

# End-to-End Weak Supervision

*Neural Information Processing Systems (NeurIPS), 2021*



**Salva Rühling Cachay<sup>1,2</sup>, Benedikt Boecking<sup>1</sup>, Artur Dubrawski<sup>1</sup>**

<sup>1</sup>Auton Lab, Carnegie Mellon University

<sup>2</sup>Technical University of Darmstadt

# Successful Machine Learning methods require large amounts of labeled data



<https://medium.com/syncedreview/sensetime-trains-imagenet-alexnet-in-record-1-5-minutes-e944ab049b2c>

Hand labeling, however, is expensive both in terms of time and cost



<https://medium.com/syncedreview/sensetime-trains-imagenet-alexnet-in-record-1-5-minutes-e944ab049b2c>

# Alternative: (Multi-source) Weak supervision <sup>[1]</sup>

[1] A. Ratner, C. De Sa, S. Wu, D. Selsam, C. Ré, “Data programming: Creating large training sets, quickly”, NeurIPS 2016.

# Weak Supervision

Multiple noisy heuristics that cheaply apply to unlabeled data!

= **Labeling functions (LFs)**

## Domain Heuristics

```
def f1(text):  
    return (SPAM  
           if `money`  
           in text  
           else ABSTAIN)
```

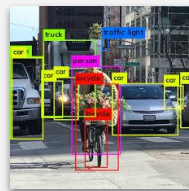
e.g. Hearst (1992), Dunnmon et al. (2020)

## Distant Supervision



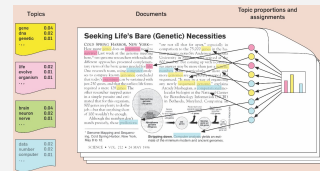
e.g. Mintz et al. (2019),  
Bach et al. (2019)

## Pretrained Models



e.g. Chen et al. (2019)

## Unsupervised Models



e.g. Hingmire et al. (2014),  
Bach et al. (2019)

