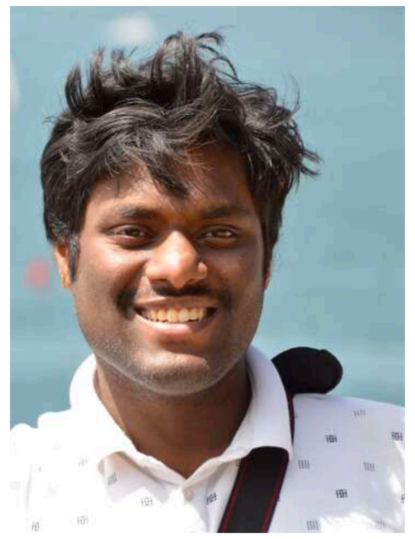
Confidence Functions for Auto-labeling

18 Mar, 2024

Harit Vishwakarma
CS Ph.D. Candidate

Yi (Reid) Chen
ECE Ph.D. Student



Srinath Namburi
CS Masters Student

Sui Jiet Tay
CS Undergrad

Advisors
Prof. Fred Sala
Prof. Ramya Korlakai Vinayak

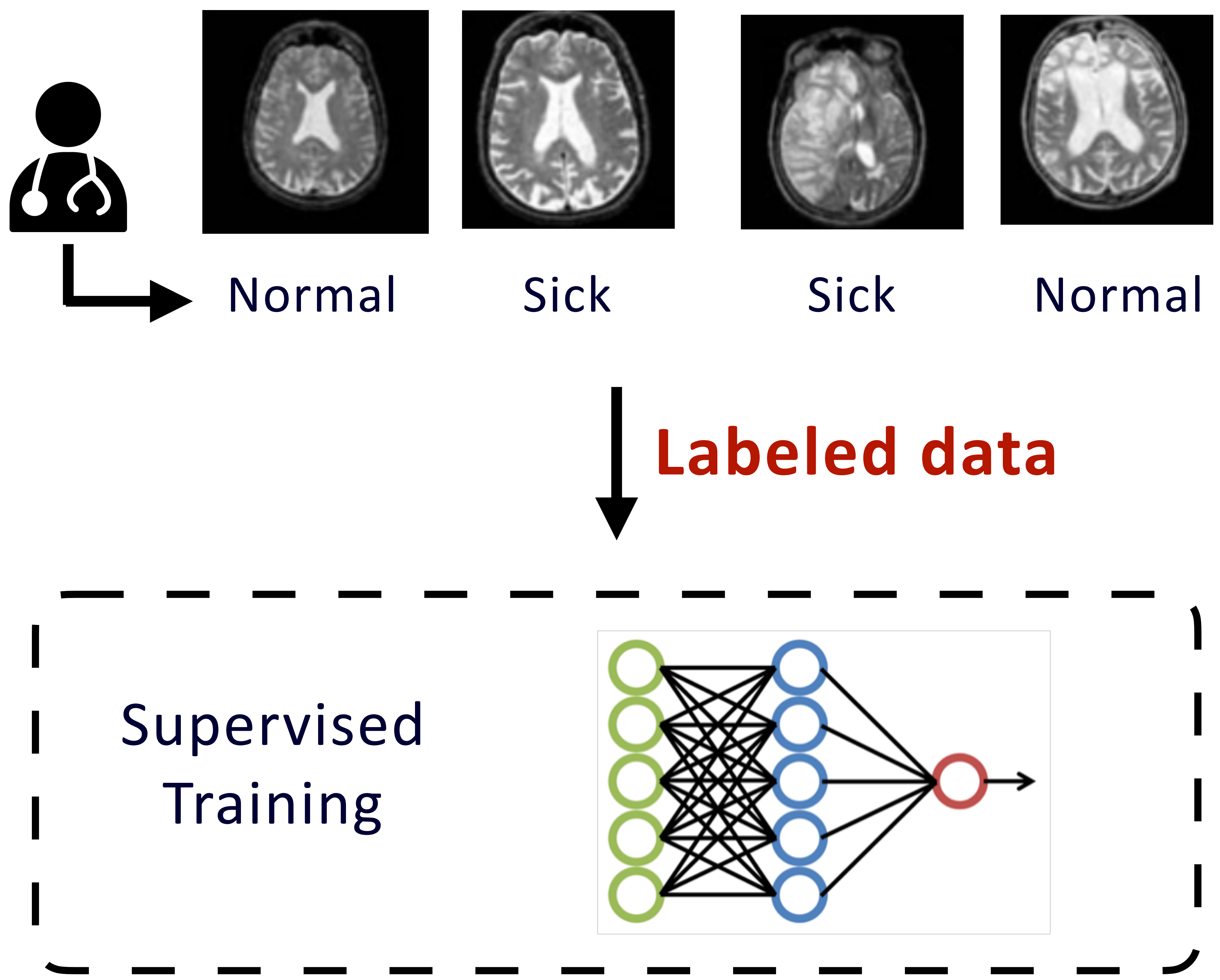
WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Roadmap

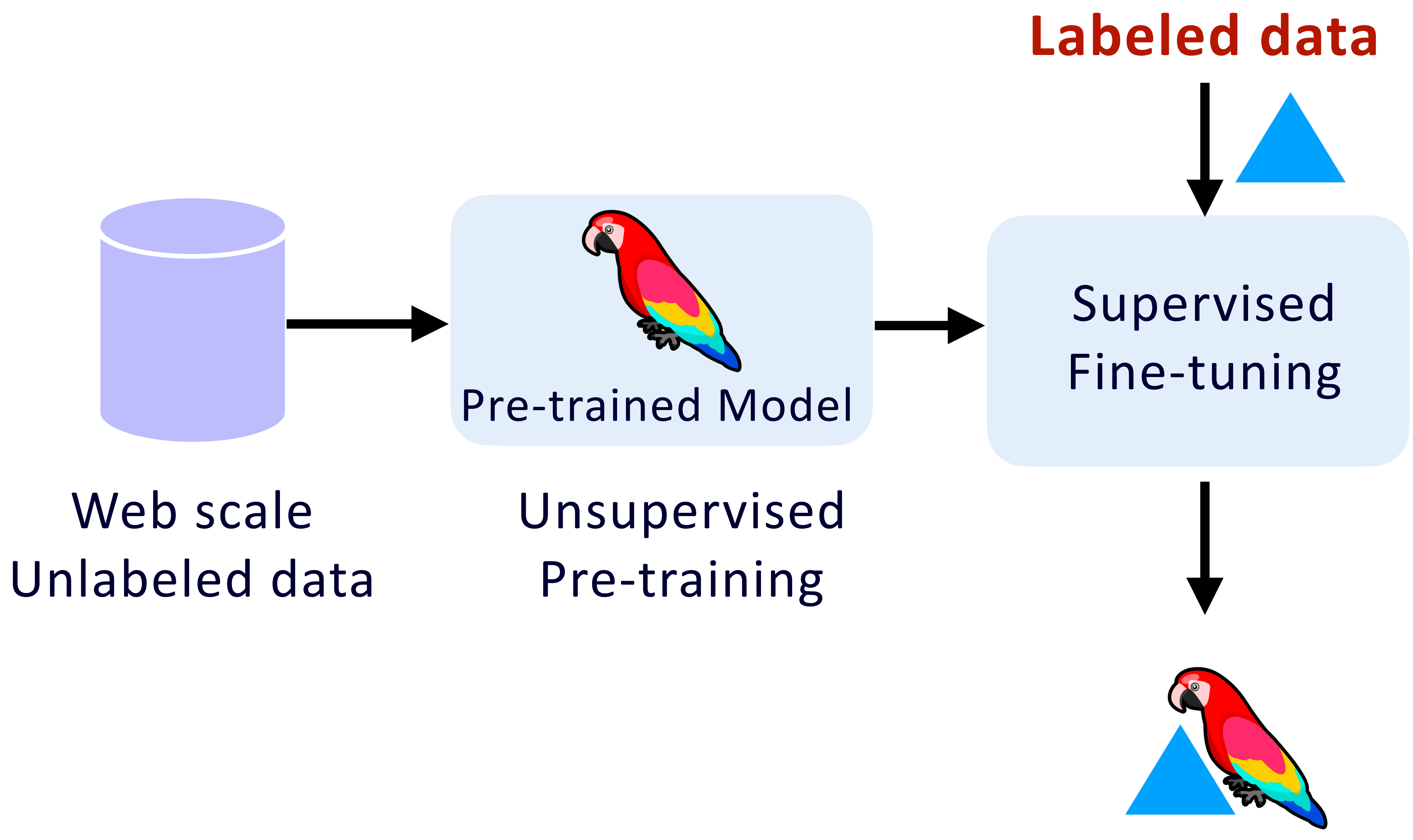

Why Auto-labeling?
Understanding Threshold-based Auto-labeling
Confidence Functions for TBAL
Empirical results

We need labeled data and often a lot of it!

Training from Scratch



Fine-tuning pre-trained models



Data Labeling costs a lot of time and money

Crowdsourcing is widely used to get labels

Wisdom of Crowd

amazon mechanical turk

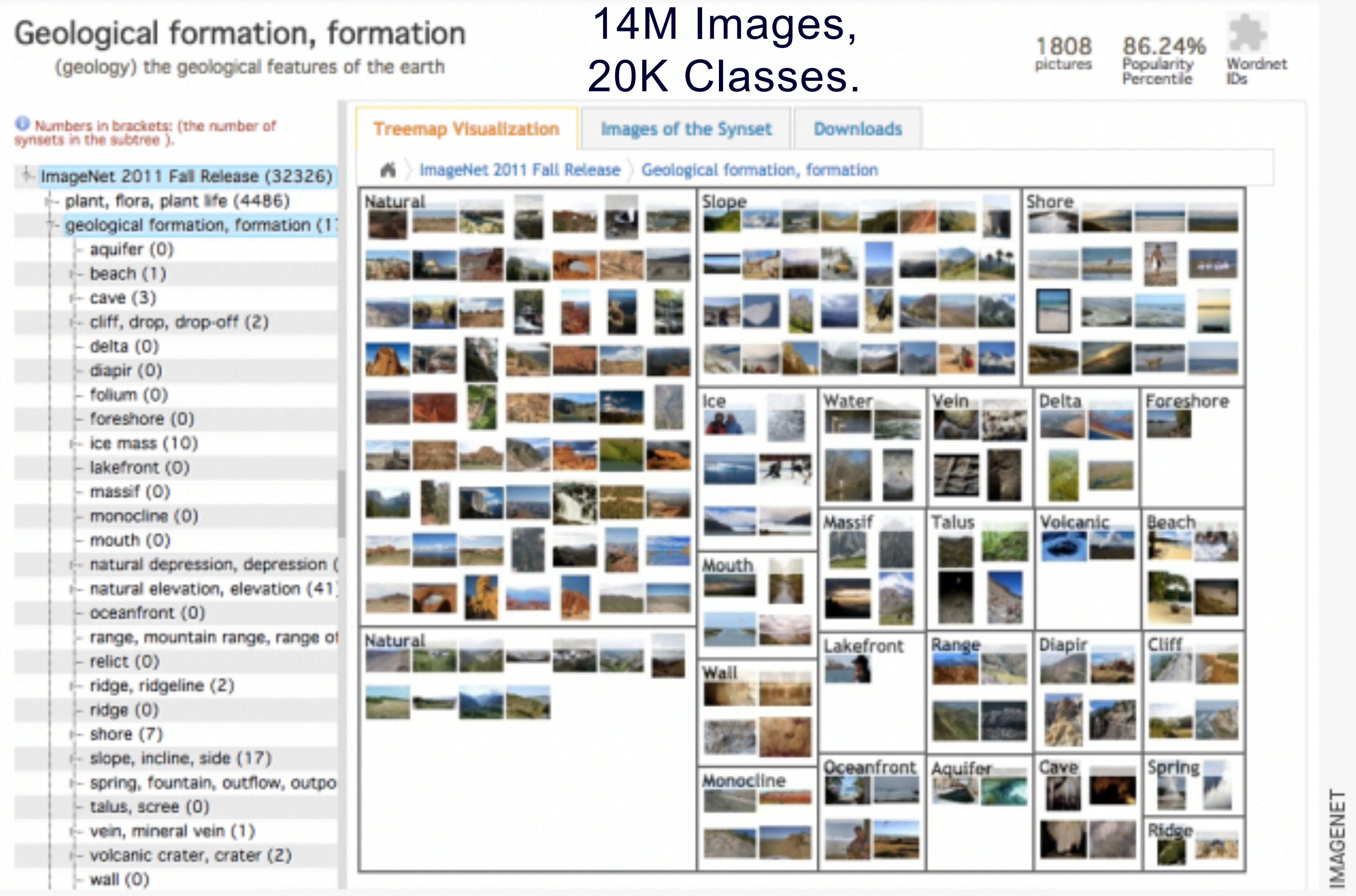
and many others…

Takes a lot of time and money to get labels.

