NeSy Learning
Day 2: Learnability

Efthymia Tsamoura
Huawei Labs

Emile van Krieken
VU, Amsterdam

ESSLLI 2025

About this Course

- ✓Day 1: Introduction to NeSy
- ✓Day 2: Learnability
- ✓Day 3: Learning Imbalances in NeSy
- ✓Day 4: Reasoning Shortcuts
- ✓Day 5: Probabilistic Reasoning

Outline of Today's Lecture

- ✓**Introduction to ML**
  - ✓Learning Paradigms
- ✓**Introduction to NeSY**
- ✓**Learnability**
  - ✓Definition
  - ✓Learnability vs Training Deep Networks
  - ✓Known and Deterministic σ
    - ✓M-Unambiguity
  - ✓Unknown and Deterministic σ
    - ✓$\mathcal{G}$-Unambiguity
- ✓**Relationship with Other Weakly-Supervised Learning Settings**
  - ✓Partial Label Learning
  - ✓Learning via Transition Matrices

Key Takeaways

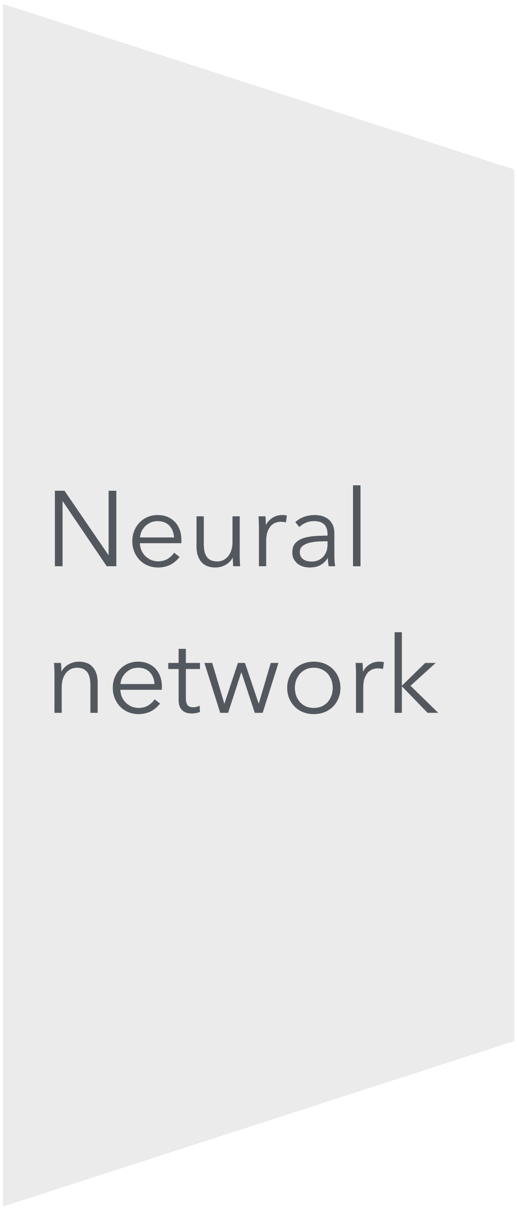
✓We can learn neural classifiers under formal guarantees

✓Problems in NeSY cannot be trivially reduced to problems in standard ML

Quick Recap of Day 1

Example

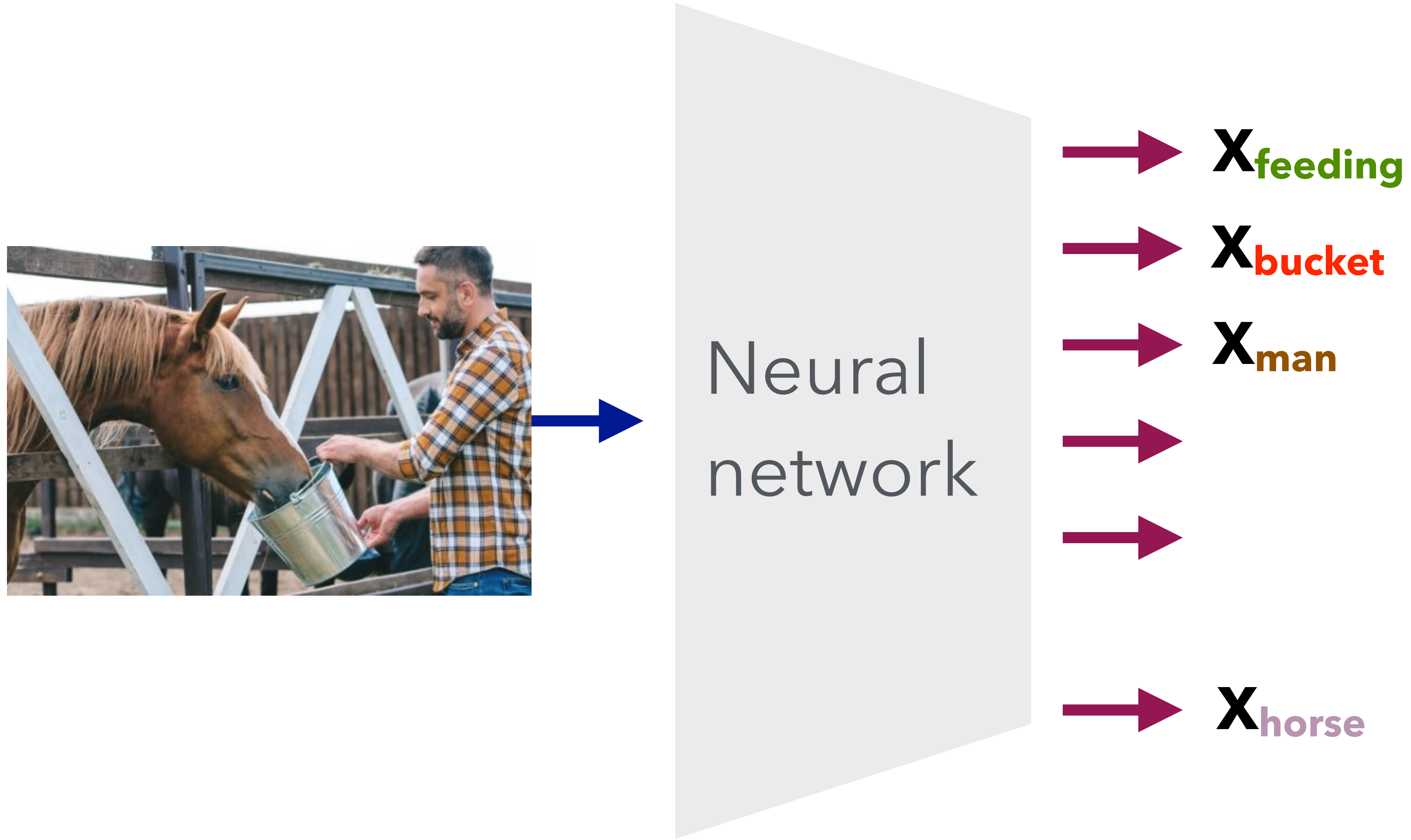
Typical when there are multiple annotators or the classes are difficult to differentiate