# The usual approach

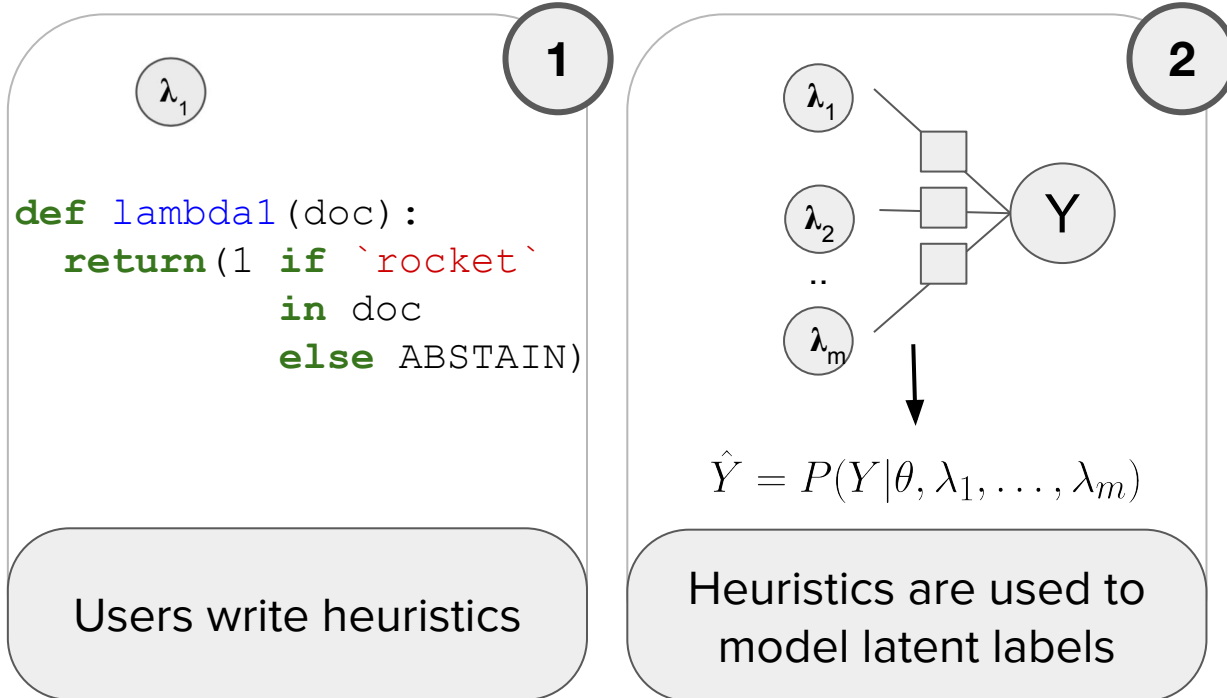
$\lambda_1$

1

```
def lambda1 (doc) :  
    return (1 if `rocket`  
            in doc  
            else ABSTAIN)
```

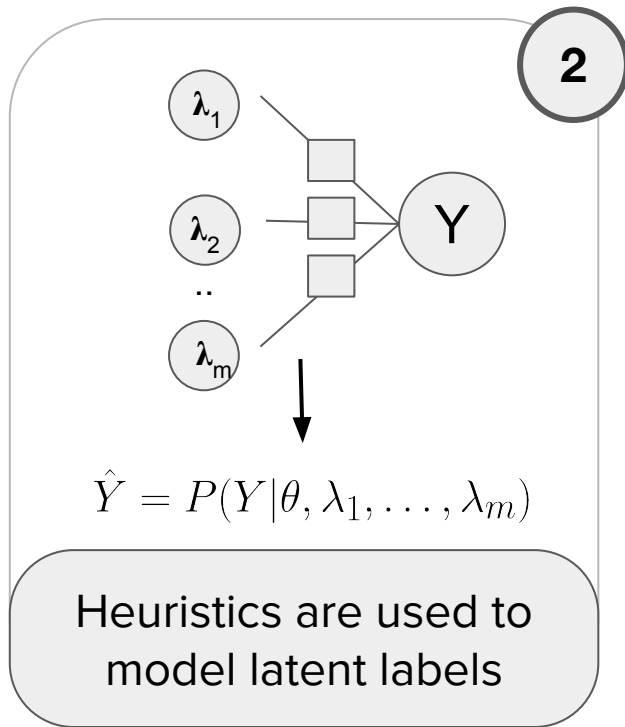
Users write heuristics

# The usual approach



## But...

- Statistical dependencies are hard to model (efficiently)
  - Thus, they are often simply ignored!
- No data features/representations are considered!



→ This & more (often) violates assumptions needed for theoretical results



# The usual approach

Two separate modeling steps!

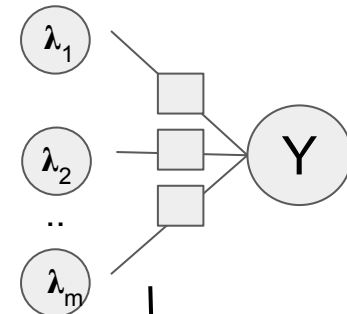
①

$\lambda_1$

```
def lambda1(doc):  
    return (1 if 'rocket'  
            in doc  
            else ABSTAIN)
```