IMAGENET Deng et. Al. 2009

Took multiple years and a lot of human effort

14M Images, 20K Classes.



A screenshot of the ImageNet database online

Re-create ImageNet using Mturk: $300,000.00

ML needs high-quality (accurately) labeled datasets.

+

Obtaining such datasets is costly.

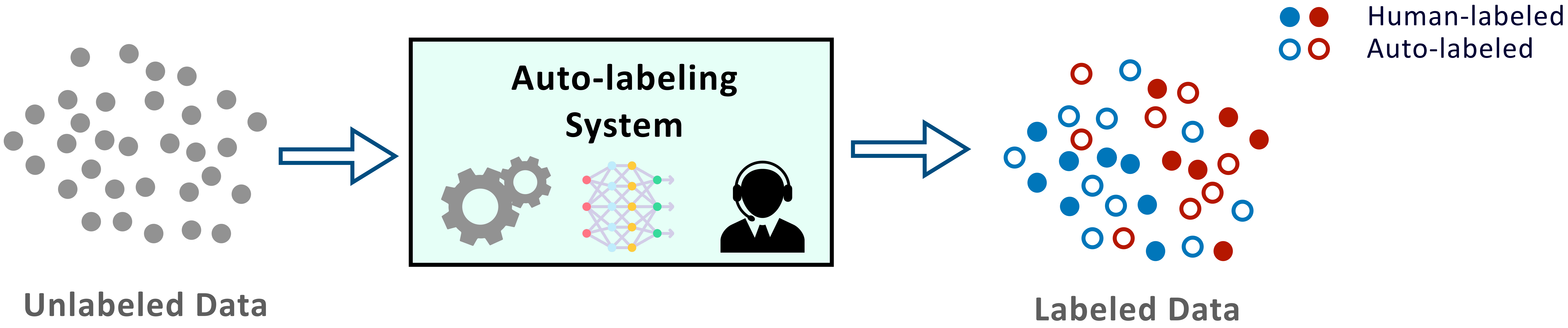
↓

Labeled data bottleneck

How to solve the labeled data bottleneck?

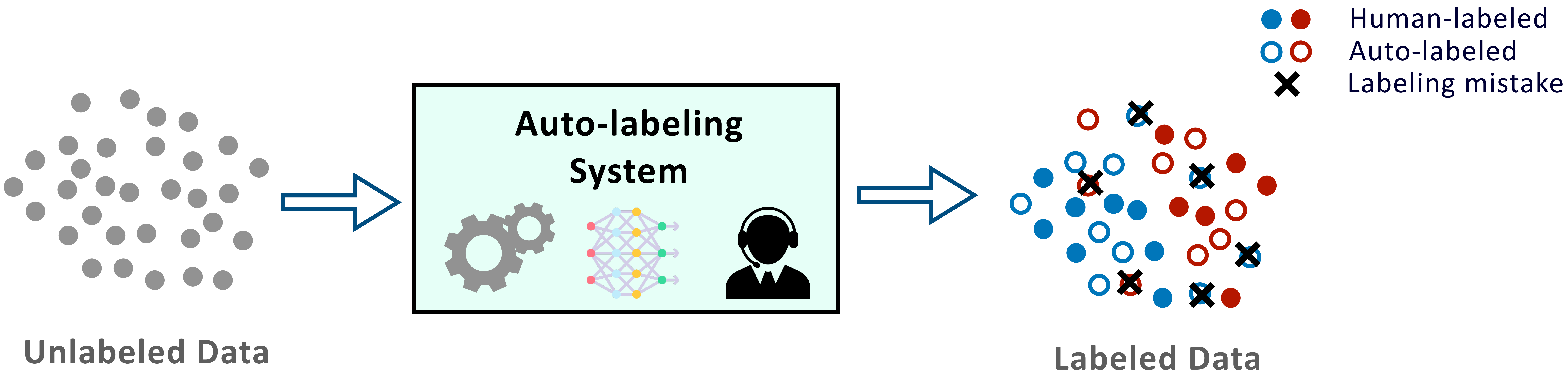
Auto-labeling

A broad set of techniques to create **labeled datasets** using **classifiers** and **human inputs.**



Auto-labeling

A broad set of techniques to create **labeled datasets** using **classifiers** and **human inputs.**



The output dataset may have labeling errors.

The impact of these errors is significant:

a. Datasets are static and have long shelf-life.

b. Multiple models are trained on the same dataset.

We need strict control over the errors in the dataset.

Threshold-based Auto-labeling (TBAL)

can provide such control.

Combines ideas from Selective Classification and Transductive Learning.

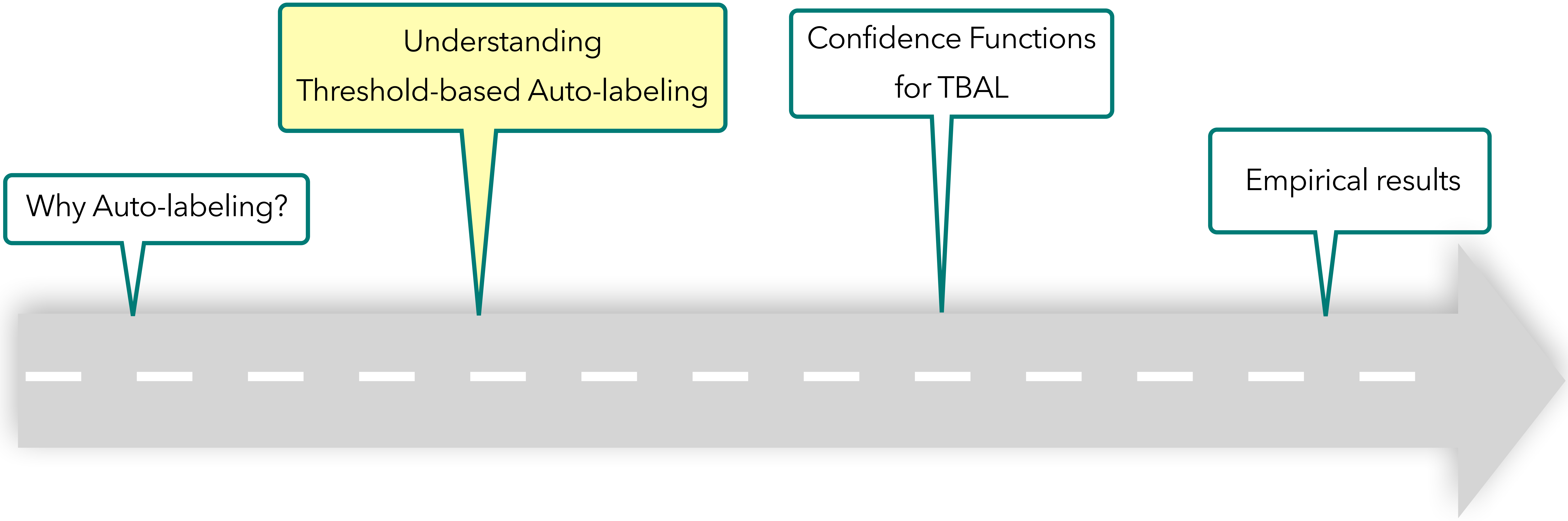
Inspired by Amazon Sagemaker Groundtruth

A commercial system getting used in practice

aws Amazon SageMaker Ground Truth

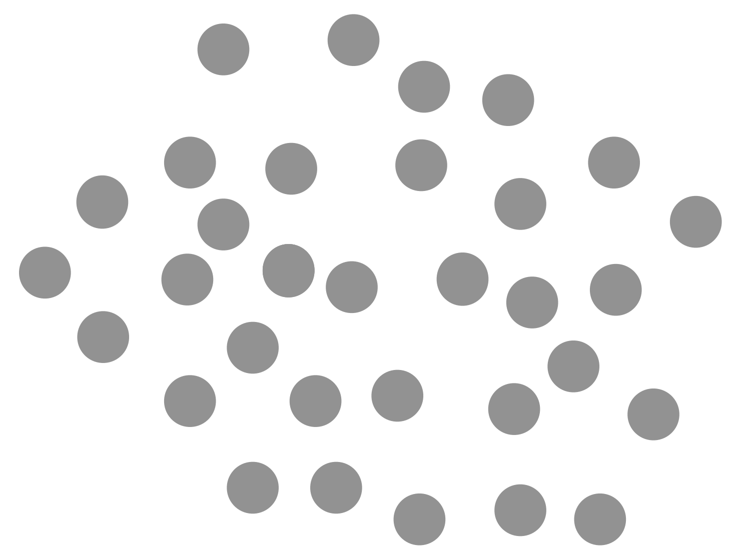
But our understanding is limited!

Roadmap

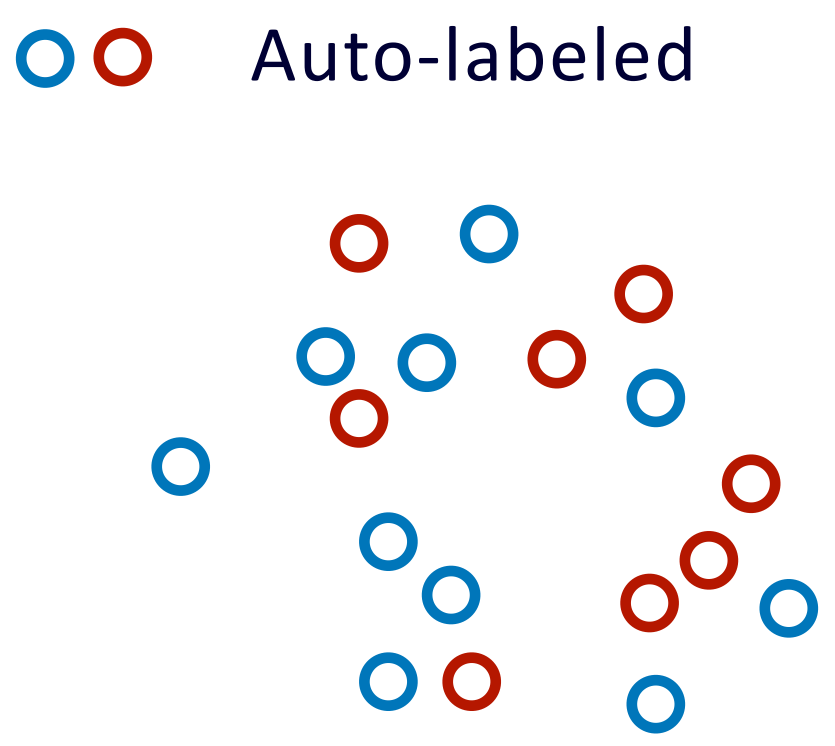

Why Auto-labeling?
Understanding
Threshold-based Auto-labeling
Confidence Functions
for TBAL
Empirical results

