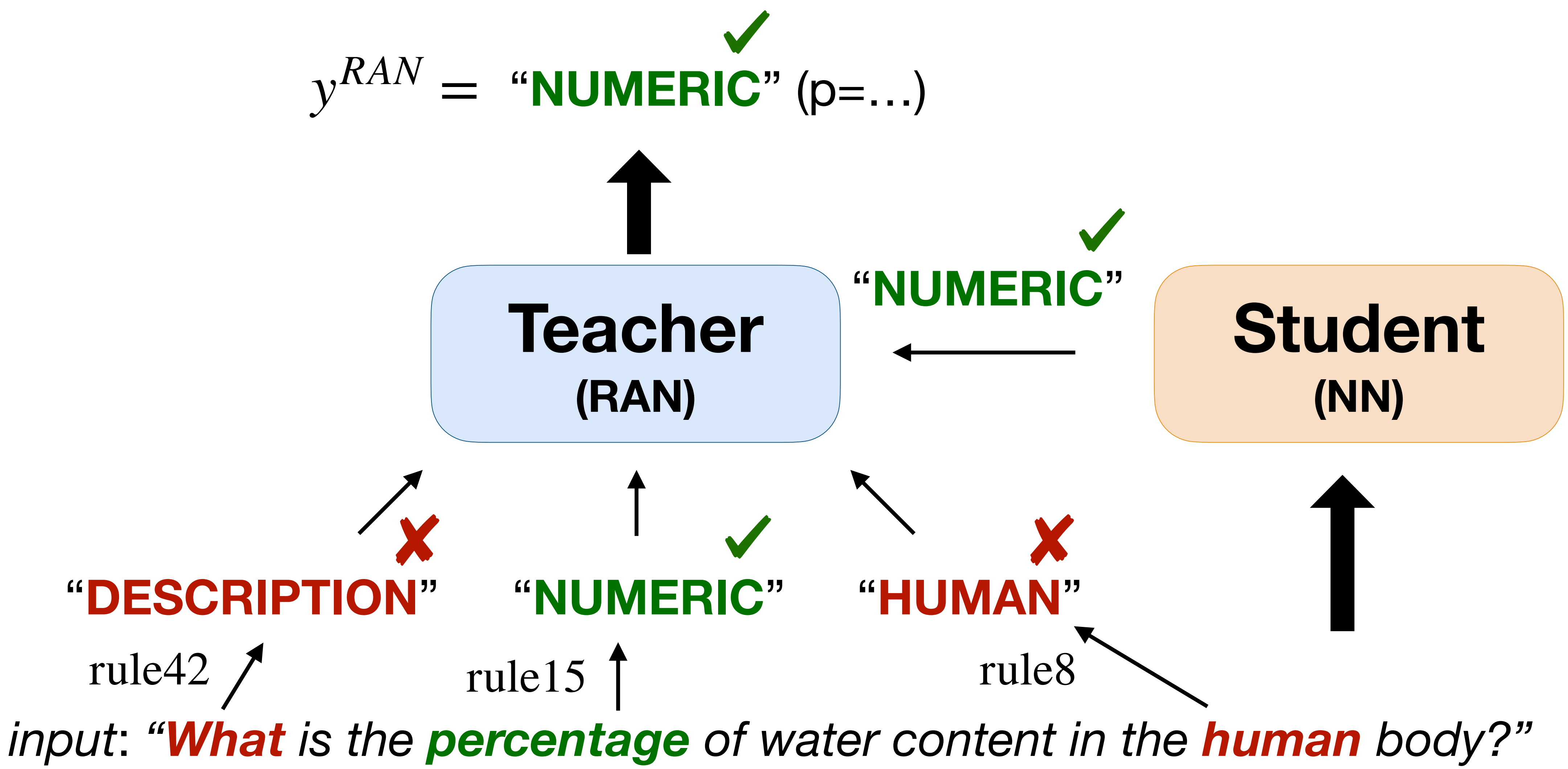
ASTRA: Weakly-Supervised Self-Training

1. Student
2. **Teacher: a Rule