Users write heuristics

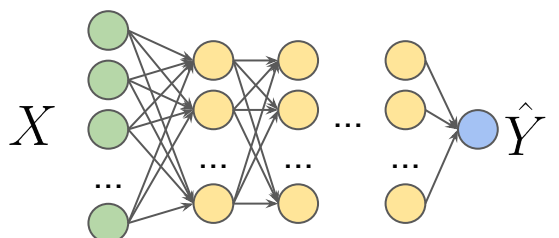
②



$\hat{Y} = P(Y|\theta, \lambda_1, \dots, \lambda_m)$

Heuristics are used to model latent labels

③



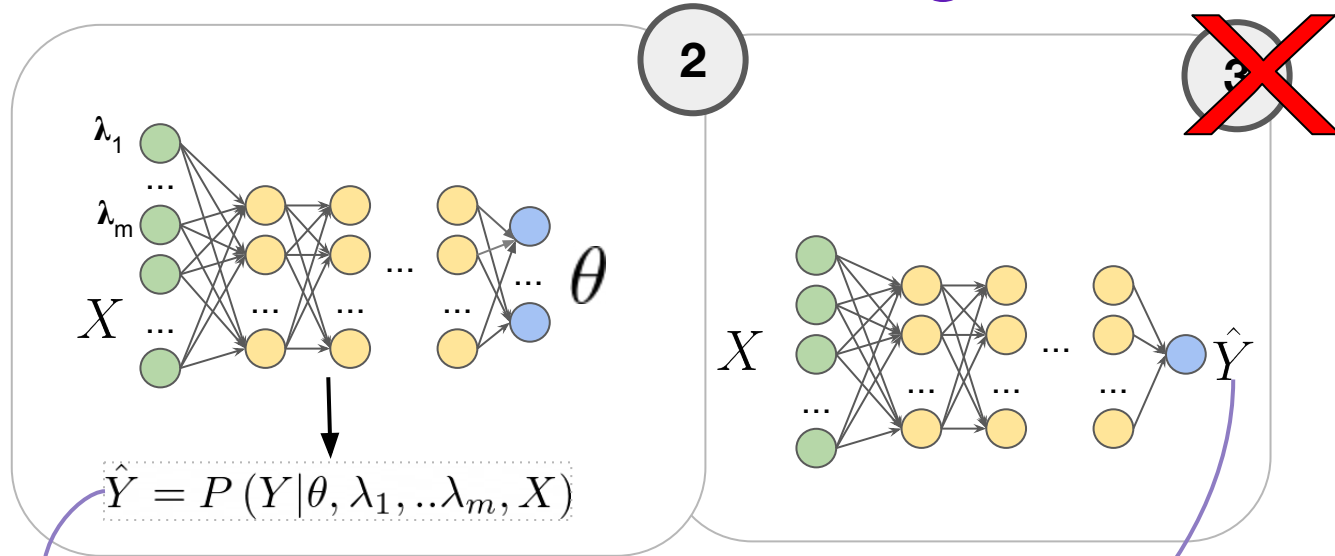
The probabilistic labels are used to train an end-model

# WeaSEL: Weakly Supervised End-to-end Learning

①

```
def lambdal(doc):  
    return(1 if 'rocket'  
           in doc  
           else ABSTAIN)
```

Users write heuristics



Maximize agreement

# Our contributions

- introduce WeaSEL: A flexible, end-to-end method for learning models from multiple sources of weak supervision.
- empirically demonstrate that the method is robust to adversarial sources and highly correlated heuristics.
- release an open-source system for arbitrary PyTorch end-models
  - <https://github.com/autonlab/weasel>
- our method outperforms, by as much as 6.1 F1 points, state-of-the-art latent label modeling approaches on 4 out of 5 benchmark datasets, and achieves state-of-the-art performance on a crowdsourcing dataset against methods specifically designed for this setting

**WeaSEL**

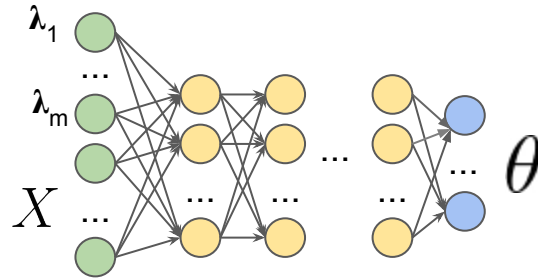


# WeaSEL: Weakly Supervised End-to-end Learning

1

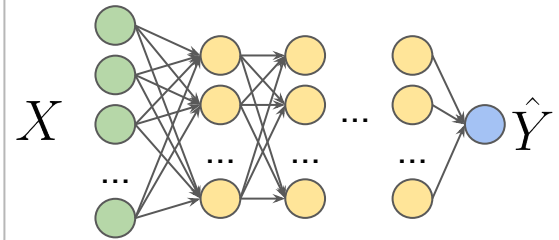
```
def lambda1(doc):  
    return(1 if `rocket`  
           in doc  
           else ABSTAIN)
```