Goal: Train the DNN knowing that the input shows a **raven or crow or blackbird**

Reduces to: Train the DNN subject to the Boolean constraint $X_{raven} \vee X_{crow} \vee X_{blackbird}$

Logic in Computer Vision



Integrity constraints

¬feeding(bucket, man)

¬feeding(man, bucket)

Boolean constraints

$$\phi_1 = \neg(X_{feeding} \wedge X_{bucket} \wedge X_{man})$$

$$\phi_2 = \neg(X_{feeding} \wedge X_{man} \wedge X_{bucket})$$

Probability computation

$$P(\phi_1 \wedge \phi_2)$$

Loss computation

$$-\log P(\phi_1 \wedge \phi_2)$$

Introduction to Machine Leaning & Neural Networks

Neural Networks

Neural networks are **functions of (unknown) parameters** from an input space $\mathscr{X}$ to an output one $\mathscr{Y}$

✓ Examples of **inputs:** images, video, text

We will use θ to represent the parameters of those networks

Neural Networks

Depending on their outputs, we have two types of networks:

✓ **Classifiers:** They output the **class** the input belongs to

✓ **Generative models:** They output new content, e.g., images, text, videos

Neural Networks

In today's and tomorrow's talk, we will focus on **classifiers.**

Neural Networks

How to learn the parameters θ of a network?

✓ We **present examples.** Three options:

  ✓ Present the **gold** output of each input (supervised learning)

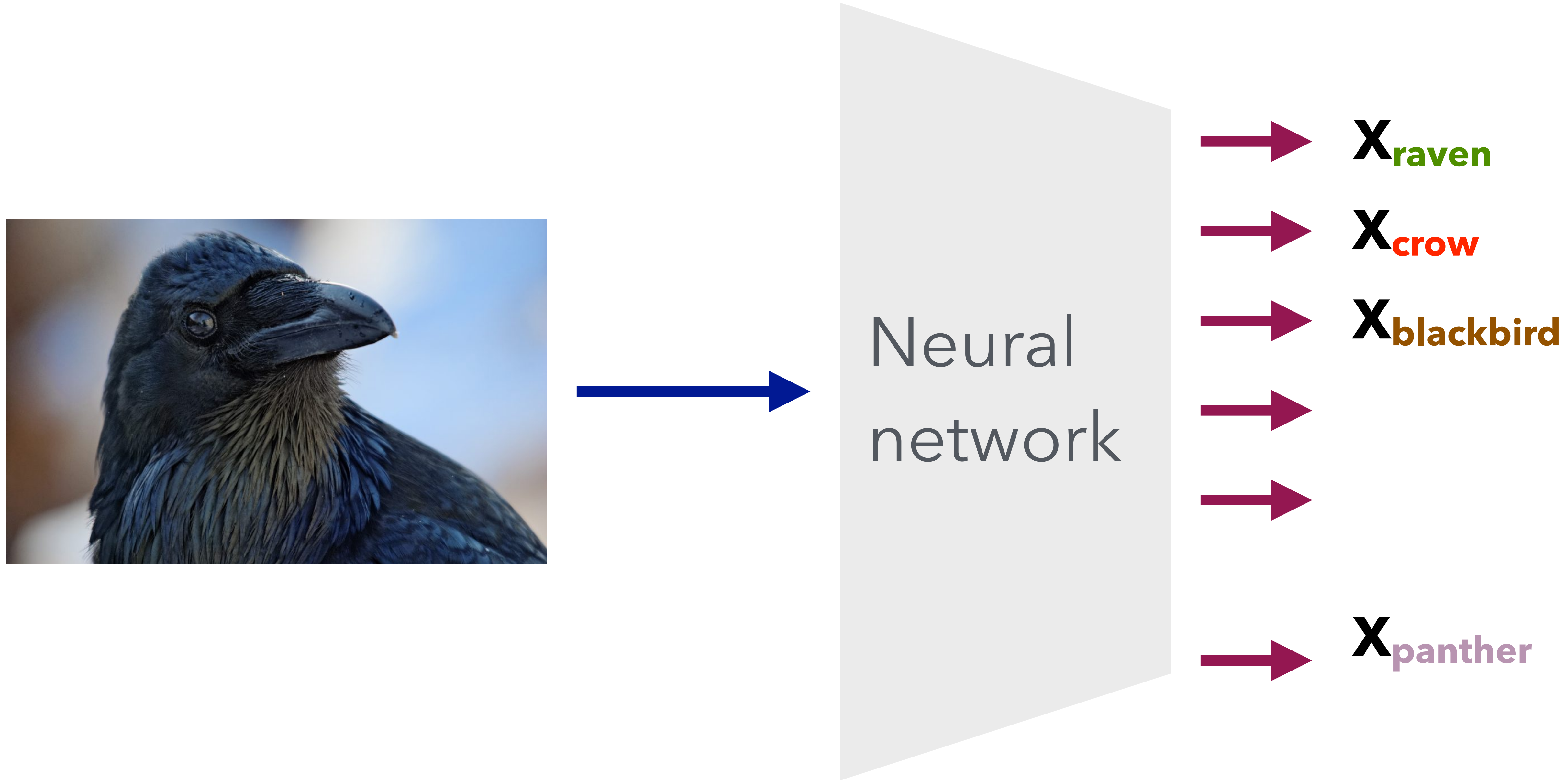
  ✓ Do not present the **gold** output of each input (unsupervised learning)

  ✓ Present a constraint the **gold** output of each input should adhere to (constrained learning)

Learning depends on a differentiable function, called a **loss.** The aim is to modify the parameters θ so that the outputs of the network minimize the **loss function on the examples.**