Understanding Threshold-based Auto-labeling

Quality and Quantity of Auto-labeled Data

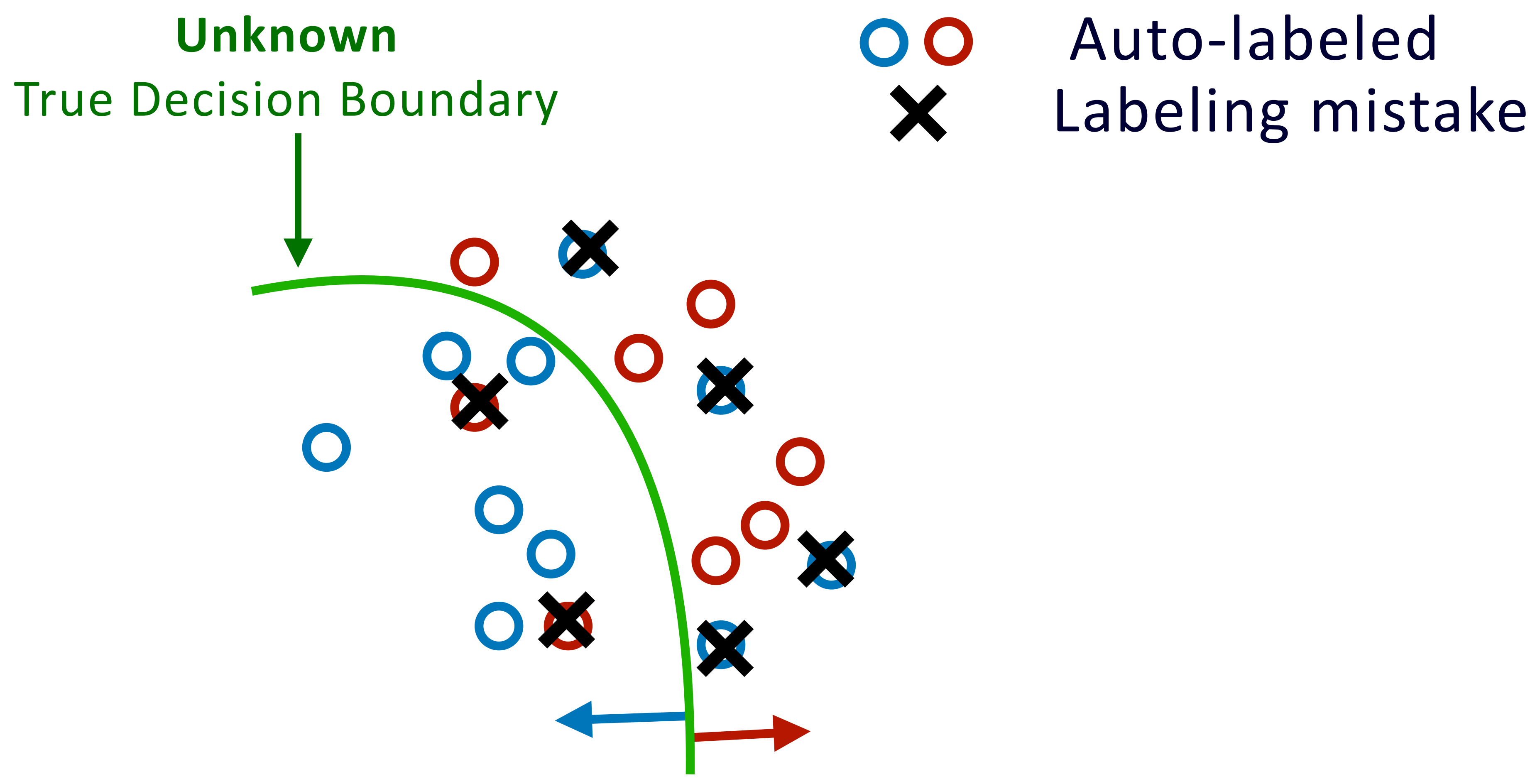


N Number of **unlabeled** points



A Set of **auto-labeled** points

N_a Number of auto-labeled points



M_a Number of labeling mistakes

Quantity

Auto-labeling Coverage

$$\hat{\mathcal{P}} = \frac{N_a}{N}$$

Good Stuff
maximize this

Quality

Auto-labeling Error

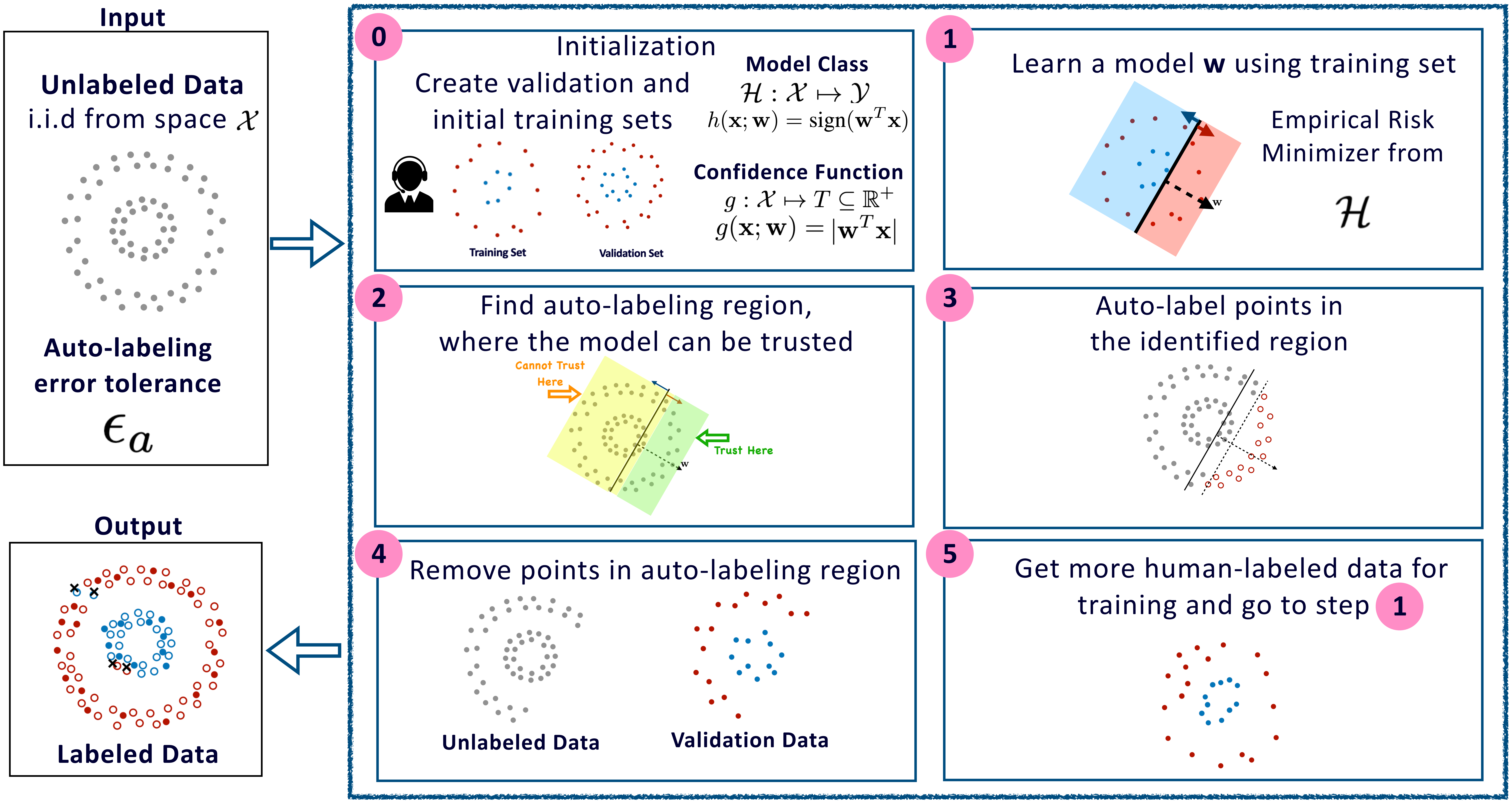
$$\widehat{\mathcal{E}} = \frac{M_a}{N_a}$$

Bad Stuff
minimize this

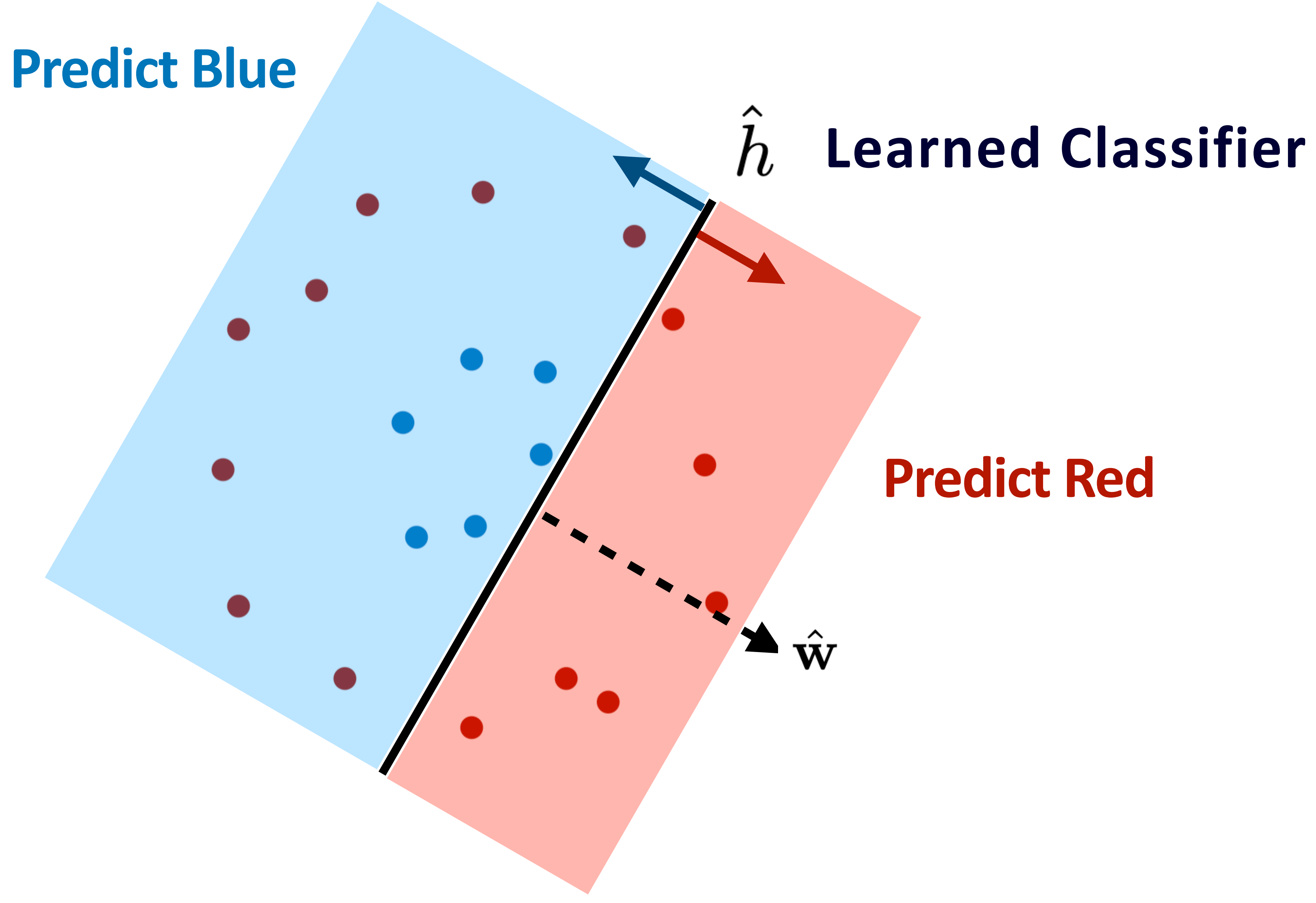
There are Trade-offs between Coverage and Error

Need to guarantee $\leq \epsilon_a$

Threshold-based Auto-labeling Workflow (TBAL)