Attention Network (RAN)**



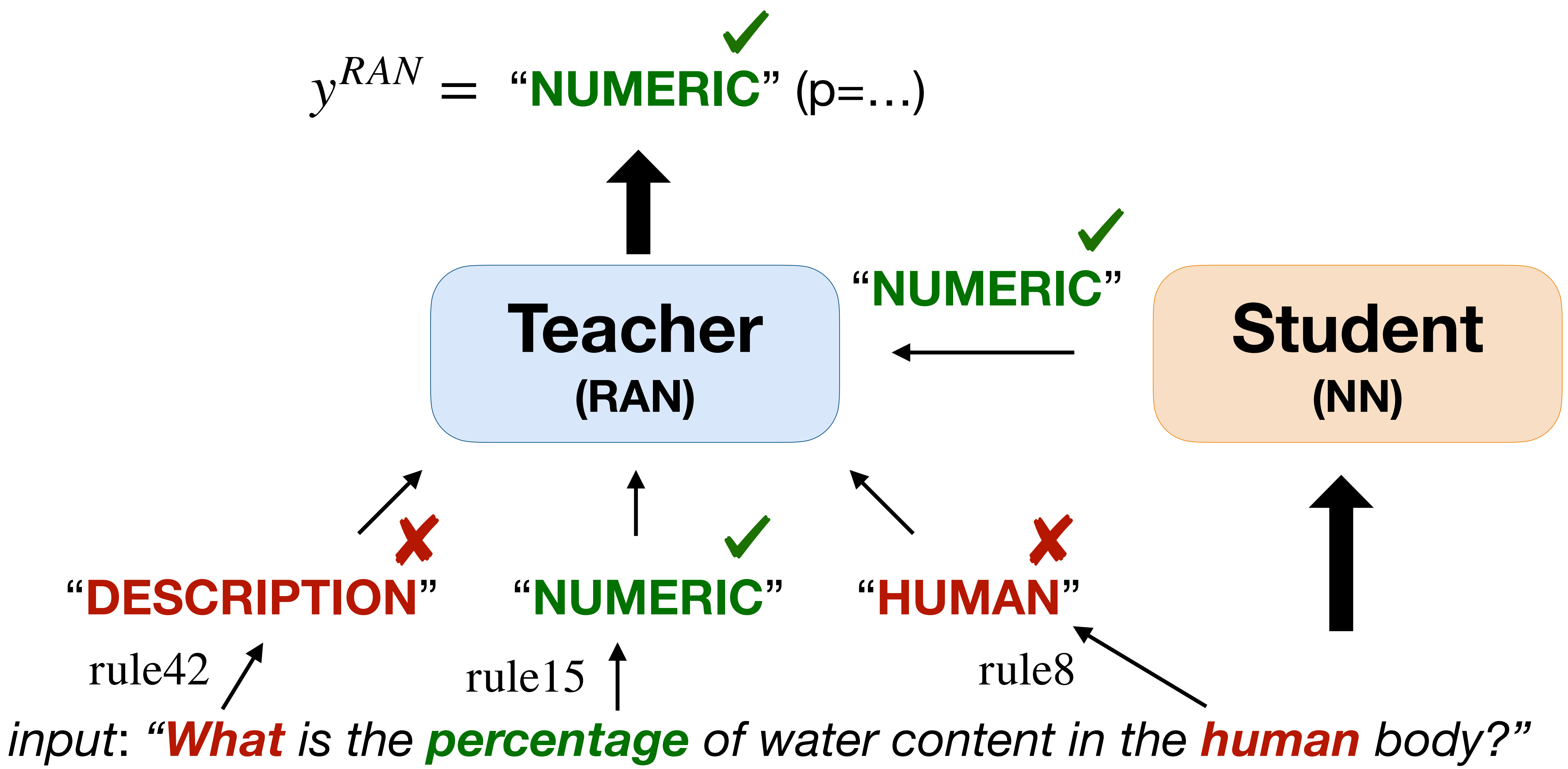
Teacher: Rule Attention Network (RAN)