Supervised Learning

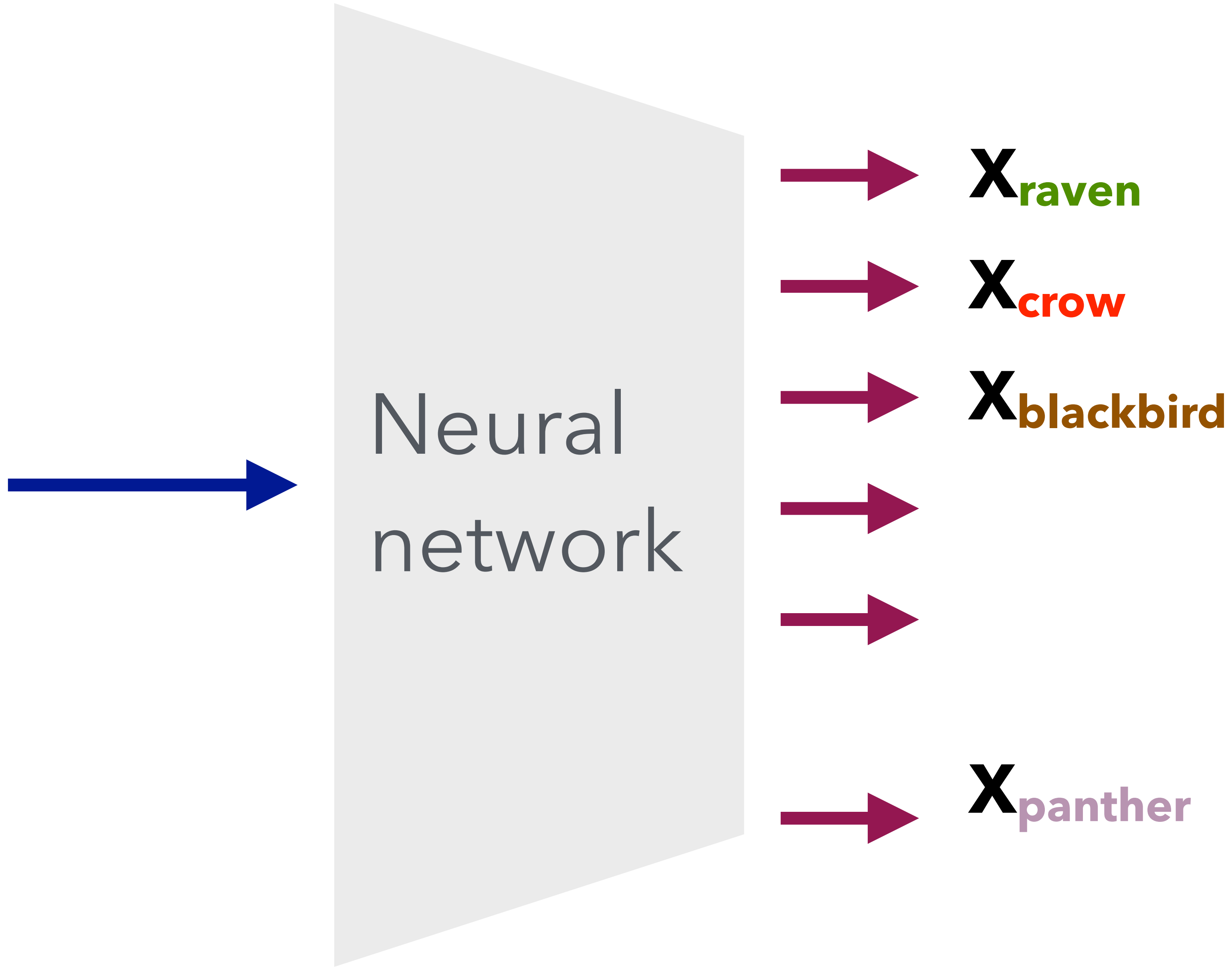
The loss measures how "far" are the current predictions from the target class



Example: An image and its target class, e.g., **raven**

Unsupervised Learning

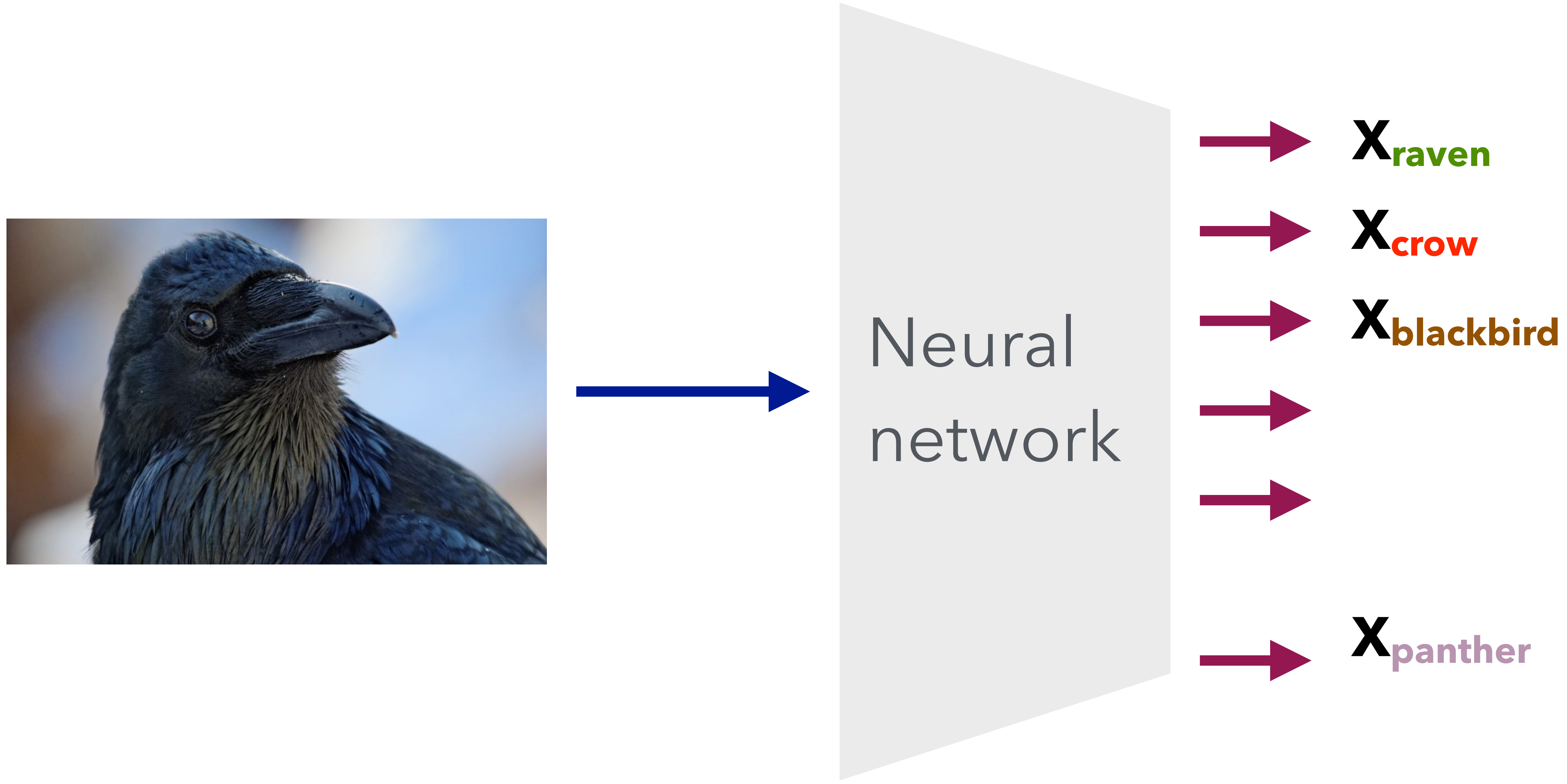
The loss measures how “far” are different inputs from each other