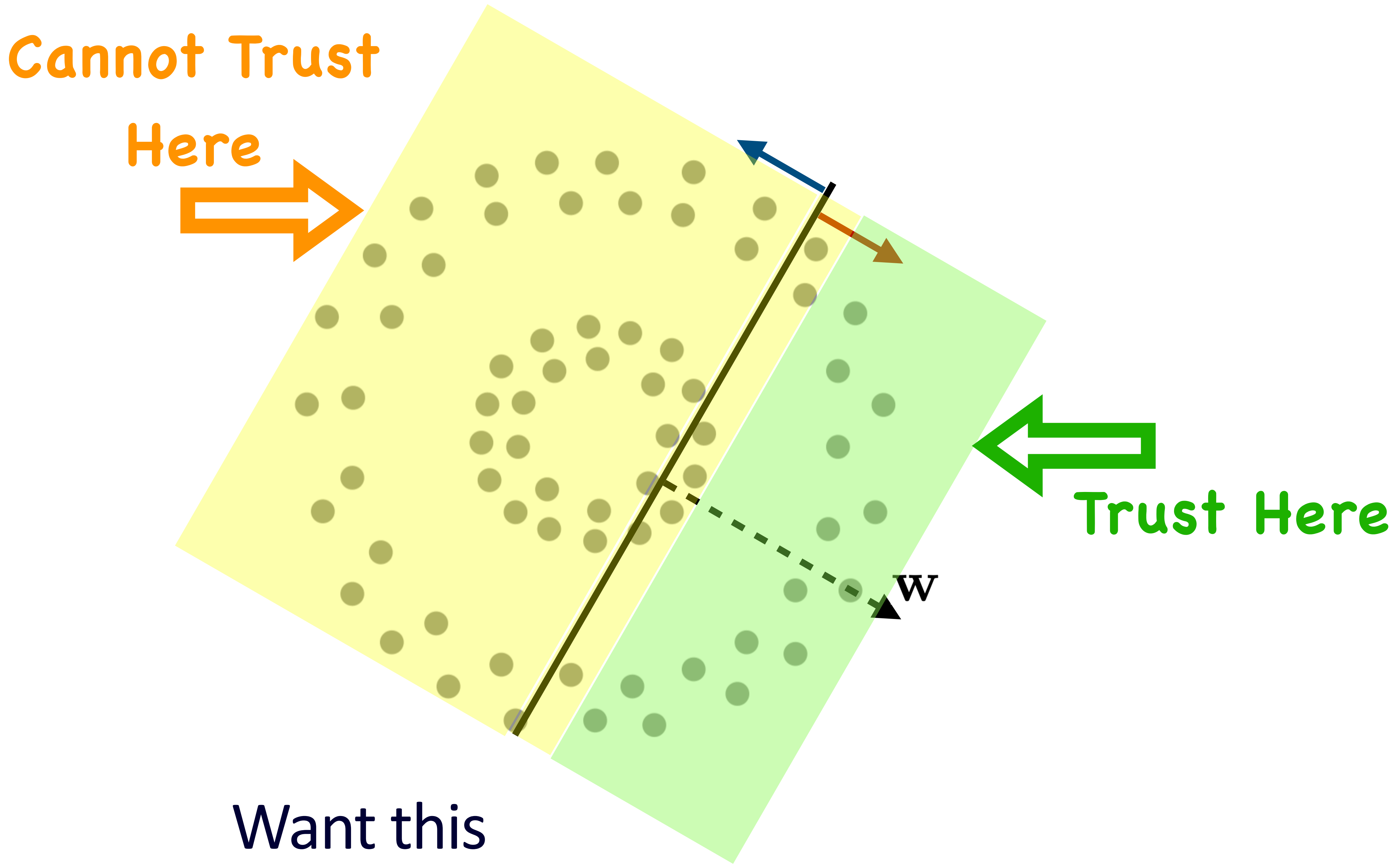


TBAL Workflow: Step 2 Find the Auto-labeling region



Learned Model

Only predict where the classifier is accurate



Want this

Auto-label only where the model is accurate (or trustworthy)

Selective Classification (SC)

El-Yaniv & Weiner, 2010; Cortes, Desalvo, Mohri 2016; Gelbhart & El-Yaniv 2019; Fisch, Jakkola et al. 2022;

Use **validation data** and **confidence scores** to find the auto-labeling region.

TBAL Workflow: Step 2

Find the Auto-labeling region

On the **validation data** we know where the **classifier** is **correct** and **incorrect**.

Confidence Function

$$\textit{confidence function } g : \mathcal{X} \rightarrow \Delta^k$$

Confidence in predictions of the classifier

Depends on h but drop it for convenience

Predicted label/class

$$\hat{y} := h(\mathbf{x})$$

Confidence Score

$$g(\mathbf{x})[\hat{y}]$$

Softmax Score

Multi-class setting

0.02	0.06	0.02	**0.9**
0	1	2	3

$\hat{y} = 3 \qquad g(\mathbf{x})[\hat{y}] = 0.9$

Margin Scores

Binary classes (Linear)

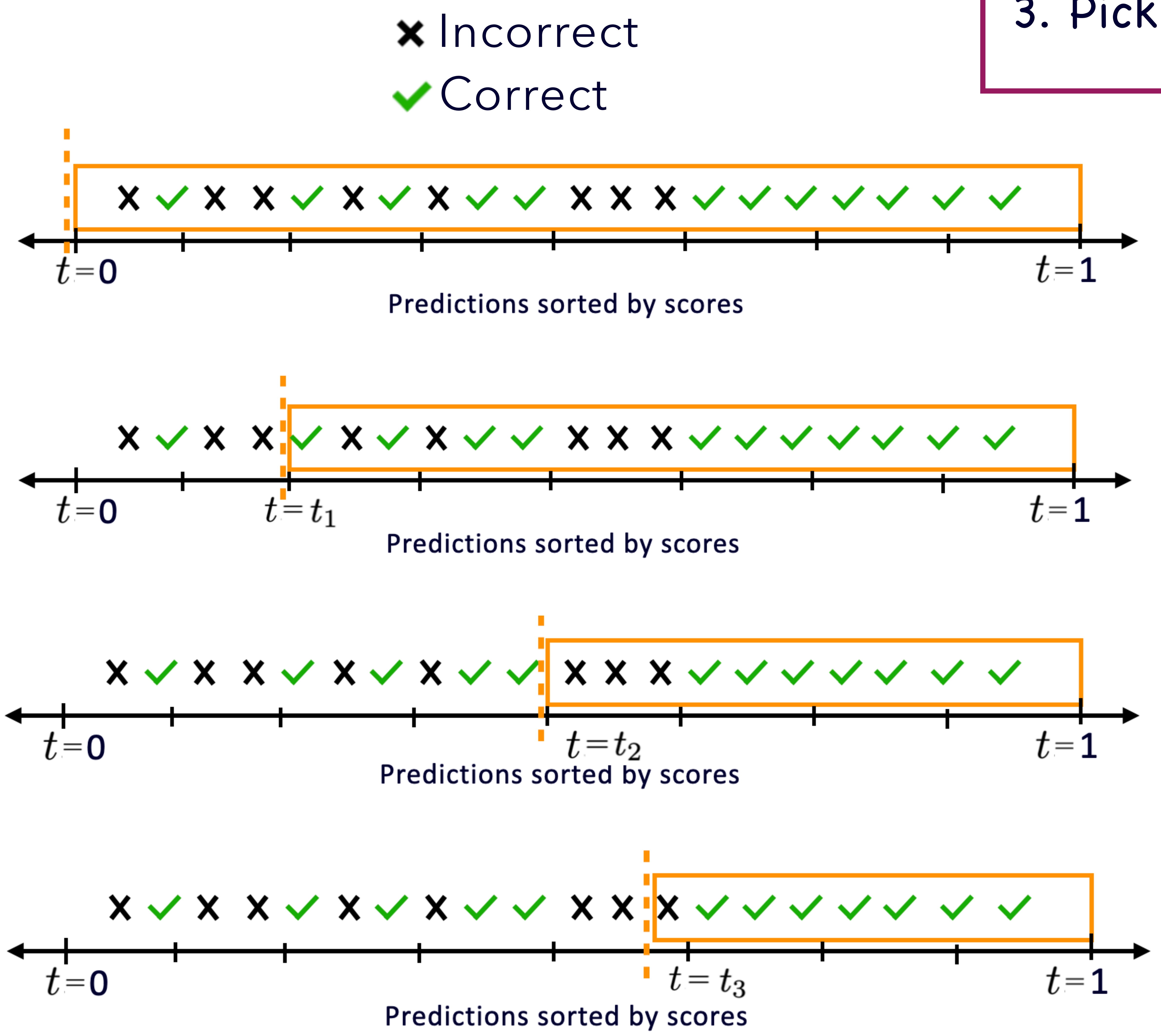
$-\mathbf{w}^T\mathbf{x}$	$\mathbf{w}^T\mathbf{x}$
0	1

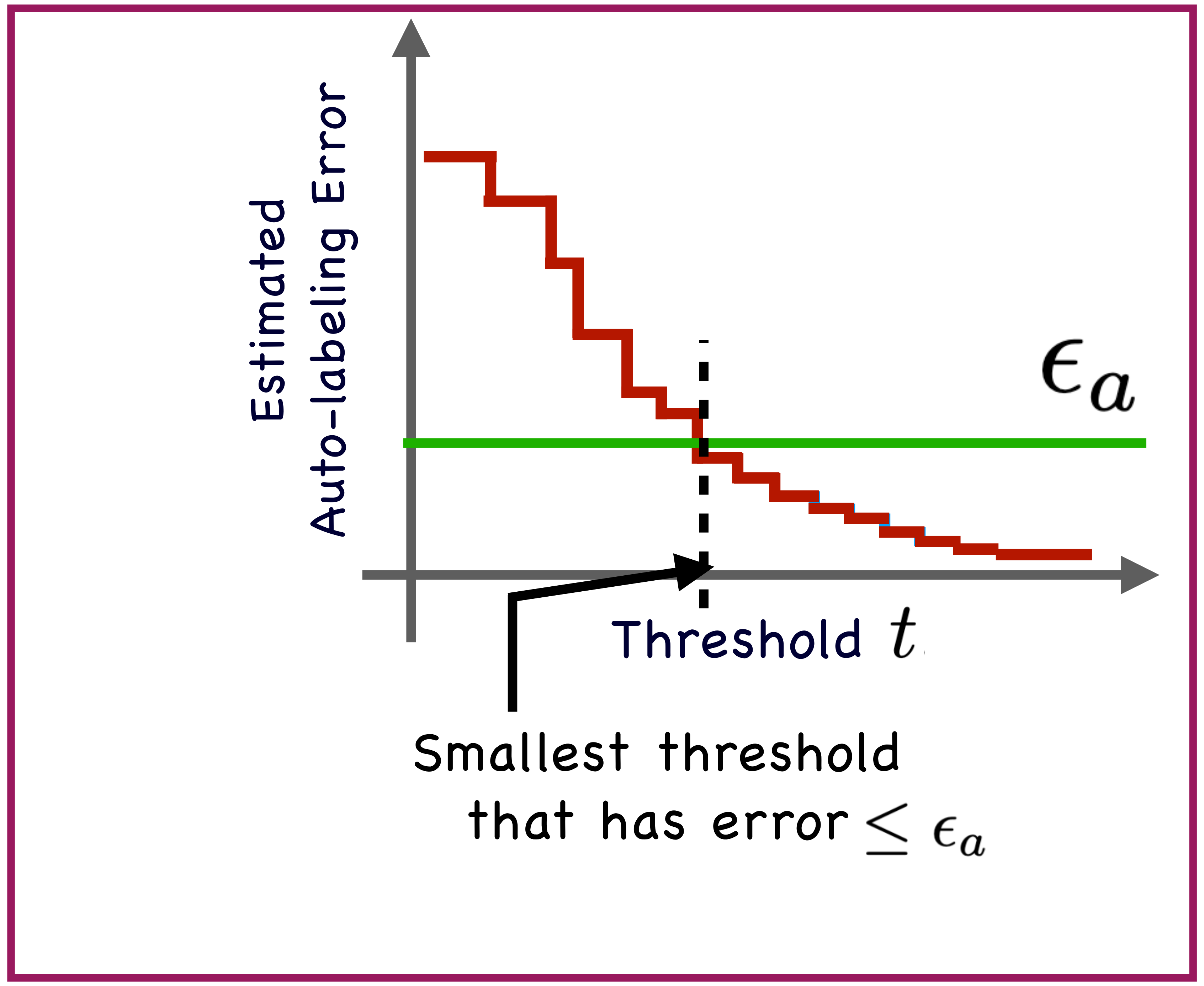
$\hat{y} = 1 \qquad g(\mathbf{x})[\hat{y}] = \mathbf{w}^T\mathbf{x}$

TBAL Workflow: Step 2

Find the Auto-labeling region

1. Order points based on the **Confidence scores.**
2. Estimate the auto-labeling error at several thresholds.
3. Pick the smallest threshold having error at most ϵ_a





The hope

We studied TBAL and the role of validation data set

Promises and Pitfalls of Threshold-based Auto-labeling

Harit Vishwakarma
hvishwakarma@cs.wisc.edu
University of Wisconsin-Madison

Heguang Lin
hglin@seas.upenn.edu
University of Pennsylvania

Frederic Sala
fredsala@cs.wisc.edu
University of Wisconsin-Madison

Ramya Korlakai Vinayak
ramya@ece.wisc.edu
University of Wisconsin-Madison

NeurIPS, 2023 (Spotlight)

More details in the paper.

https://arxiv.org/abs/2211.12620v2

Long talk on
MLOpt Youtube Channel
https://www.youtube.com/@UWMadisonMLOPTIdeaSeminar

TL;DR

Theoretical and empirical results,

TBAL can produce accurately labeled dataset,
provided there is sufficient validation data.