- RAN aggregates weak labels predicted by **rules** and **Student**



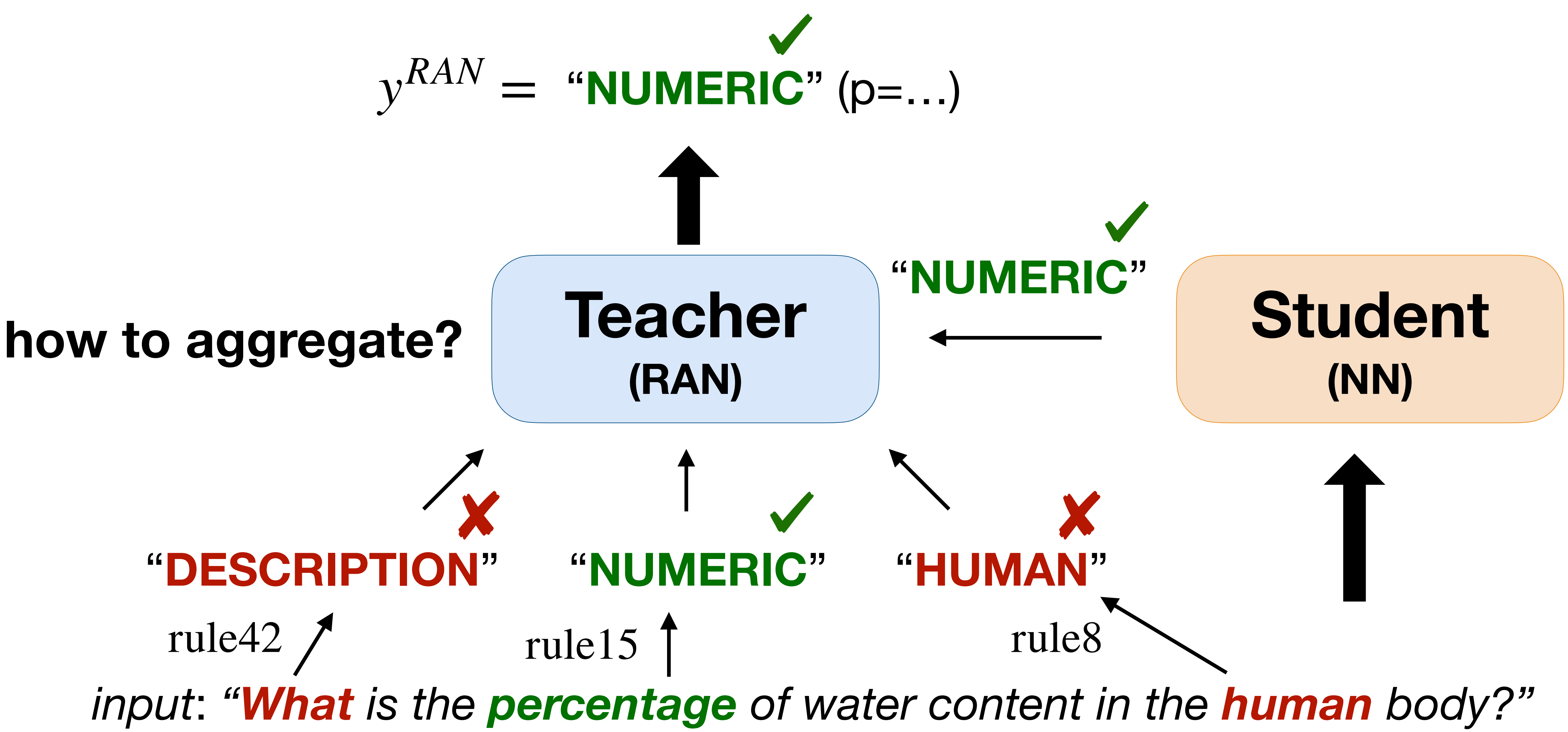
Teacher: Rule Attention Network (RAN)

- RAN aggregates weak labels predicted by **rules** and **Student**
  - **Heuristic rules** cover only a subset of the data
  - **Student** covers more data via contextualized embeddings



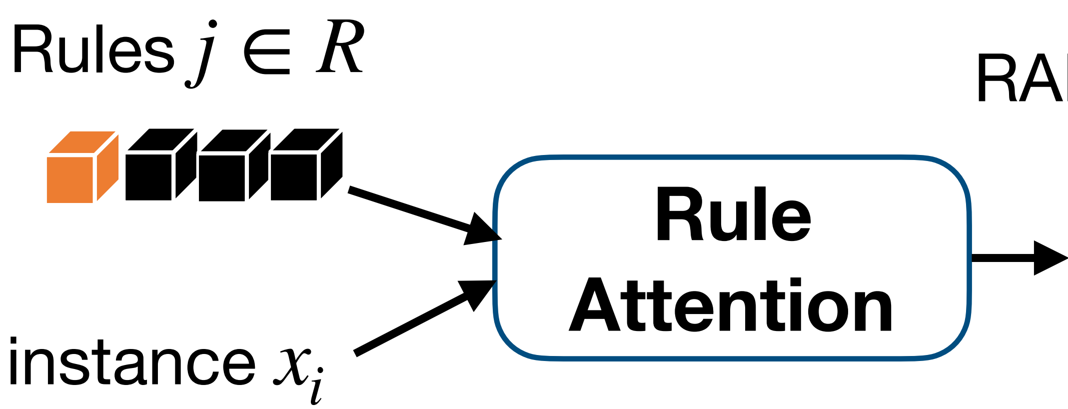
Teacher: Rule Attention Network (RAN)