Example: An image

Constrained Learning

The loss measures how "far" are current predictions from satisfying the constraint

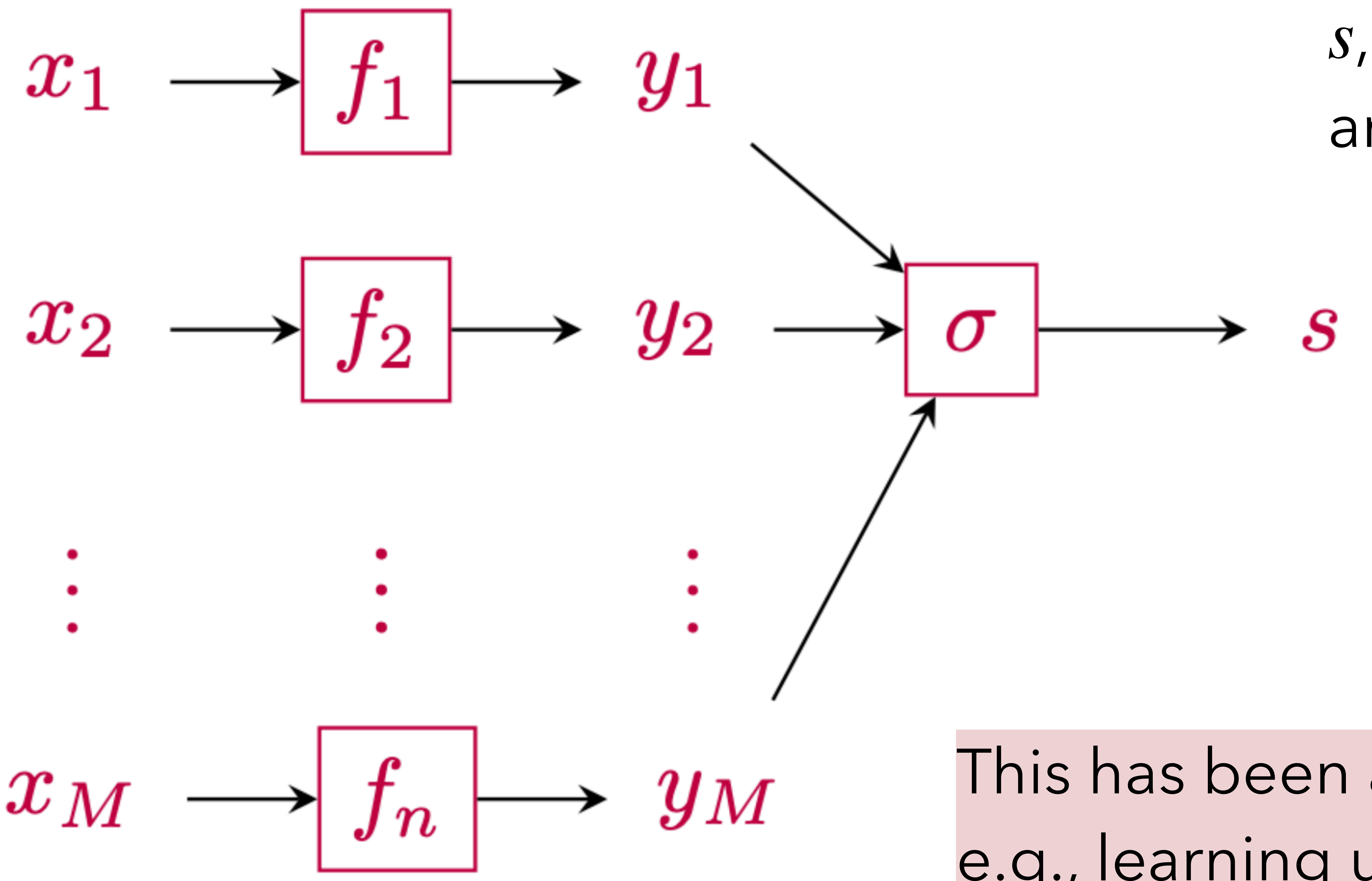


Our NeSy learning setting falls under this category

Example: An image and the constraint the image adheres to, e.g., the image shows a **raven** **or** **crow** **or** **blackbird**

NeSy Learning
-Introduction

Learning Setting



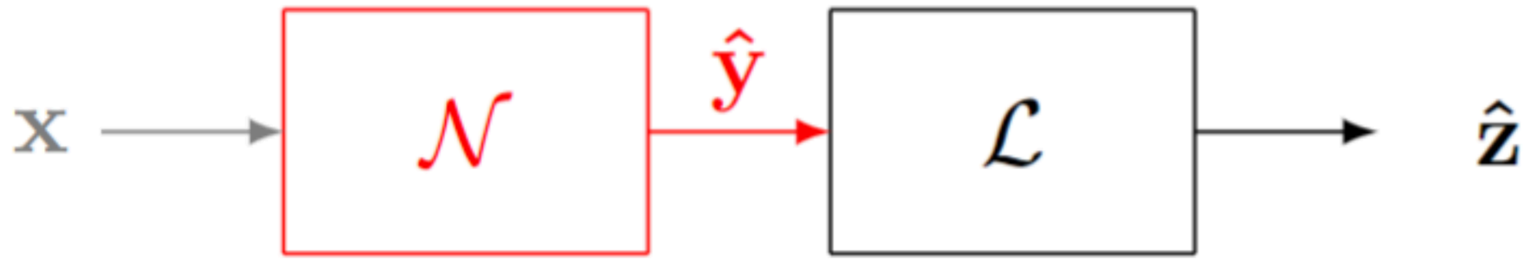
Problem formulation: Given the x_i's and s, **learn** the f_i's. The gold labels $y_1, \ldots, y_M$ are **unknown**.

This has been an open problem in other relevant fields, e.g., learning under indirect supervision.

Kaifu Wang, Efthymia Tsamoura, and Dan Roth. On Learning Latent Models with Multi-Instance Weak Supervision. In NeurIPS, 2023.