Users write heuristics



End-model

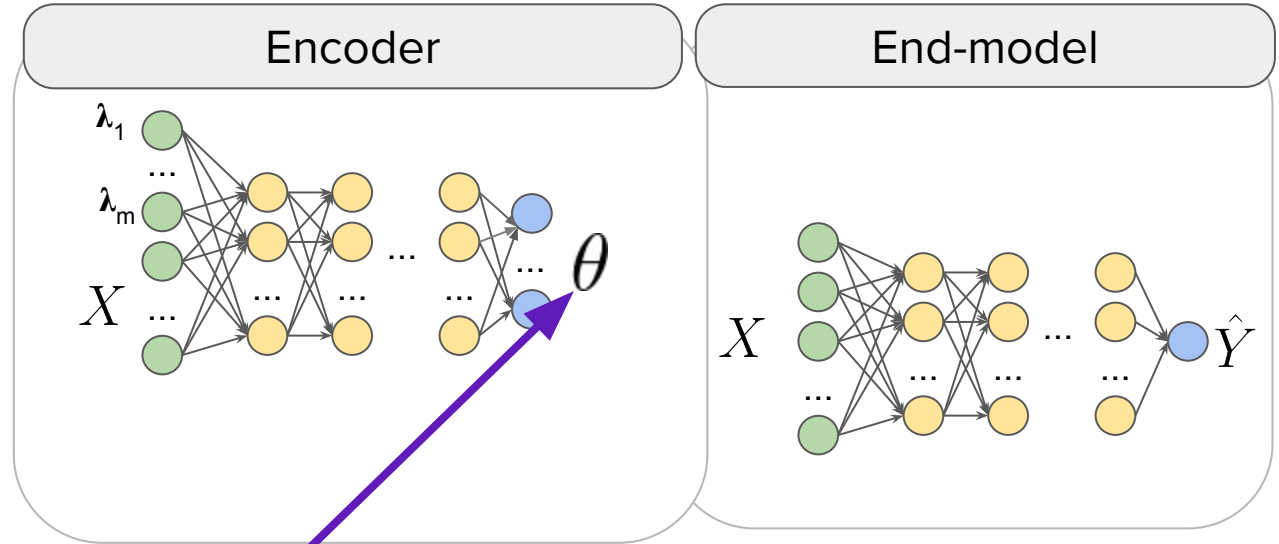
Same as before!



# WeaSEL: Weakly Supervised End-to-end Learning

```
def lambdal(doc):  
    return(1 if `rocket`  
           in doc  
           else ABSTAIN)
```

Users write heuristics



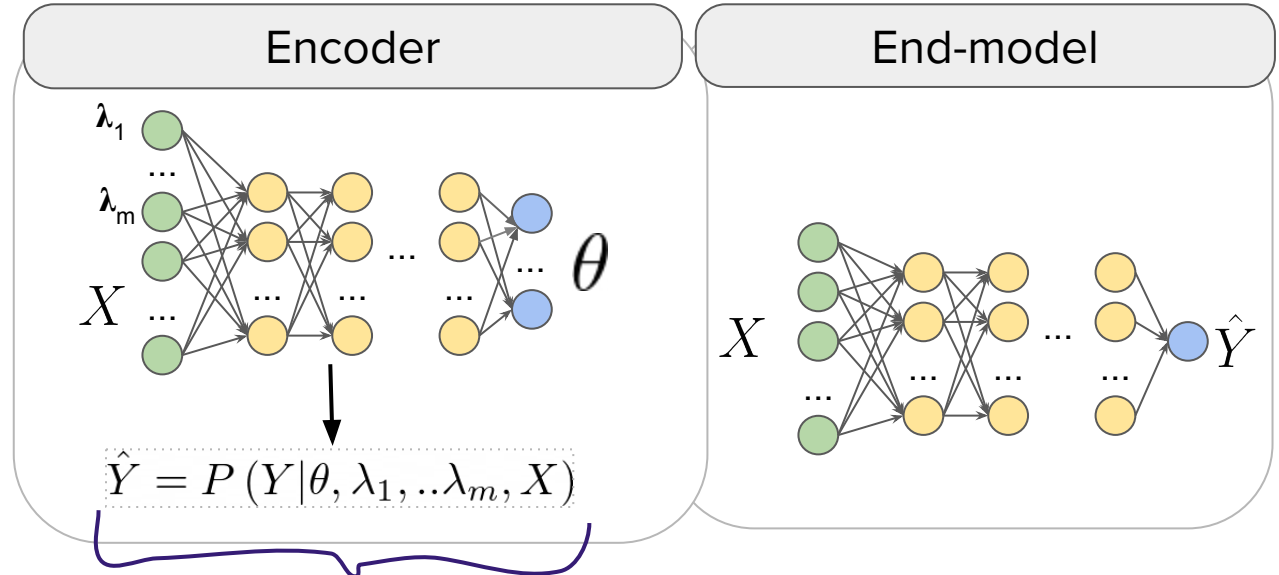
Encoder predicts the accuracy of each heuristic  
**Accuracy may vary across samples!**