- RAN aggregates weak labels predicted by **rules** and **Student**
  - **Heuristic rules** cover only a subset of the data
  - **Student** covers more data via contextualized embeddings



Teacher: Rule Attention Network (RAN)

- RAN aggregates weak labels predicted by **rules** and **Student**
- RAN learns to predict **instance-specific** weights using **rule attention**



RAN label

$$q_i = \frac{1}{Z} \sum_{j \in R} a_i^j q_i^j + (1 - a_i^j) u$$

weight a_i^j **label** q_i^j $u = [\frac{1}{K}, \ldots, \frac{1}{K}]$

Teacher: Rule Attention Network (RAN)

- RAN aggregates weak labels predicted by **rules** and **Student**
- RAN learns to predict **instance-specific** weights using **rule attention**
- RAN does **not** require rule supervision: we employ a **SSL objective**

RAN label

$$q_i = \frac{1}{Z} \sum_{j \in R} a_i^j q_i^j + (1 - a_i^j) u$$

Semi-Supervised Training Objective: $\mathscr{L}^{RAN} = - \sum_{(x_i, y_i) \in D_L} y_i \log q_i - \sum_{x_i \in D_U} q_i \log q_i \, .$

Cross-Entropy (labeled data) **Min-Entropy (unlabeled data)**

Teacher: Rule Attention Network (RAN)

- RAN aggregates weak labels predicted by **rules** and **Student**
- RAN learns to predict **instance-specific** weights using **rule attention**
- RAN does **not** require rule supervision: we employ a **SSL objective**

RAN label

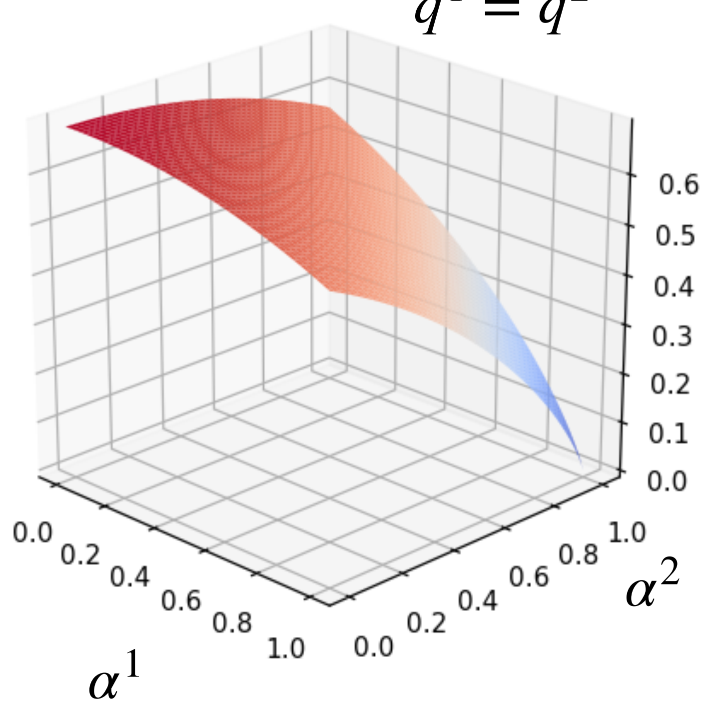
$$q_i = \frac{1}{Z} \sum_{j \in R} a_i^j q_i^j + (1 - a_i^j) u$$

Semi-Supervised Training Objective: $\mathscr{L}^{RAN} = - \sum_{(x_i, y_i) \in D_L} y_i \log q_i - \sum_{x_i \in D_U} q_i \log q_i .$

Cross-Entropy (labeled data)

Min-Entropy (unlabeled data)