We also observed a blocker/spoilsport.

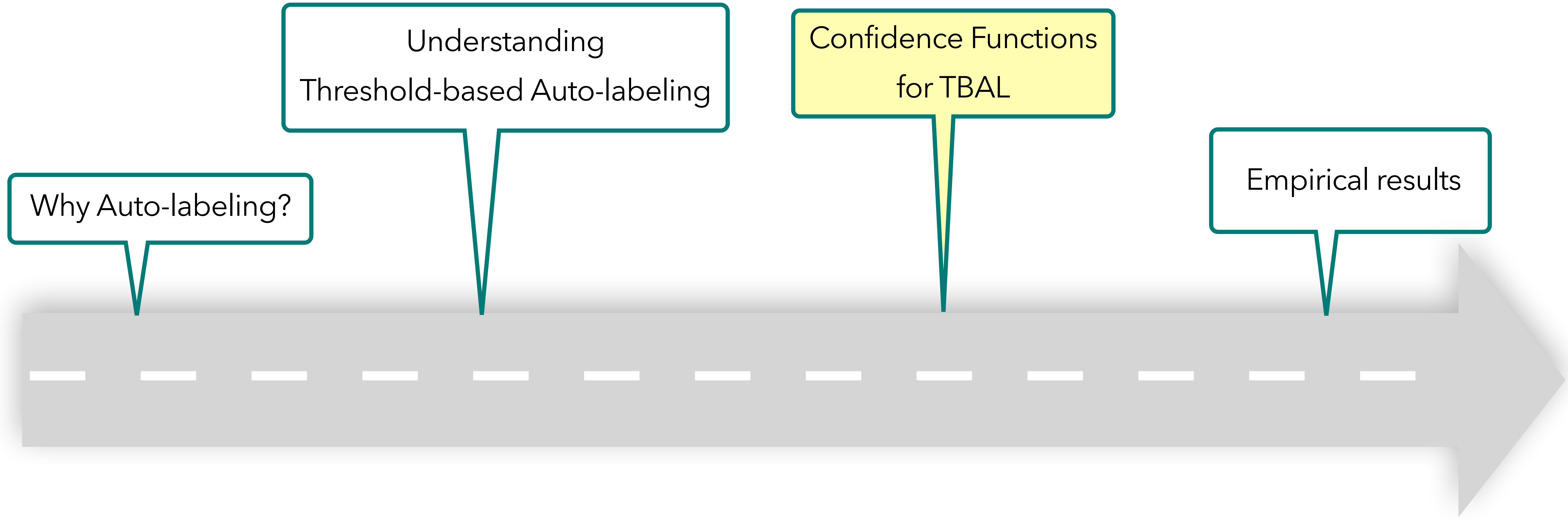
We had models with around **50% test accuracy for a 10 class** prediction problem.

But TBAL could get **very little coverage**, irrespective of the validation data size.

Confidence scores were the culprit.

So we started thinking about confidence functions for TBAL.

Roadmap



Why Auto-labeling?
Understanding Threshold-based Auto-labeling
Confidence Functions for TBAL
Empirical results

Confidence Functions for Auto-labeling

Pearls from Pebbles: Improved Confidence Functions for Auto-labeling

Harit Vishwakarma
hvishwakarma@cs.wisc.edu

Reid (Yi) Chen
reid.chen@wisc.edu

Sui Jiet Tay
sstay2@wisc.edu

Satya Sai Srinath Namburi
sgnamburi@cs.wisc.edu

Frederic Sala
fredsala@cs.wisc.edu

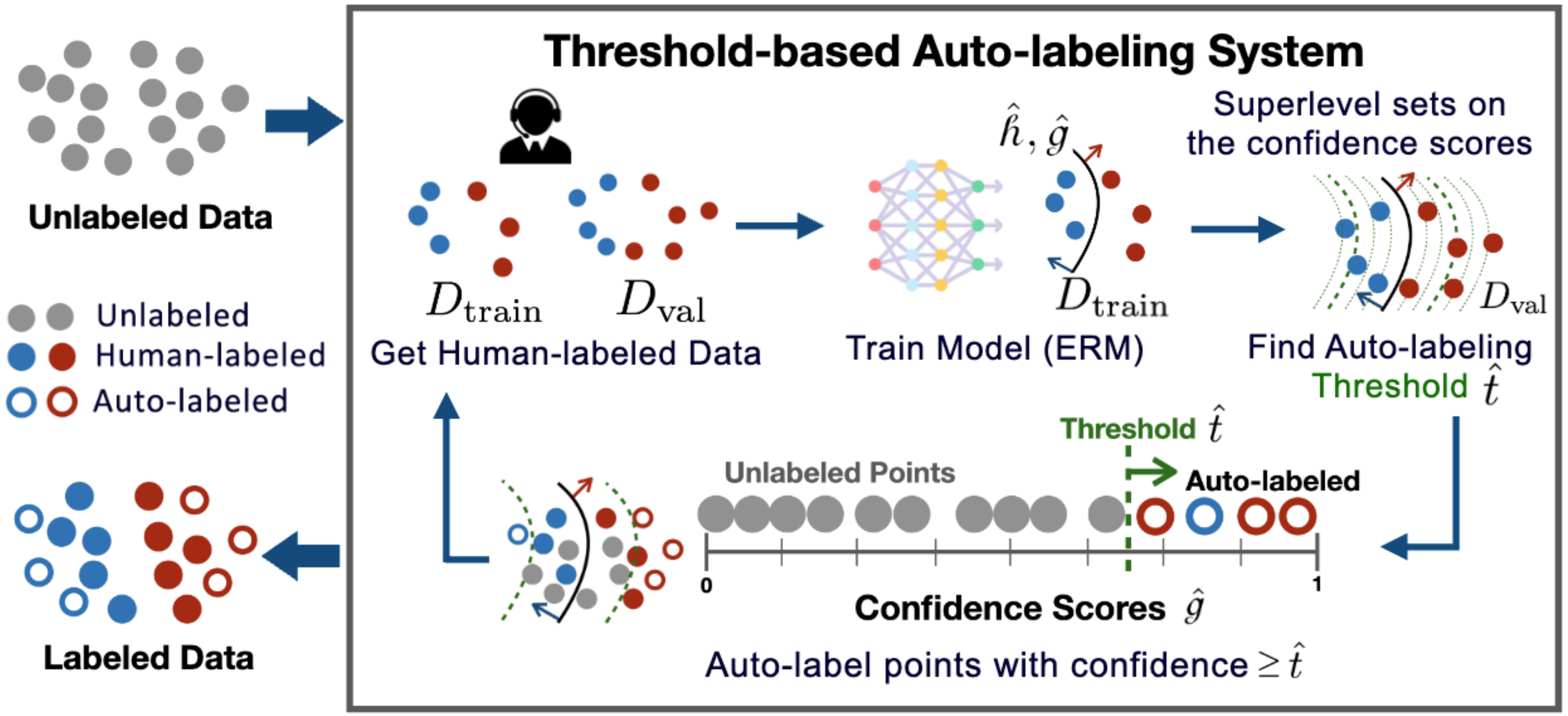
Ramya Korlakai Vinayak
ramya@ece.wisc.edu

University of Wisconsin-Madison, WI, USA

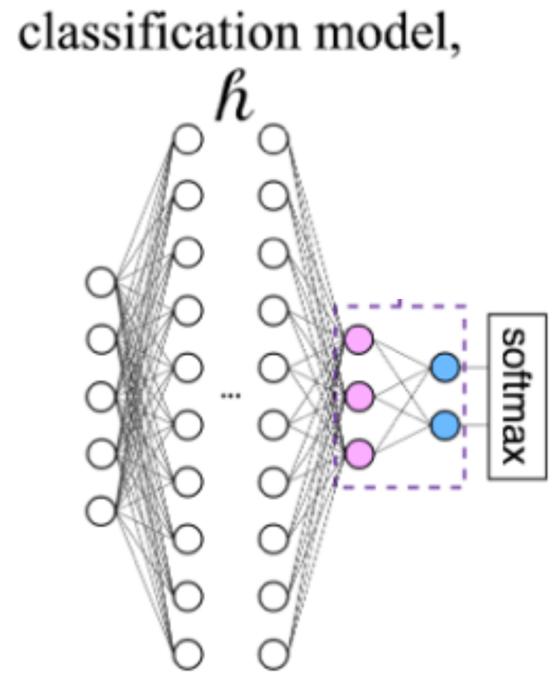
https://arxiv.org/pdf/2404.16188

Confidence Functions for TBAL

Recap of TBAL workflow



Standard Training Procedure (Vanilla)



Pick your favorite Neural Net
(MLP, CNN, RNN, Transformer, ...)

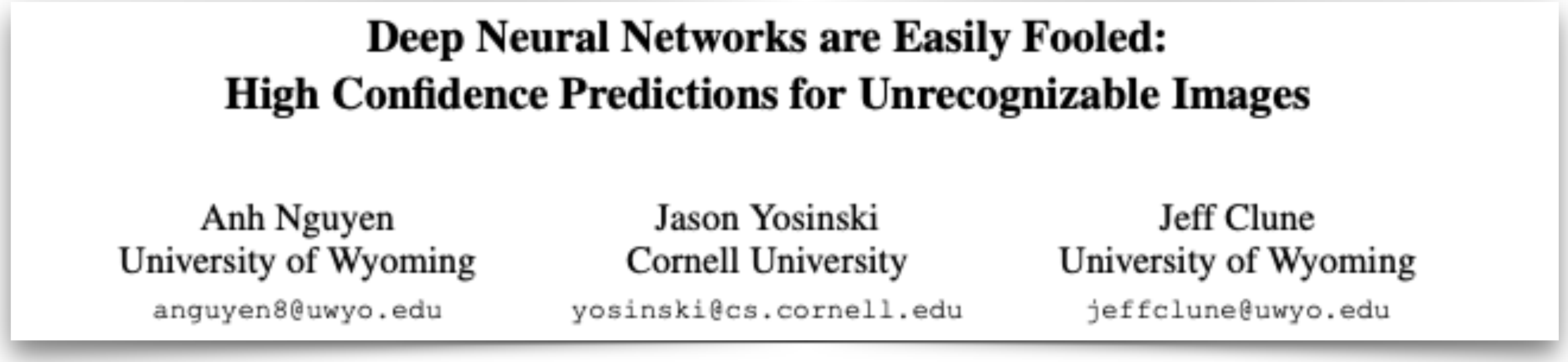
Minimize the Cross-Entropy Loss on training data using SGD