# WeaSEL: Weakly Supervised End-to-end Learning

①

```
def lambdal(doc):  
    return(1 if `rocket`  
           in doc  
           else ABSTAIN)
```

Users write heuristics



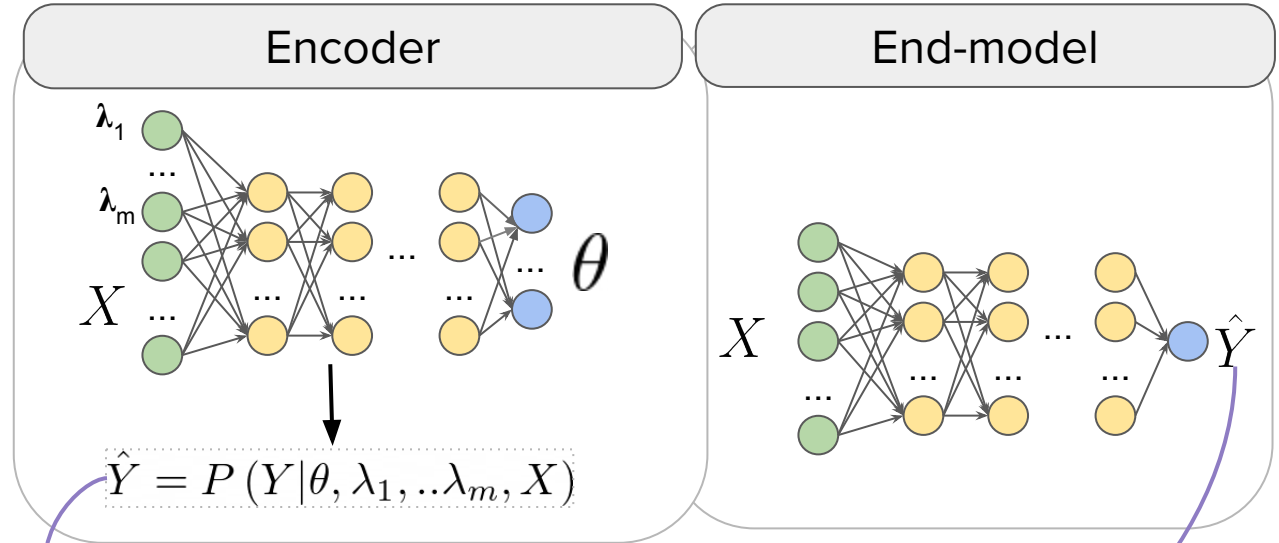
**Same weighted aggregation as before!**

# WeaSEL: Weakly Supervised End-to-end Learning

①

```
def lambda1(doc):  
    return(1 if `rocket`  
           in doc  
           else ABSTAIN)
```