$q^1 = q^2$

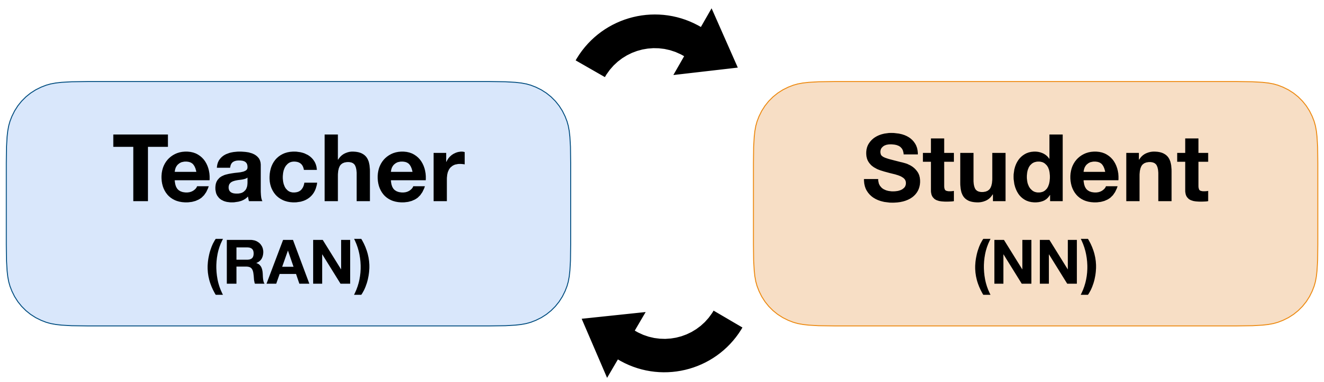


high weights $a^j = 1$ for rules j that agree in predictions q^j

more details in our paper!

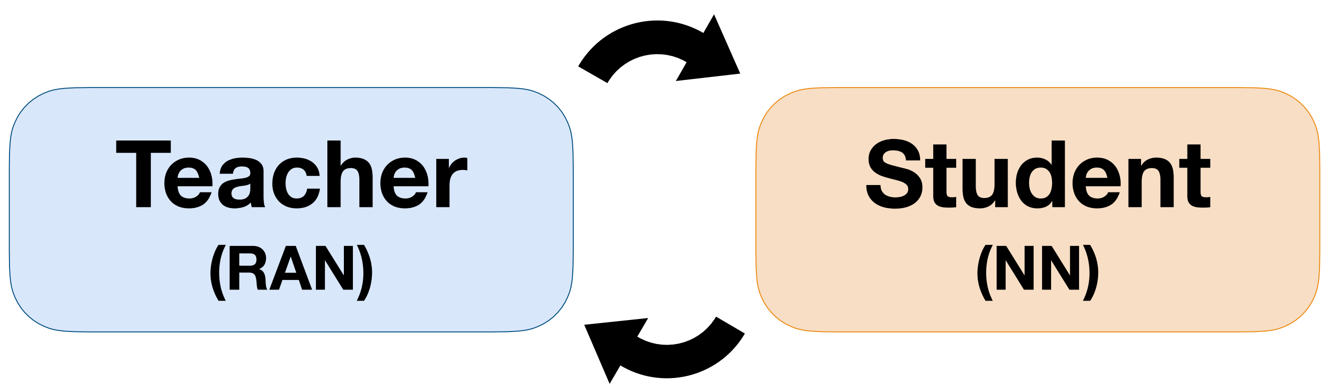
Summary of our ASTRA Framework

1. Train **Student** using few labeled data
2. Iterate:
    1. Train **RAN Teacher** to aggregate weak rules and Student
    2. Train **Student** using Teacher's labels



Summary of our ASTRA Framework

1. Train **Student** using few labeled data
2. Iterate:
    1. Train **RAN Teacher** to aggregate weak rules and Student
    2. Train **Student** using Teacher's labels



Access to rules during test time?