Use softmax scores for auto-labeling

Standard training procedure and softmax scores can be bad for auto-labeling

Prone to the overconfidence problem

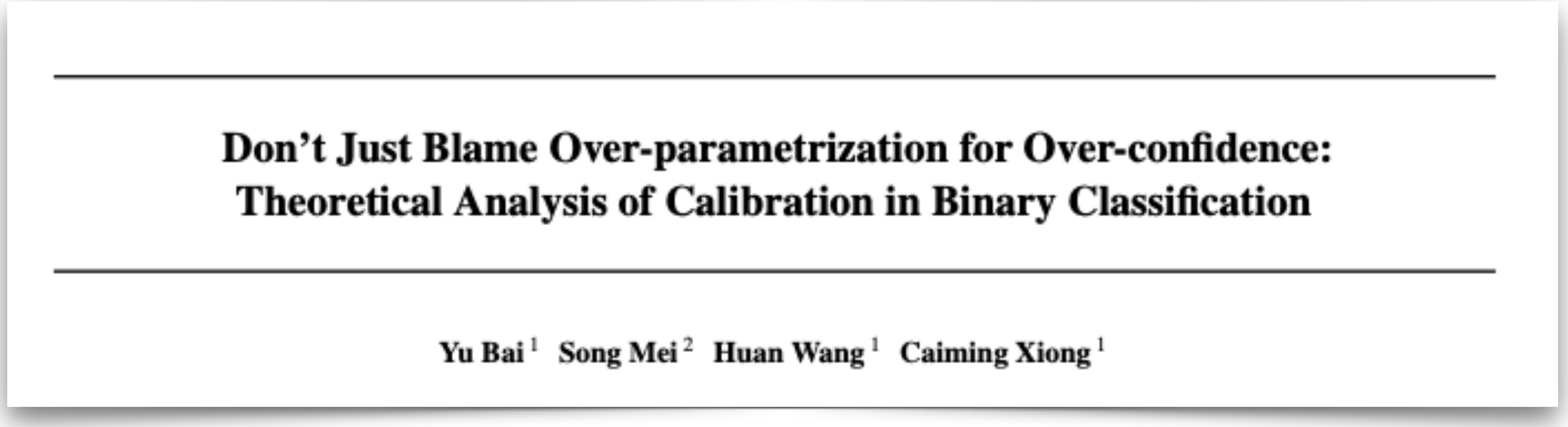
High scores even for incorrect predictions


Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images

Anh Nguyen
University of Wyoming
anguyen8@uwyo.edu

Jason Yosinski
Cornell University
yosinski@cs.cornell.edu

Jeff Clune
University of Wyoming
jeffclune@uwyo.edu


Don't Just Blame Over-parametrization for Over-confidence: Theoretical Analysis of Calibration in Binary Classification

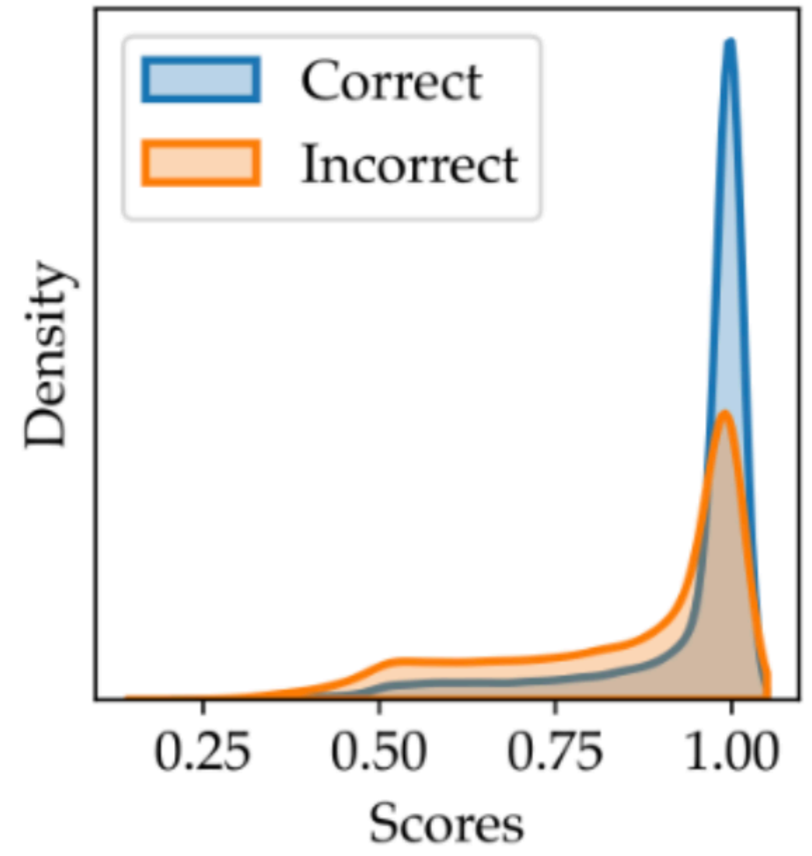
Yu Bai[1] Song Mei[2] Huan Wang[1] Caiming Xiong[1]

Szegedy et al. 2014; Nguyen et al. 2015; Hendricks & Gimpel 2017; Guo et al. 2017; Hein et al. 2018, Bai et al. 2021

Experiment

Run 1 round of TBAL

Data	CIFAR-10
Model	CNN model (5.8 M parameters)
Training data	4000 points drawn randomly
Validation data	1000 points drawn randomly
Error Tolerance	5%



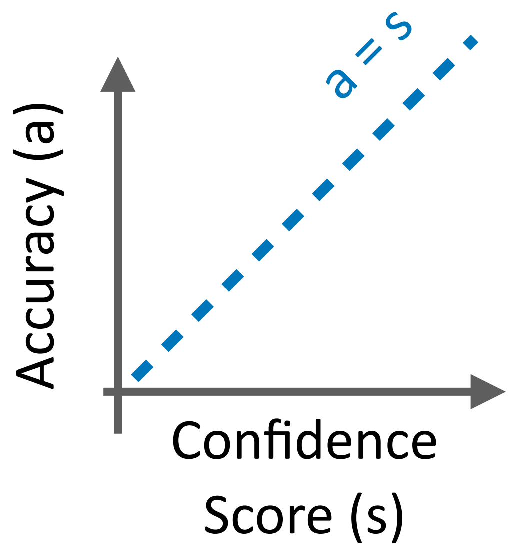
Kernel Density Estimate(KDE) of scores on the remaining unlabeled data

Test Accuracy	55%
Coverage	2.9%
Auto-labeling Error	10.1%

Ad-hoc Methods to Reduce Overconfidence may not help either

Calibration

Points where score is t, the accuracy on those points should be t



On Calibration of Modern Neural Networks

Chuan Guo [*1] Geoff Pleiss [*1] Yu Sun [*1] Kilian Q. Weinberger [1]

Verified Uncertainty Calibration

Ananya Kumar, Percy Liang, Tengyu Ma

TOP-LABEL CALIBRATION AND MULTICLASS-TO-BINARY REDUCTIONS

Chirag Gupta & Aaditya Ramdas

Cut your Losses with Squentropy

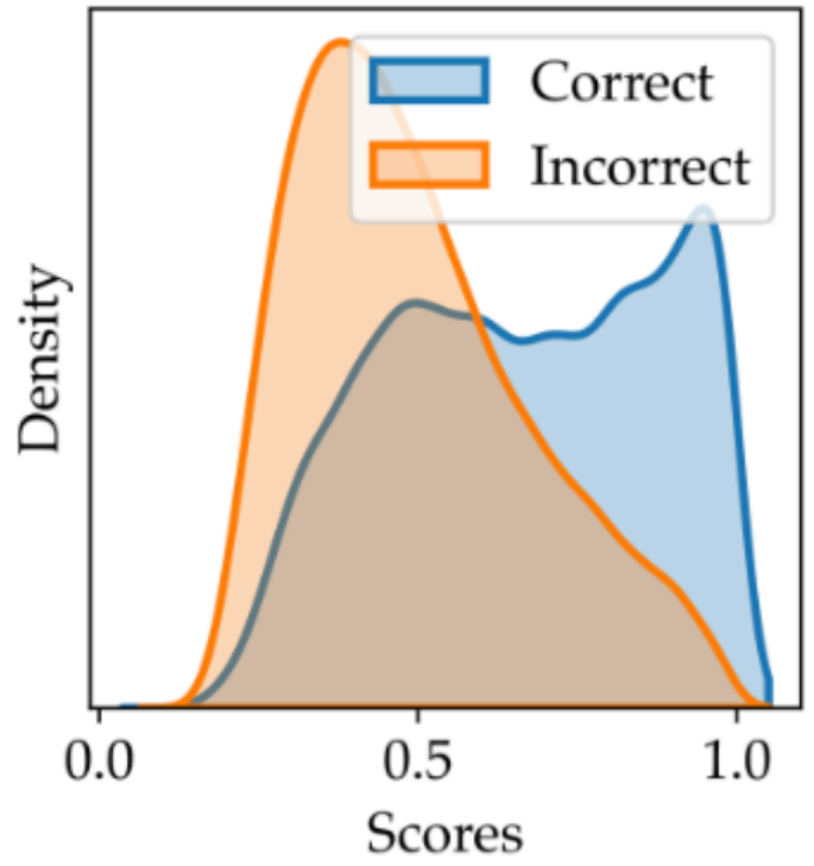
Like Hui [1 2] Mikhail Belkin [2 1] Stephen Wright [3]

Platt 1999; Zadrozny & Elkan, 2001; 2002; Guo et al. 2017; Kumar et al. 2019; Corbiére et al. (2019); Kull et al. 2019, Mukhoti et al. 2020; Gupta & Ramdas 2021; Moon et al. 2020; Zhu et al. 2022; Hui et al. 2023

Experiment

Run 1 round of TBAL + **Temperature Scaling**

Data	CIFAR-10
Model	CNN model (5.8 M parameters)
Training data	4000 points drawn randomly
Validation data	1000 points drawn randomly
Error Tolerance	5%



Kernel Density Estimate(KDE) of scores on the remaining unlabeled data

Test Accuracy	55%
Coverage	4.9%
Auto-labeling Error	14.1%

What are the right choices of confidence functions for TBAL and how can we obtain such functions?

The Optimal Confidence Functions for TBAL

In any round, given the classifier h

We want to find function g that can,

a) Give maximum coverage

b) Ensure auto-labeling error $\leq \epsilon_a$

$\hat{y} := h(\mathbf{x})$

confidence function $g : \mathcal{X} \to \Delta^k$

Depends on h

but drop it for convenience

Address Two Challenges

Do not know the true quantities

Efficient method to solve the optimization

Hypothetically, if we know true distribution and labels,

Coverage $\mathcal{P}(g, \mathbf{t} \mid h) := \mathbb{P}_{\mathbf{x}}\big(g(\mathbf{x})[\hat{y}] \geq \mathbf{t}[\hat{y}]\big),$

Auto-labeling Error $\mathcal{E}(g, \mathbf{t} \mid h) := \mathbb{P}_{\mathbf{x}}\big(y \neq \hat{y} \mid g(\mathbf{x})[\hat{y}] \geq \mathbf{t}[\hat{y}]\big).$

$$\boxed{\underset{g \in \mathcal{G}, \mathbf{t} \in T^k}{\arg\max} \quad \mathcal{P}(g, \mathbf{t} \mid h) \ \text{ s.t. } \ \mathcal{E}(g, \mathbf{t} \mid h) \leq \epsilon_a. \qquad \text{(P1)}}$$

$$\Downarrow$$

$g^\star \quad \mathbf{t}^\star$

Use part of validation data to estimate the quantities

$$\widehat{\mathcal{P}}(g, \mathbf{t} \mid h, D) := \frac{1}{|D|} \sum_{(\mathbf{x}, y) \in D} \boxed{\mathbb{1}\left(g(\mathbf{x})[\hat{y}] \geq \mathbf{t}[\hat{y}]\right),}$$

$$\widehat{\mathcal{E}}(g, \mathbf{t} \mid h, D) := \frac{\sum_{(\mathbf{x}, y) \in D} \boxed{\mathbb{1}\left(y \neq \hat{y} \wedge g(\mathbf{x})[\hat{y}] \geq \mathbf{t}[\hat{y}]\right)}}{\sum_{(\mathbf{x}, y) \in D} \boxed{\mathbb{1}\left(g(\mathbf{x})[\hat{y}] \geq \mathbf{t}[\hat{y}]\right)}}.$$