Users write heuristics



Maximize agreement



# Key design choices

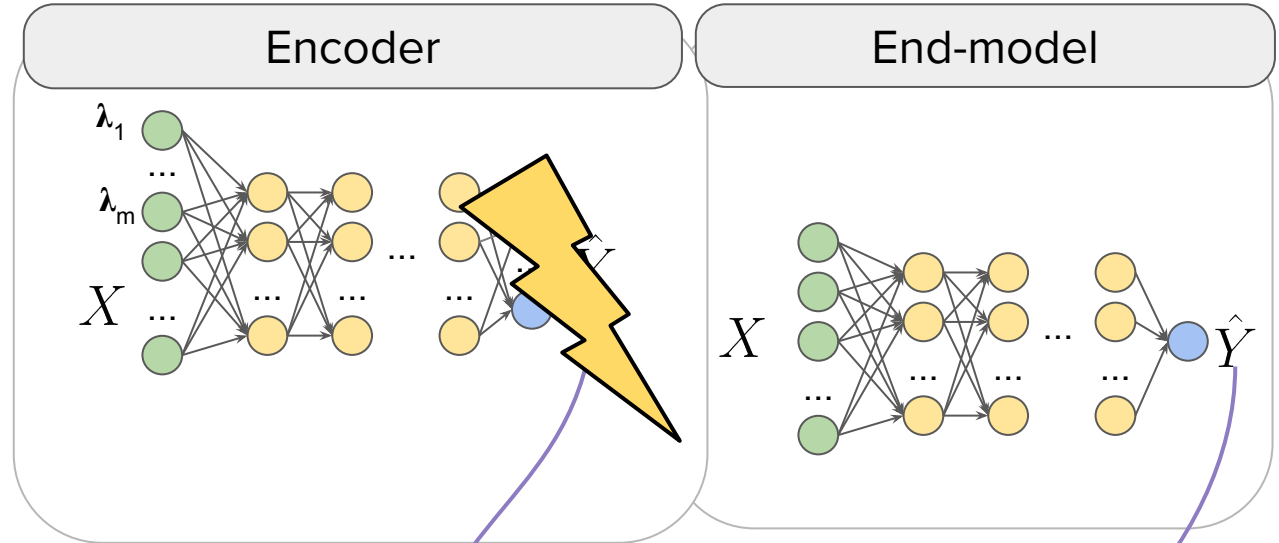


# Predict accuracy scores, not labels

①

```
def lambda1(doc):  
    return(1 if 'rocket'  
           in doc  
           else ABSTAIN)
```

Users write heuristics



Without labels -> Collapse

Maximize agreement

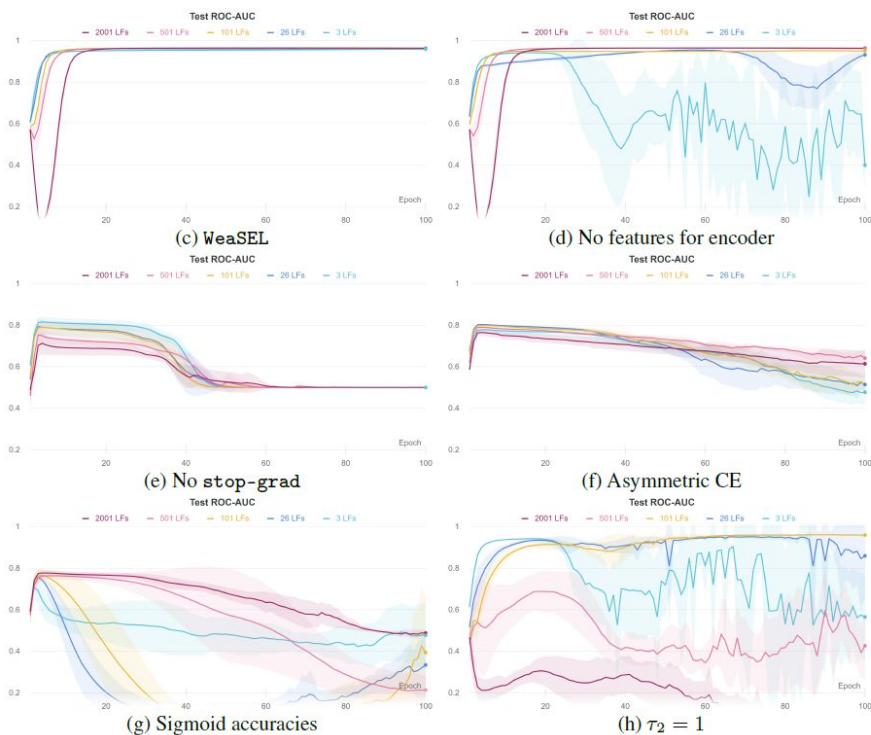


Figure 5: We start with a 100% accurate LF (i.e. ground truth labels) and plot test performances at each training epoch for a varying number of duplicates  $\in \{2, 25, 100, 500, 2000\}$  of a LF that is no better than a coin flip. Performances are averaged out over five random seeds, and the standard deviation is shaded. More details are given in [F.2.1].