- YES -> use **Teacher** (Student + Rules)
- NO -> use **Student**

Outline

1. Learning with Domain-Specific Rules

2. ASTRA: weAkly-supervised Self-TRAining

3. Experiments on 6 classification benchmarks

4. Conclusions

Experiments: Learning with Weak Supervision

Benchmark	# Rules	Rule Coverage
TREC (question classification)	68	46%
SMS (spam classification)	73	9%
YouTube (spam classification)	10	48%
CENSUS (income classification)	83	94%
MIT-R (slot filling)	15	1%
Spouse (relation classification)	9	8%

- **Rule types:** keywords, regular expressions, lexicons, knowledge bases

Experiments: Learning with Weak Supervision

Benchmark	**# Rules**	**Rule Coverage**
TREC (question classification)	68	46%
SMS (spam classification)	73	9%
YouTube (spam classification)	10	48%
CENSUS (income classification)	83	94%
MIT-R (slot filling)	15	1%
Spouse (relation classification)	9	8%

- **Rule types:** keywords, regular expressions, lexicons, knowledge bases
- Rules are **sparse**:
  - 66% of the examples are covered by **fewer than 2 rules**
  - 40% of the examples are **not covered** by any rule

Results Summary Across 6 Benchmarks

Method	Learning to Weight: Rules	Learning to Weight: Instances	Unlabeled (no rules)	Average Accuracy
PosteriorReg (Hu et al., 2016)	✓	-	-	82.6
Snorkel (Ratner et al., 2017)	✓	-	-	82.9
L2R (Ren et al., 2018a)	-	✓	-	82.8
Standard self-training	-	-	✓	83.5 (+0.7%)

- **Self-training** outperforms weak supervision approaches…

… using unlabeled data and no rules!

Results Summary Across 6 Benchmarks

Method	Learning to Weight Rules	Learning to Weight Instances	Unlabeled (no rules)	Average Accuracy
PosteriorReg (Hu et al., 2016)	✓	-	-	82.6
Snorkel (Ratner et al., 2017)	✓	-	-	82.9
L2R (Ren et al., 2018a)	-	✓	-	82.8
Standard self-training	-	-	✓	83.5
ImplyLoss (Awasthi et al., 2020)	✓	✓	-	85.2
ASTRA	✓	✓	✓	**88.0** (+3.3%)

- **Self-training** outperforms weak supervision approaches
- **ASTRA** outperforms all previous approaches:

Results Summary Across 6 Benchmarks

Method	Learning to Weight Rules	Learning to Weight Instances	Unlabeled (no rules)	Average Accuracy
PosteriorReg (Hu et al., 2016)	✓	-	-	82.6
Snorkel (Ratner et al., 2017)	✓	-	-	82.9
L2R (Ren et al., 2018a)	-	✓	-	82.8
Standard self-training	-	-	✓	83.5
ImplyLoss (Awasthi et al., 2020)	✓	✓	-	85.2
ASTRA	✓	✓	✓	**88.0** (+3.3%)

- **Self-training** outperforms weak supervision approaches
- **ASTRA** outperforms all previous approaches:
    - (+) Learns **instance-specific** rule weights
    - (+) Leverages **all unlabeled data**

Results Summary Across 6 Benchmarks

Method	Learning to Weight: Rules	Learning to Weight: Instances	Unlabeled (no rules)	Average Accuracy
PosteriorReg (Hu et al., 2016)	✓	-	-	82.6
Snorkel (Ratner et al., 2017)	✓	-	-	82.9
L2R (Ren et al., 2018a)	-	✓	-	82.8
Standard self-training	-	-	✓	83.5
ImplyLoss (Awasthi et al., 2020)	✓	✓	-	85.2
ASTRA	✓	✓	✓	**88.0** (+3.3%)

- **Self-training** outperforms weak supervision approaches
- **ASTRA** outperforms all previous approaches:
  - (+) Learns **instance-specific** rule weights
  - (+) Leverages **all unlabeled data**
  - (+) Does **not** require rule supervision ("rule exemplars" in Awasthi et al., 2020)

Results Summary Across 6 Benchmarks

Method	Learning to Weight Rules	Learning to Weight Instances	Unlabeled (no rules)	Average Accuracy
PosteriorReg (Hu et al., 2016)	✓	-	-	82.6
Snorkel (Ratner et al., 2017)	✓	-	-	82.9
L2R (Ren et al., 2018a)	-	✓	-	82.8
Standard self-training	-	-	✓	83.5
ImplyLoss (Awasthi et al., 2020)	✓	✓	-	85.2
ASTRA	✓	✓	✓	**88.0** (+3.3%)

- **Self-training** outperforms weak supervision approaches
- **ASTRA** outperforms all previous approaches:
- **ASTRA** shows strongest improvements under **high rule sparsity**

More results and ablation studies in our paper!