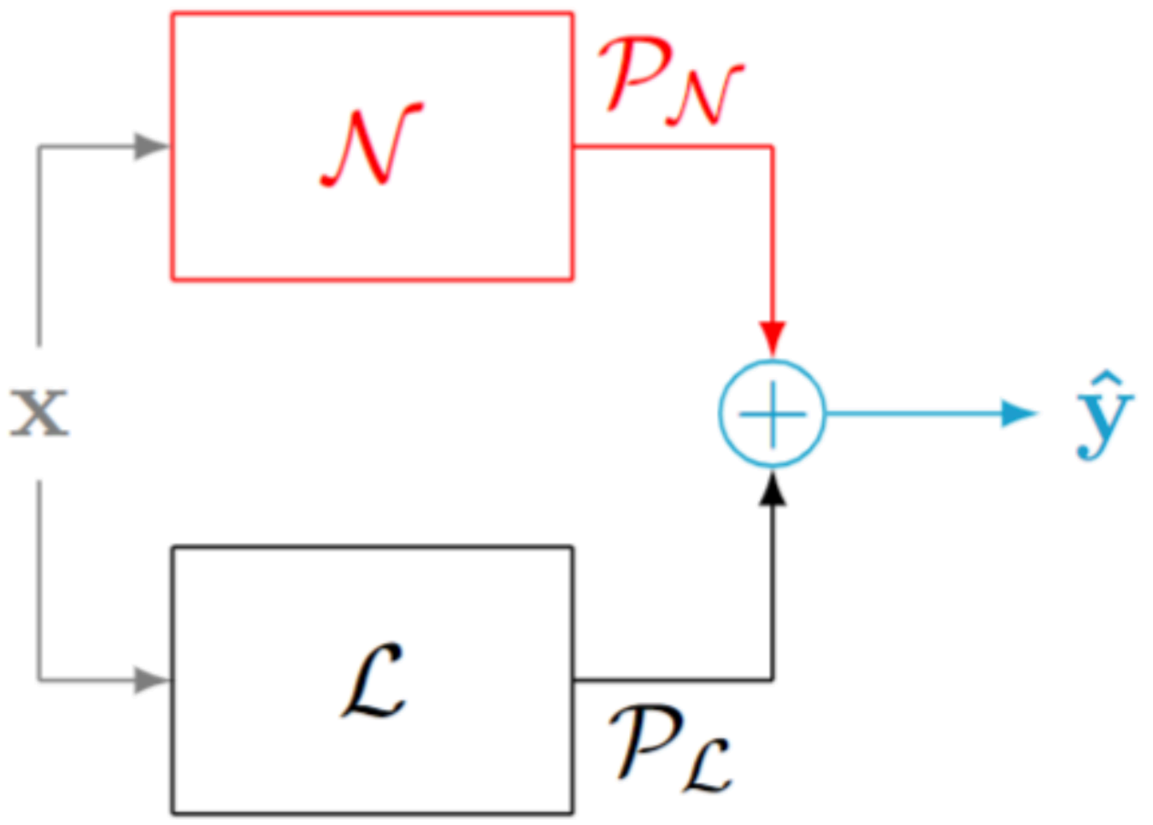
Types of Integration

Stratified



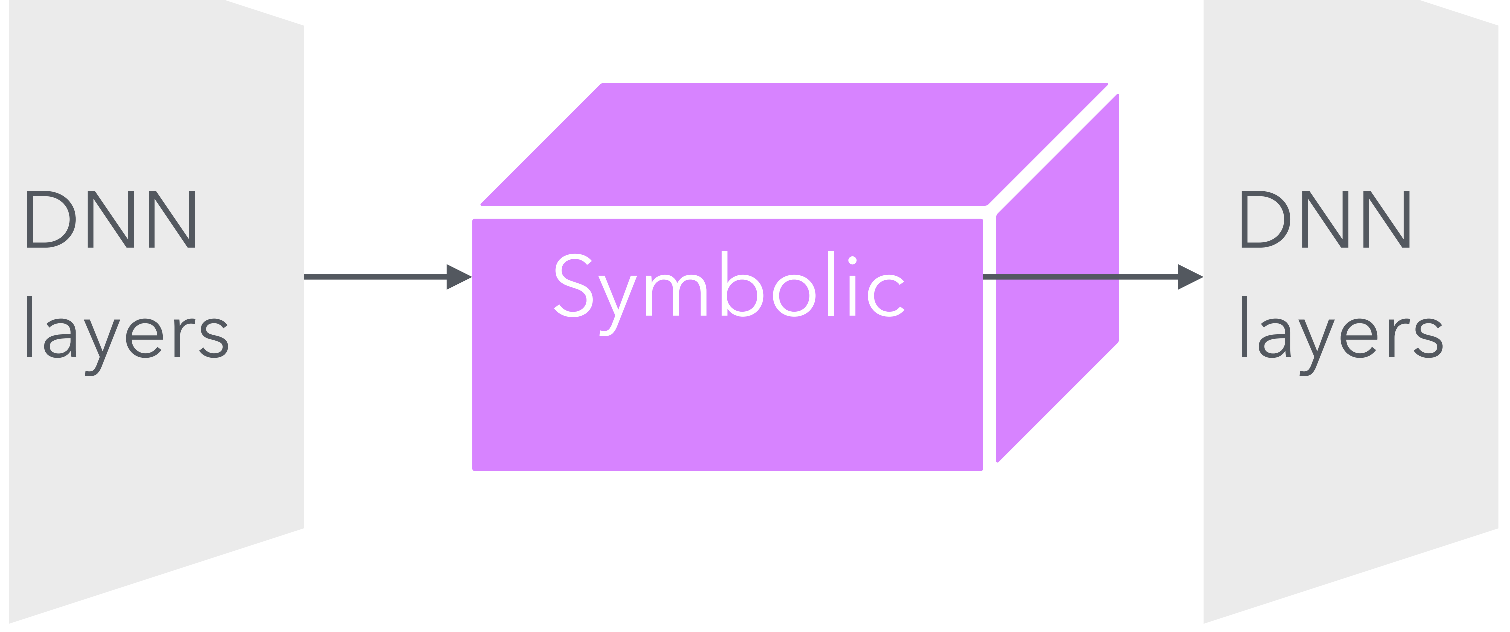
- ✓DeepProbLog [NeurIPS 2018]
- ✓ABL [NeurIPS 2019]
- ✓NeurASP [IJCAI 2020]
- ✓NeuroLog [AAAI 2021]
- ✓Scallop [NeurIPS 2021]
- ✓ENT [ICLR 2023]
- ✓DeepSoftLog [NeurIPS 2023]
- ✓ISED [NeurIPS 2023]
- ✓Dolphin [arXiv 2024]