Adding duplicated, “bad” heuristics (up to 2000) can break ablated versions of WeaSEL, but not WeaSEL

# Experiments



# WeaSEL is more robust against “bad” LFs

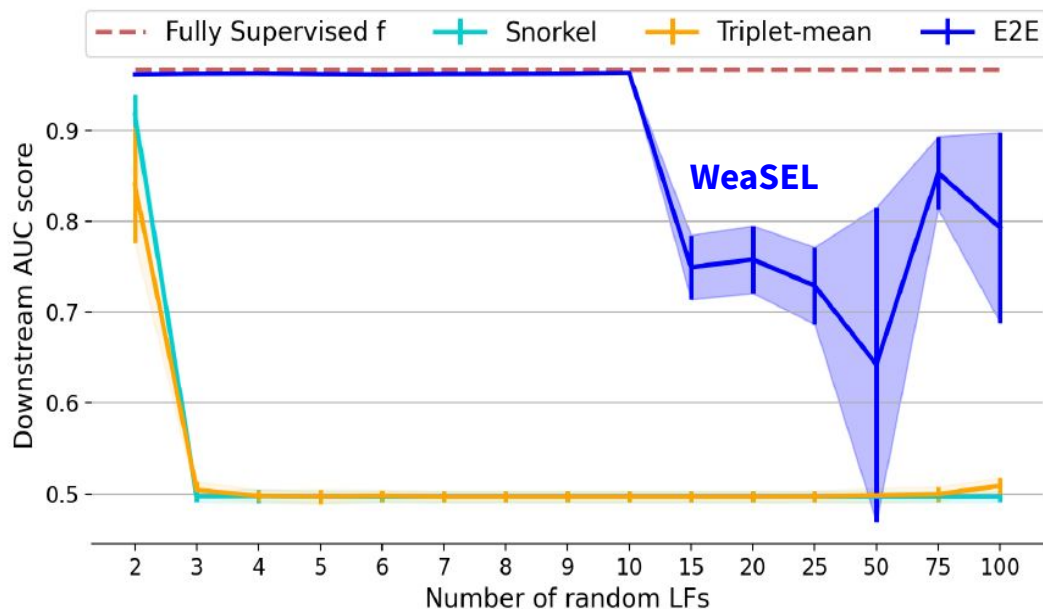


Figure 4: We start with a 100% accurate LF (i.e. ground truth labels) and incrementally add new, independent LFs that are no better than a random guess. WeaSEL recovers the performance of training directly on the ground truth labels (Fully Supervised  $f$ ), for up to 10 such randomly voting LFs that are independent of each other. The PGM-based prior work, rapidly degrades in performance (AUC  $\approx 0.5$ ) and is not able to recover any of the 100% accurate signal of the true-labels-LF, as soon as the LF set is corrupted by three or more random LFs. Performances are averaged out over five random seeds, and the standard deviation is shaded. For more details, see [F.2.2](#)

# Datasets

Table 3: Dataset details, where training, validation and test set sizes are  $N_{train}$ ,  $N_{val}$ ,  $N_{test}$  respectively, and  $f$  denotes the downstream model type. We also report the total coverage Cov. of all LFs, which refers to the percentage of training samples which are labeled by at least one LF (the rest is not used). For IMDB we used two different sets of labeling functions of sizes 12 and 136.

Dataset	#LFs	$N_{train}$	Cov. (in %)	$N_{val}$	$N_{test}$	$f$
Spouses	9	22,254	25.8	2811	2701	LSTM
BiasBios	99	12,294	81.8	250	12,044	MLP
IMDB	12	25k	88.0	250	24,750	MLP
IMDB	136	25k	83.1	250	24,750	MLP
Amazon	175	160k	65.5	500	39,500	MLP

# Results

Table 1: Test F1 performance of various label models over seven runs using different random seeds, are averaged out  $\pm$  standard deviation. The top 2 performance scores are highlighted as **First**, **Second**. Triplet-median [10] is not listed as it only converged for IMDB with 12 LFs (F1 =  $73.0 \pm 0.22$ ), and Spouses (F1 =  $48.7 \pm 1.0$ ). Sup. (Val. set) is the performance of the downstream model trained in a supervised manner on the labeled validation set. The rest are state-of-the-art latent label models. For reference, we also report the *Ground truth* performance of a fully supervised model trained on true training labels (which are unused by all other models, and not available for Spouses).