Outline

1. Learning with Domain-Specific Rules

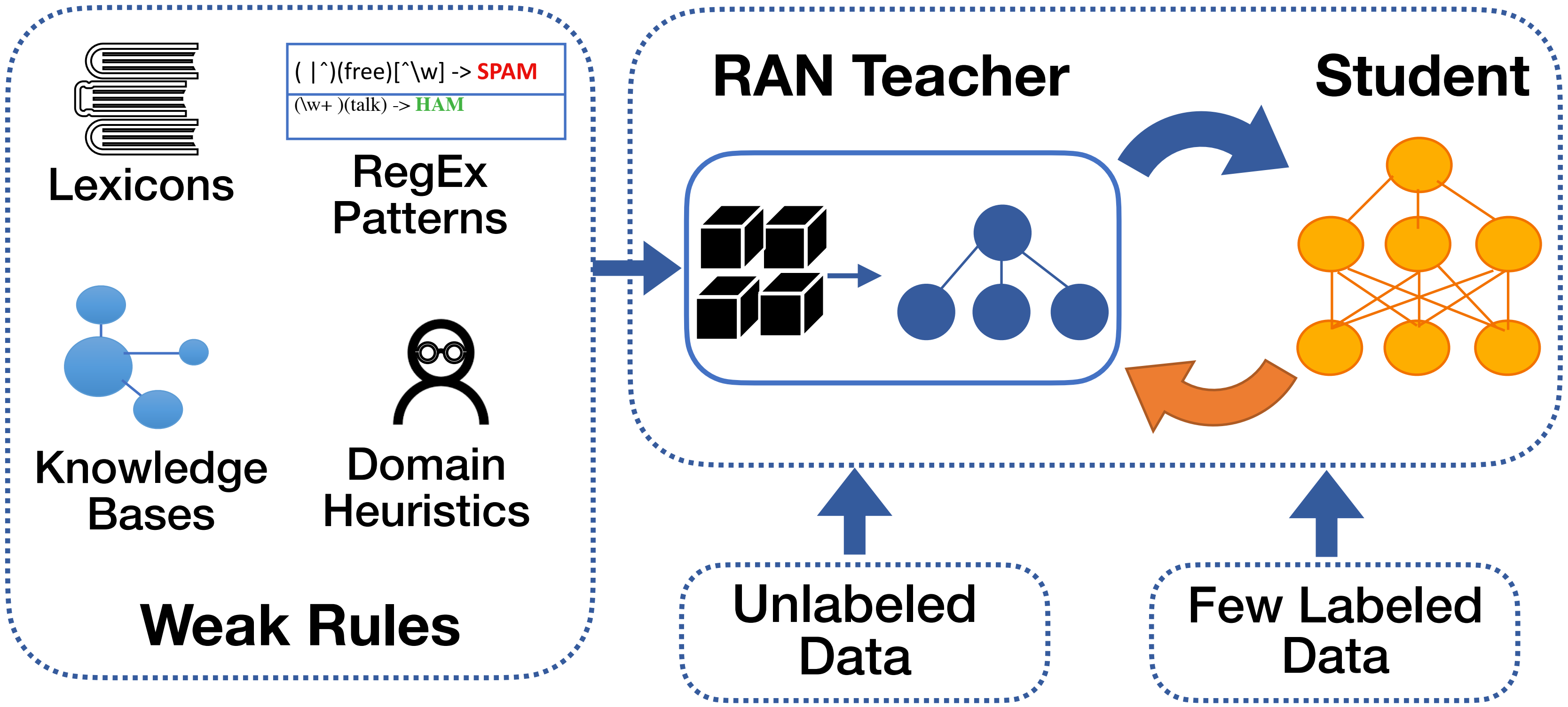
2. ASTRA: weAkly Supervised self-TRAining

3. Experiments

4. Conclusions

Our ASTRA Framework for Weak Supervision

ASTRA: weAkly-supervised Self-TRAining



1. **Iterative self-training framework** for weak supervision
2. **Rule attention network (RAN)** for combining weak rules and Student
3. Effectiveness on **six benchmark datasets**

Our code is available at **https://github.com/microsoft/ASTRA**

Thank you!

Contact
Giannis Karamanolakis
gkaraman@cs.columbia.edu

Our code is available at **https://github.com/microsoft/ASTRA**