Teacher-Student



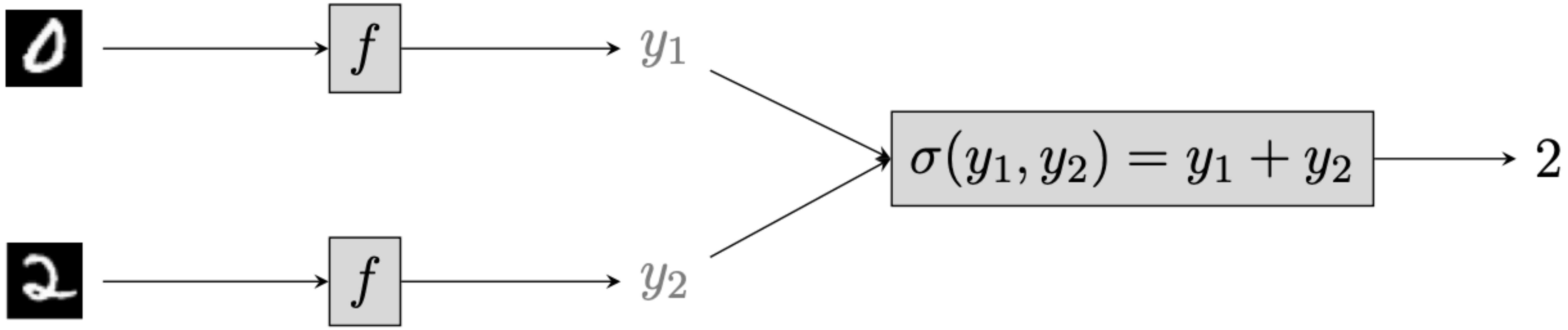
- ✓T-S, ACL [EMNLP 2016]
- ✓DPL [EMNLP 2018]
- ✓Concordia [ICML 2023]

As a layer



- ✓MIPaaL [AAAI 2020]
- ✓BB-backprop [ICLR 2020]
- ✓CombOptNet [ICML 2021]
- ✓SurCo [ICML 2023]
- ✓GenCO [ICML 2024]