D_{th}	D_{cal}
For threshold estimation	To learn the confidence function

D_{val}

$$\underset{g \in \mathcal{G}, \mathbf{t} \in T^k}{\arg\max} \quad \widehat{\mathcal{P}}(g, \mathbf{t} \mid h, D_{\text{cal}}) \ \text{ s.t. } \ \widehat{\mathcal{E}}(g, \mathbf{t} \mid h, D_{\text{cal}}) \leq \epsilon_a. \qquad \text{(P2)}$$

Address Two Challenges

~~Do not know the true quantities~~

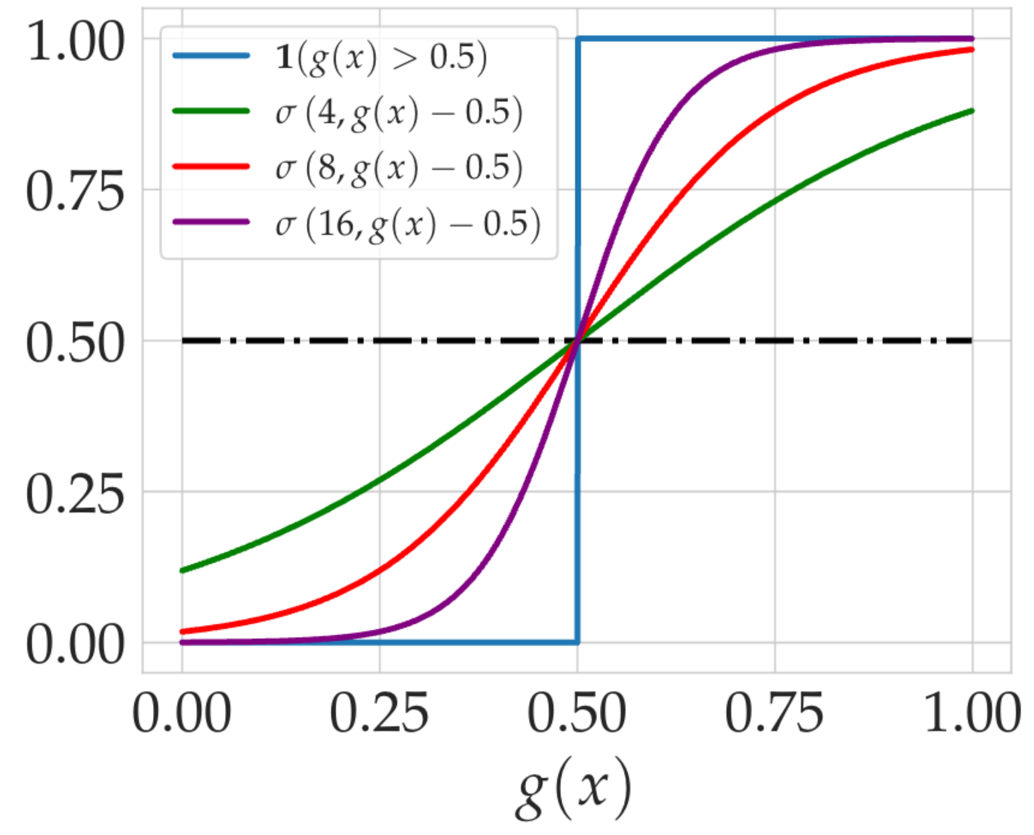
Use part of validation data

Efficient method to solve the optimization

0-1 loss, hard to optimize

Use surrogates for 0-1 variables

$$\sigma(\alpha, z) := 1/(1+\exp(-\alpha z))$$



$$\boxed{\mathbb{1}\left(g(\mathbf{x})[\hat{y}] \geq \mathbf{t}[\hat{y}]\right)} \longrightarrow \boxed{\sigma\left(\alpha, g(\mathbf{x})[\hat{y}] - \mathbf{t}[\hat{y}]\right)}$$

$$\widetilde{\mathcal{P}}(g, \mathbf{t} | h, D_{\text{cal}}) := \frac{1}{|D_{\text{cal}}|} \sum_{(\mathbf{x},y) \in D_{\text{cal}}} \boxed{\sigma\left(\alpha, g(\mathbf{x})[\hat{y}] - \mathbf{t}[\hat{y}]\right),}$$

$$\widetilde{\mathcal{E}}(g, \mathbf{t} \mid h, D_{\text{cal}}) := \frac{\sum_{(\mathbf{x},y)\in D_{\text{cal}}} \mathbb{1}\left(y \neq \hat{y}\right) \boxed{\sigma\left(\alpha, g(\mathbf{x})[\hat{y}] - \mathbf{t}[\hat{y}]\right)}}{\sum_{(\mathbf{x},y)\in D_{\text{cal}}} \boxed{\sigma\left(\alpha, g(\mathbf{x})[\hat{y}] - \mathbf{t}[\hat{y}]\right)}}.$$

$$\underset{g \in \mathcal{G}, \mathbf{t} \in T^k}{\arg\min} \quad -\widetilde{\mathcal{P}}(g, \mathbf{t} \mid h, D_{\text{cal}}) + \lambda \widetilde{\mathcal{E}}(g, \mathbf{t} \mid h, D_{\text{cal}}) \qquad \textbf{(P3)}$$

Address Two Challenges

~~Do not know the true quantities~~

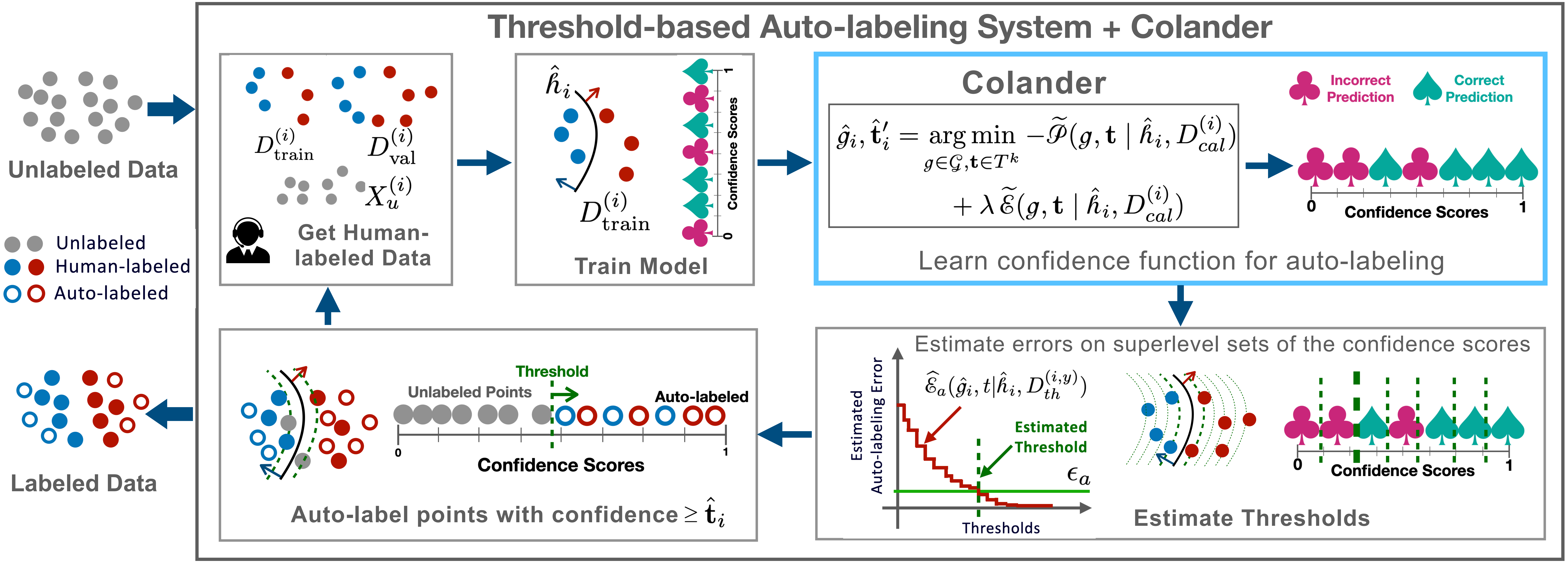
Estimate using part of validation data

~~Efficient method to solve opt.~~

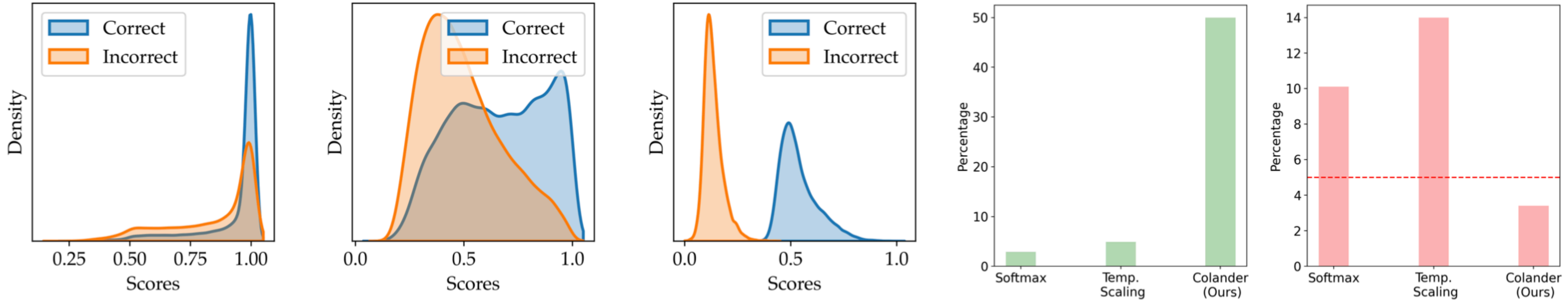
Replace 0-1 variables by sigmoids.

Solve it using gradient-based methods SGD, Adam etc.

Updated workflow of TBAL



It boosts coverage significantly



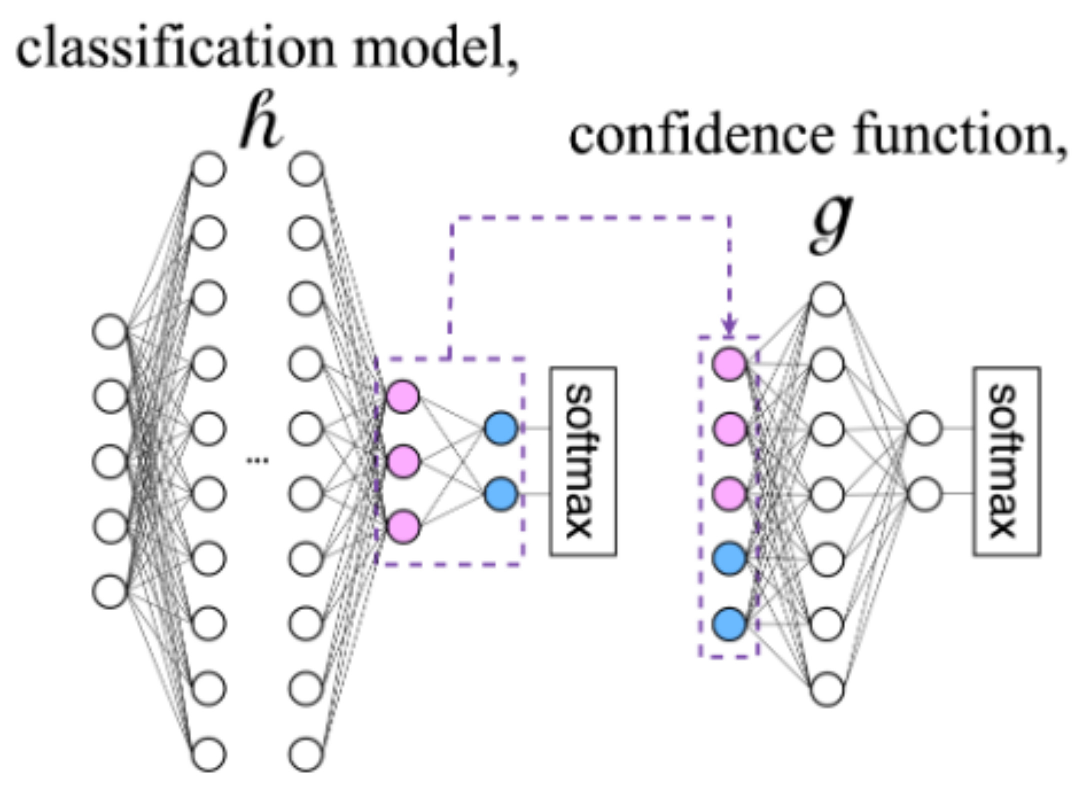
(a) Softmax (b) Temp. Scaling (c) Colander (Ours) (d) Coverage (e) Auto-labeling error