Model	Spouses (9 LFs)	ProfTeacher (99 LFs)	IMDB (136 LFs)	IMDB (12 LFs)	Amazon (175 LFs)
Ground truth	–	$90.65 \pm 0.29$	$86.72 \pm 0.40$	$86.72 \pm 0.40$	$92.93 \pm 0.68$
Sup. (Val. set)	$20.4 \pm 0.2$	$73.34 \pm 0.00$	$68.76 \pm 0.00$	$68.76 \pm 0.00$	$84.18 \pm 0.00$
Snorkel	$48.79 \pm 2.69$	$85.12 \pm 0.54$	<b><math>82.22 \pm 0.18</math></b>	<b><math>74.45 \pm 0.58</math></b>	$80.54 \pm 0.41$
Triplet	$45.88 \pm 3.64$	$74.43 \pm 10.59$	$75.36 \pm 1.92$	$73.15 \pm 0.95$	$75.44 \pm 3.21$
Triplet-Mean	<b><math>49.94 \pm 1.47</math></b>	$82.58 \pm 0.32$	$79.03 \pm 0.26$	$73.18 \pm 0.23$	$79.44 \pm 0.68$
Majority vote	$40.67 \pm 2.01$	<b><math>85.44 \pm 0.37</math></b>	$80.86 \pm 0.28$	$74.13 \pm 0.31$	<b><math>84.20 \pm 0.52</math></b>
WeaSEL	<b><math>51.98 \pm 1.60</math></b>	<b><math>86.98 \pm 0.45</math></b>	<b><math>82.10 \pm 0.45</math></b>	<b><math>77.22 \pm 1.02</math></b>	<b><math>86.60 \pm 0.71</math></b>

# Evaluation on a crowdsourcing-worker aggregation dataset

Table 2: Test accuracy scores on the crowd-sourced, multi-class LabelMe image classification dataset.

Model	Accuracy
Majority vote	79.23 $\pm$ 0.5
MBEM [26]	76.84 $\pm$ 0.4
DoctorNet [21]	81.31 $\pm$ 0.4
CrowdLayer [34]	82.83 $\pm$ 0.4
AggNet [1]	84.35 $\pm$ 0.4
MaxMIG [8]	<b>85.45 <math>\pm</math> 1.0</b>
Snorkel+CE	82.89 $\pm$ 0.7
WeaSEL+CE	82.46 $\pm$ 0.8
Snorkel+MIG	85.15 $\pm$ 0.8
WeaSEL+MIG	<b>86.36 <math>\pm</math> 0.3</b>



# Practical aspects

- Early-stopping on a small labeled validation set
- In binary classification: Tune decision threshold

# Limitations and future work

- When the end-model is slow to train, the process of finding a “final” set of heuristics is slowed down with WeaSEL → use Snorkel or less complex end-model
- How to completely avoid collapses? How to detect them without validation set?
- Use probabilistic heuristics!
- Applicable to regression?

# Conclusion

- We proposed WeaSEL, a new approach for end-to-end learning of neural network models for classification from, exclusively, multiple sources of weak supervision that streamlines prior latent variable models.
- Strong empirical performance and outperforms several state-of-the-art crowdsourcing methods on a crowdsourcing task.
- More robust to dependencies and correlations between the heuristics
- Works with discrete and probabilistic labeling functions and can utilize various neural network designs for probabilistic label generation.

# Thanks! :)



**End-to-End Weak Supervision,**

Salva Rühling Cachay, Benedikt Boecking, Artur Dubrawski,  
In Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS), 2021

**Code:** <https://github.com/autonlab/weasel>

**Contact:** [salvaruehling@gmail.com](mailto:salvaruehling@gmail.com)