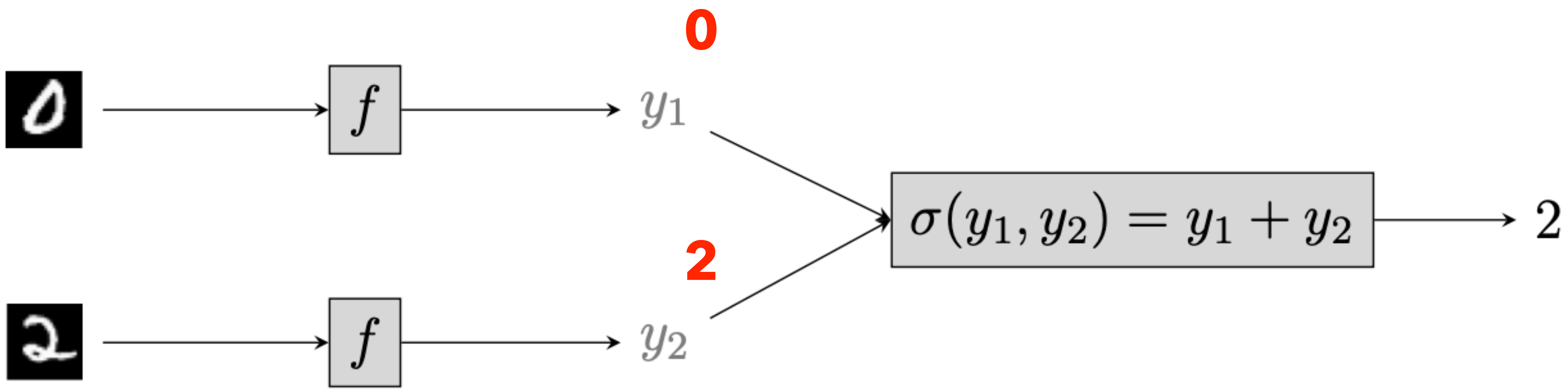
Example



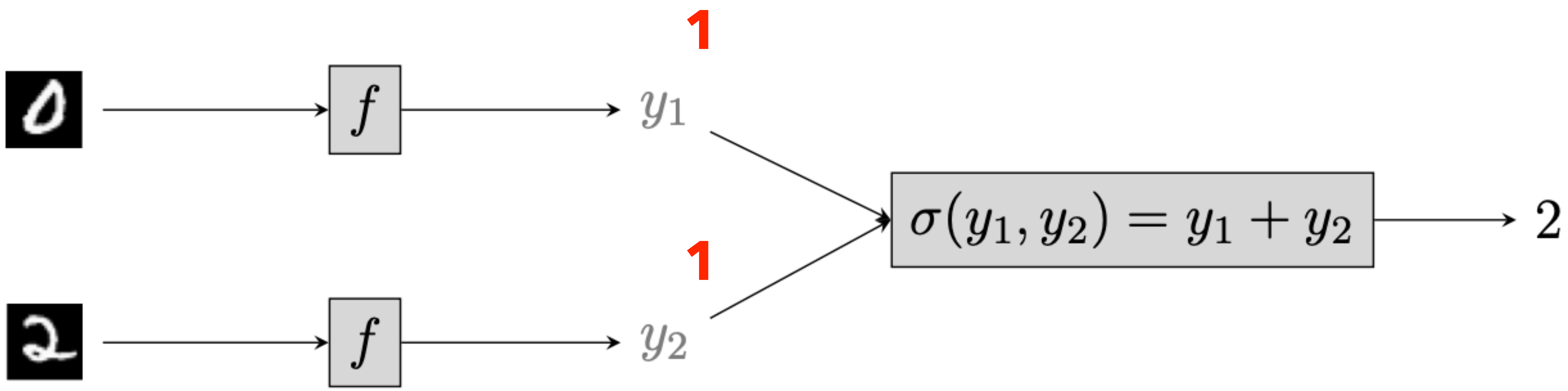
Challenges

✓ σ may be non-invertible

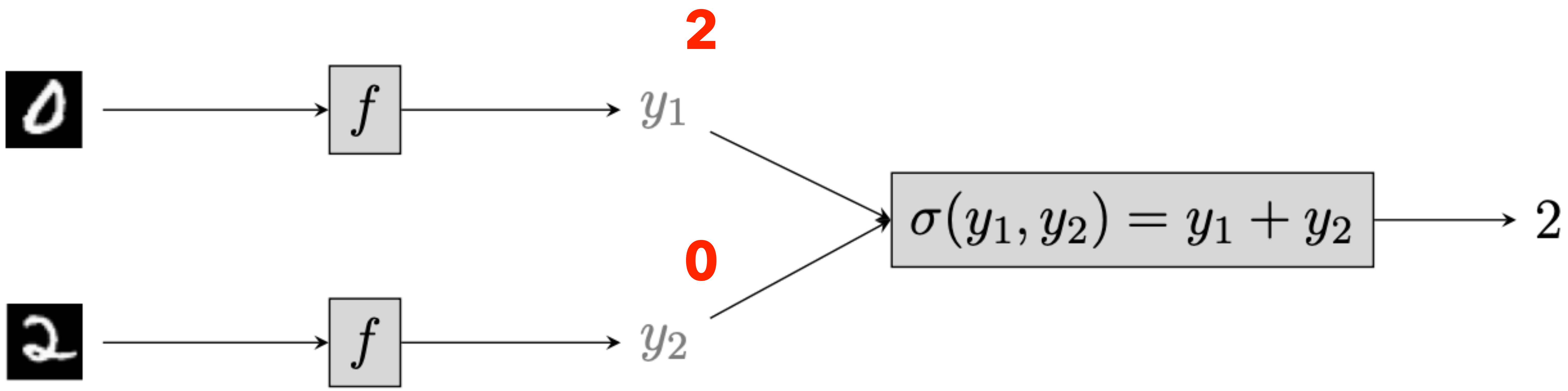
Challenges



Challenges



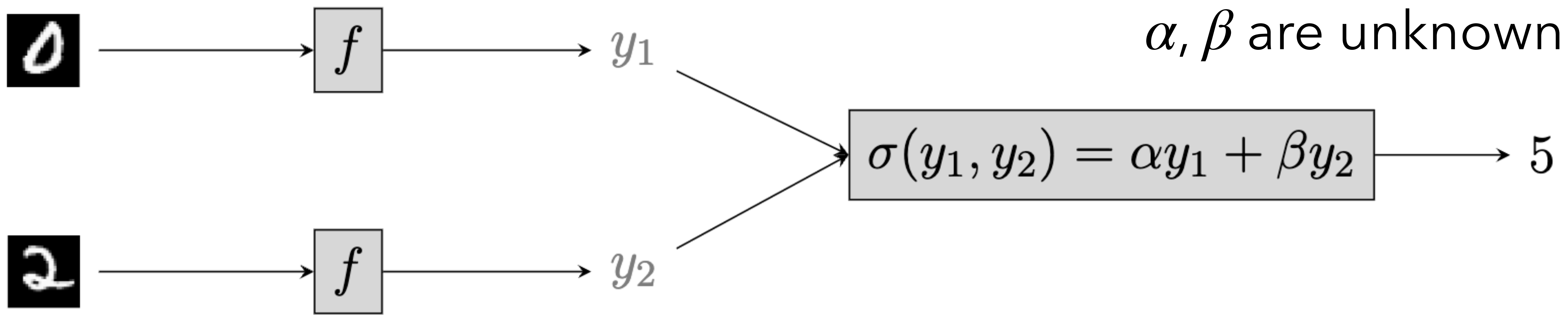
Challenges



Challenges

- ✓ σ may be non-invertible
- ✓ σ may be unknown

Challenges



α, β are unknown

Question: Is this learning setting any interesting?

(Some) Relevant Neurosymbolic Frameworks

- ✓DeepProbLog [NeurIPS 2018]
- ✓ABL [NeurIPS 2019]
- ✓NeurASP [IJCAI 2020]
- ✓NeuroLog [AAAI 2021]
- ✓Scallop [NeurIPS 2021]
- ✓ENT [ICLR 2023]
- ✓DeepSoftLog [NeurIPS 2023]
- ✓ISED [NeurIPS 2023]
- ✓Dolphin [arXiv 2024]

The NLP community was using such architectures before the NeSy community

Some Applications

✓Fine-tuning LLMs (Li et al, 2023)

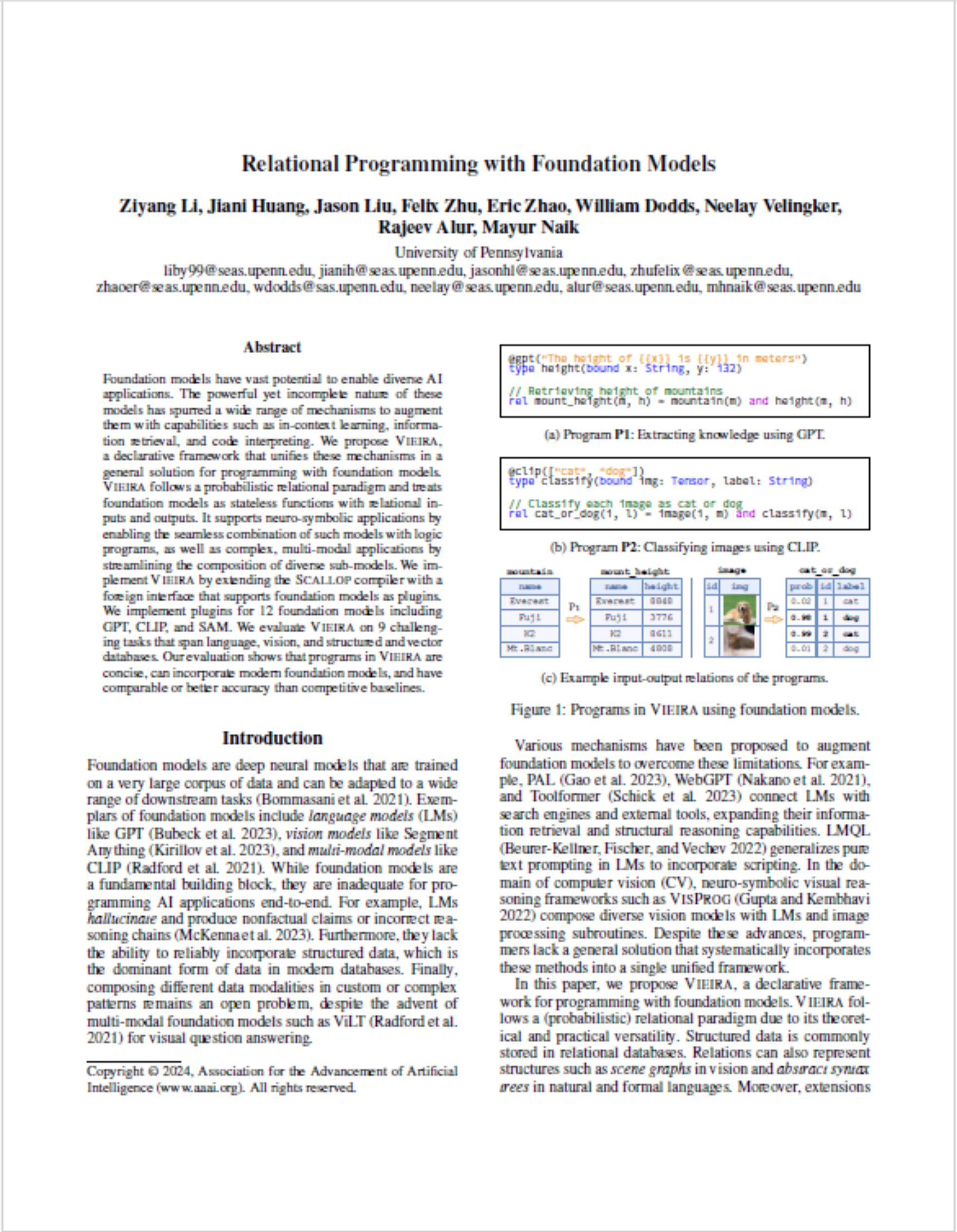
✓Visual question answering (Li et al, 2023, Tsamoura et al, 2023)

✓Spatio-temporal scene graph generation using VLMs (Huang et al., 2025)

✓Learning knowledge graph embeddings (Maene & Tsamoura, 2025)