Data	CIFAR-10
Model	CNN model (5.8 M parameters)
Training data	4000 points drawn randomly
Validation data	1000 points drawn randomly
Error Tolerance	5%

Run 1 round of TBAL +
Temperature Scaling or **Colander**

Experiments Setup

Choice of $\mathcal{G}$



Protocol for Experiments

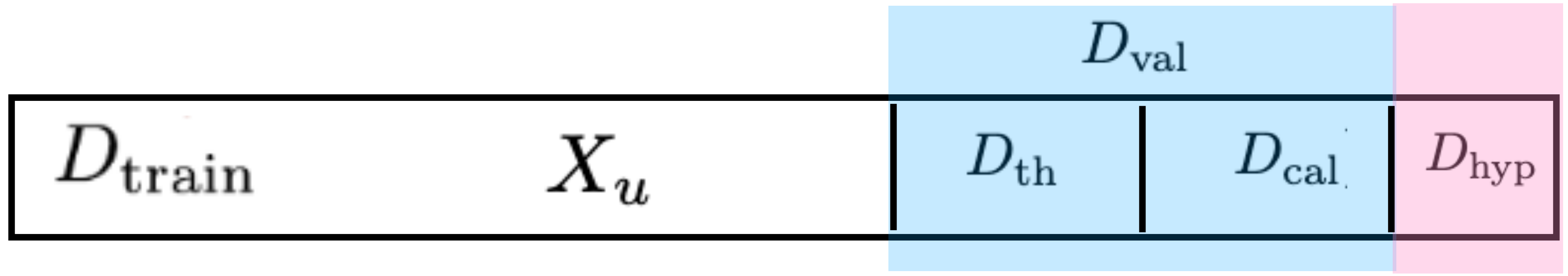
We want to simulate how it would be run in practice.

Hyperparameter Search

For any combination of hyperparameters

run one round of TBAL and evaluate on D_{hyp}

and pick the combination with maximum coverage while having error below $\leq \epsilon_a$

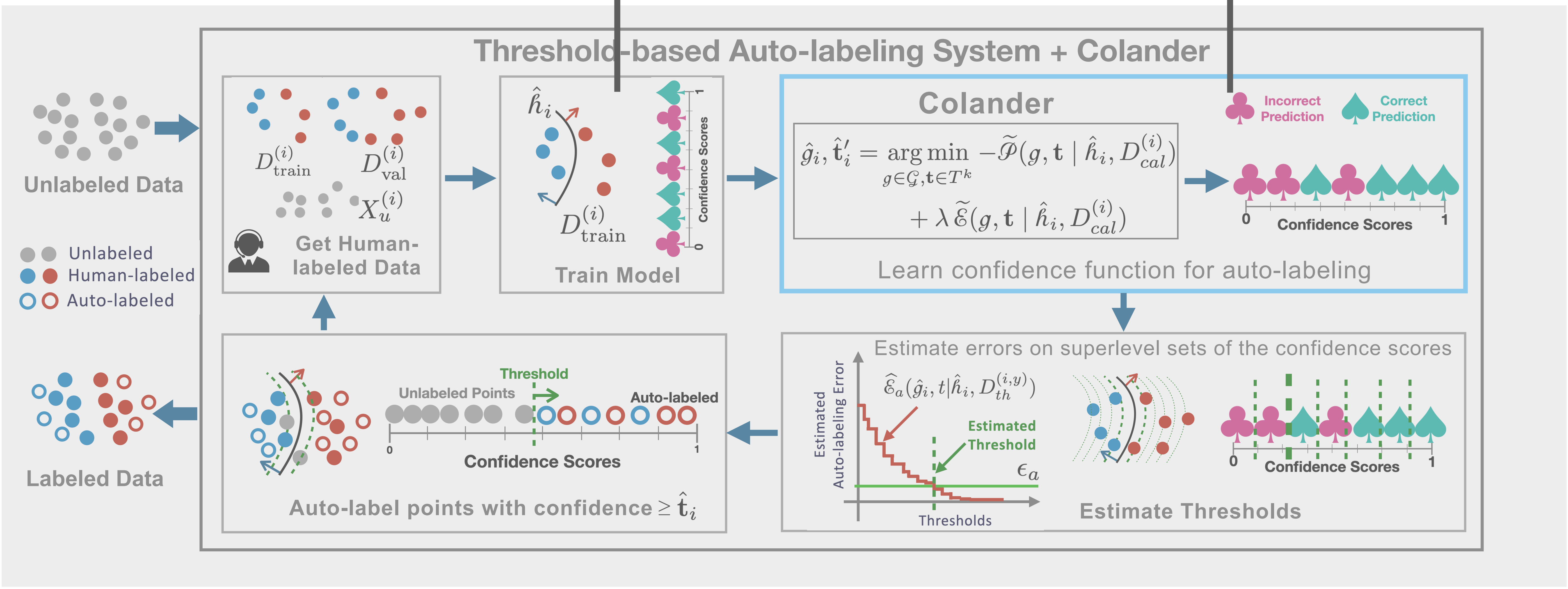


Train-time

1. Vanilla
2. CRL (Moon et al. 2020)
3. FMFP (Zhu et al. 2022)
4. Squentropy (Hui et al. 2023)

Post-hoc

1. Colander (Ours)
2. Temperature Scaling (Guo et al. 2017)
3. Histogram Binning (Gupta & Ramdas, 2021)
4. Scaling Binning (Kumar et al. 2019)
5. Dirichlet (Kull et al. 2019)



Cross product, resulting in 20 methods.

Empirical Results

Dataset	Model $\hbar$	N	N_u	K	N_t	N_v	N_{hyp}	Modality	Preprocess	Dimension
MNIST	LeNet-5	70k	60k	10	500	500	500	Image	None	$1 \times 28 \times 28$
CIFAR-10	CNN	50k	40k	10	10k	8k	2k	Image	None	$3 \times 32 \times 32$
Tiny-Imagenet	MLP	110k	90k	200	10k	8k	2k	Image	CLIP	512
20 Newsgroup	MLP	11.3k	9k	20	2k	1.6k	600	Text	FlagEmb.	1,024

Train-time	Post-hoc	MNIST		CIFAR-10		20 Newsgroups		Tiny-ImageNet	
		Err (↓)	Cov (↑)	Err (↓)	Cov (↑)	Err (↓)	Cov (↑)	Err (↓)	Cov (↑)
Vanilla	Softmax	**4.1±0.7**	85.0±2.5	4.8±0.2	14.0±2.1	6.0±0.6	48.2±1.6	11.1±0.3	32.6±0.5
	TS	7.8±0.6	94.2±0.5	7.3±0.3	23.2±0.7	9.7±0.6	60.7±2.3	16.3±0.5	37.4±1.5
	Dirichlet	7.9±0.7	93.2±2.2	7.7±0.5	22.4±1.2	9.4±0.9	59.4±1.8	17.1±0.4	33.3±2.0
	SB	6.7±0.5	92.6±1.5	6.1±0.4	18.6±1.1	8.1±0.6	58.1±1.8	15.7±0.6	35.4±1.2
	Top-HB	7.4±1.4	93.1±3.6	6.0±0.7	15.6±1.9	9.2±1.0	59.0±2.0	16.6±0.5	37.6±2.2
	Ours	4.2±1.5	**95.6±1.4**	**3.0±0.2**	**78.5±0.2**	**2.5±1.1**	**80.6±0.7**	**1.4±2.1**	**59.2±0.8**
CRL	Softmax	4.7±0.4	86.0±4.5	5.2±0.3	15.9±0.8	5.8±0.5	48.3±0.3	10.4±0.4	32.5±0.6
	TS	8.0±0.8	94.8±0.8	6.8±0.8	20.3±1.1	9.5±1.0	61.7±1.6	15.8±0.6	37.4±1.7
	Dirichlet	8.6±0.6	93.1±1.6	7.7±0.2	20.9±1.1	8.7±0.9	58.0±1.4	16.3±0.4	33.1±1.9
	SB	7.4±0.8	93.1±2.7	5.9±0.9	17.9±1.5	8.9±1.1	57.9±3.9	15.0±0.4	35.5±1.2
	Top-HB	7.7±0.8	94.1±1.5	4.4±0.5	12.3±0.4	8.8±1.0	58.8±2.7	16.5±0.5	38.9±1.6
	Ours	**4.5±1.4**	**95.6±1.3**	**2.2±0.6**	**77.9±0.2**	**1.8±1.2**	**81.3±0.5**	**2.8±2.1**	**61.2±1.4**
FMFP	Softmax	4.8±0.8	84.2±4.1	4.9±0.4	15.6±1.7	5.4±0.7	45.4±1.9	10.5±0.3	32.4±1.4
	TS	8.0±0.6	95.3±1.6	6.5±0.3	21.0±1.5	9.5±0.5	57.7±2.2	16.2±1.1	37.7±1.8
	Dirichlet	8.2±1.3	94.0±2.2	6.9±0.4	21.7±1.2	8.9±1.0	56.6±2.4	17.4±0.8	33.0±1.8
	SB	7.2±1.1	93.1±2.3	6.1±0.5	19.5±1.0	8.6±0.4	55.8±1.3	15.5±0.6	36.1±0.5
	Top-HB	7.1±0.6	93.3±4.9	5.2±0.5	14.2±2.4	9.0±0.7	57.9±2.4	16.2±0.4	37.4±1.1
	Ours	**4.6±0.8**	**95.7±0.2**	**3.0±0.4**	**77.4±0.2**	**2.5±0.9**	**80.8±0.6**	**1.8±2.0**	**60.8±1.4**
Squentropy	Softmax	**3.7±1.0**	88.2±3.9	5.2±0.5	21.2±1.8	4.6±0.4	52.0±1.2	7.8±0.3	36.2±0.8
	TS	6.2±1.1	95.6±0.9	6.9±0.6	28.2±2.5	8.3±0.6	66.6±1.4	13.3±0.1	44.9±1.0
	Dirichlet	6.5±1.2	95.9±0.8	7.3±0.3	29.4±1.1	7.8±0.6	64.0±1.3	14.1±0.3	42.5±0.7
	SB	6.0±0.8	95.3±1.2	6.2±0.4	23.8±1.9	7.8±0.7	63.0±2.9	13.0±0.5	45.2±2.0
	Top-HB	5.3±0.4	96.4±0.9	4.3±0.5	15.8±1.4	8.2±0.8	66.5±2.2	13.7±0.1	45.9±1.4
	Ours	4.1±0.8	**97.2±0.5**	**2.3±0.5**	**79.0±0.3**	**3.3±0.8**	**82.9±0.4**	**0.6±0.2**	**66.5±0.7**

Results

Colander works as expected,
achieves high coverage while maintaining error guarantee.

Colander improves upon all training methods

Squentropy does better than other training methods

Other post-hoc methods
increase the coverage but also leading to higher error

The literature has focused on calibrating highly accurate models. May need rethinking when calibrating bad models.

Summary

Confidence functions play a crucial role in TBAL.

Commonly used choices such as **softmax scores** can lead to poor auto-labeling performance.

Applying ad-hoc solutions (e.g. **calibration**) may not help much.

We proposed **Colander** a principled method to learn the **optimal confidence functions for TBAL** and show that it boosts the performance significantly.

Thank You

:-)

Questions and Feedback

\end{talk}