Improved Logical Reasoning of Language Models via Differentiable Symbolic Programming

Hanlin Zhang, Jiani Huang, Ziyang Li, Mayur Naik, Eric Xing


Relational Programming with Foundation Models

Ziyang Li, Jiani Huang, Jason Liu, Felix Zhu, Eric Zhao, William Dodds, Neelay Velingker, Rajeev Alur, Mayur Naik


Published as a conference paper at ICLR 2025

LASER: A Neuro-Symbolic Framework for Learning Spatio-Temporal Scene Graphs with Weak Supervision

Jiani Huang, Ziyang Li, Mayur Naik, Ser-Nam Lim

On the Power of σ

Our formulation is general enough to represent **different languages,** e.g.,

✓ non-linear functions

✓ systems of Boolean equations

✓ Datalog programs

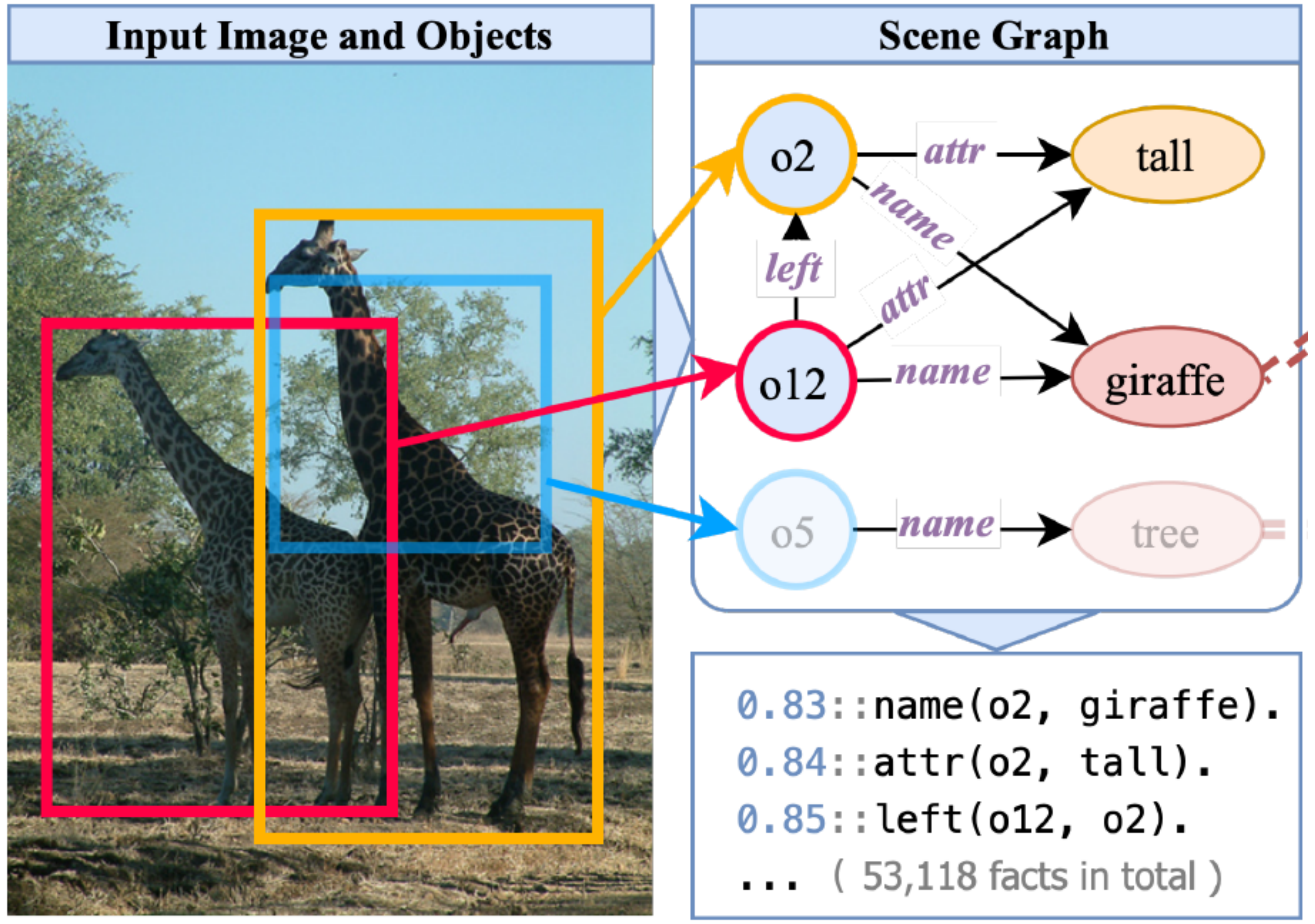
Our formulation can express logical theories via backward reasoning, *aka* **abduction**.

Efthymia Tsamoura and Loizos Michael. Neural-Symbolic Integration: A Compositional Perspective In AAAI, 2023.

Benefits of NeSy Learning

- ✓Re-usability
- ✓Higher accuracy
- ✓Reduced model size

Example: VQA



Task: Answer questions over an image

$$\mathsf{Q}(O) \leftarrow \mathsf{NAME}(herbivore, O)$$
$$\mathsf{NAME}(N, O) \land \mathsf{NAME}(N', O) \rightarrow \mathsf{ISA}(N', N)$$
$$\rightarrow \mathsf{ISA}(giraffe, herbivore)$$
$$\rightarrow \mathsf{ISA}(dear, herbivore)$$

Table. R@5 for answering visual queries over VQAR (NeurIPS 2021). C5 & C6 denote the number of reasoning steps to answer the query.

Testset	LXMERT (EMNLP 2019)	RVC (PAMI 2023)	TG-Guided VQA
C5	64.05%	74.62%	**87.01%**
C6	56.51%	72.04%	**85.45%**
	836 MB	**100 MB**	**42 MB**

Efthymia Tsamoura, Jaehun Lee, and Jacopo Urbani. Probabilistic Reasoning as Scale: Trigger Graphs to the Rescue. In SIGMOD, 2023.

NeSy Learning
-Learnability

Learnability

Objective: Develop necessary (and sufficient) conditions that must be satisfied to ensure classifier PAC learnability

Cases:

✓Known and Deterministic σ

  ✓M-Unambiguity

✓Unknown and Deterministic σ

  ✓$\mathcal{G}$-Unambiguity

Learnability vs Training Neural Networks

Disclaimer: We will not discuss techniques to train deep networks subject to symbolic components – to be fair, we will present one such technique at the very end

Practical Options:

✓**Losses based on probabilistic logic semantics**

  ✓We discussed this yesterday – More will be covered in the next days

✓**Losses based on fuzzy logic semantics**

  ✓We discussed this yesterday

✓**Learning based on integer linear programming**

  ✓We will discuss this tomorrow

✓**Learning based on expectation maximization**

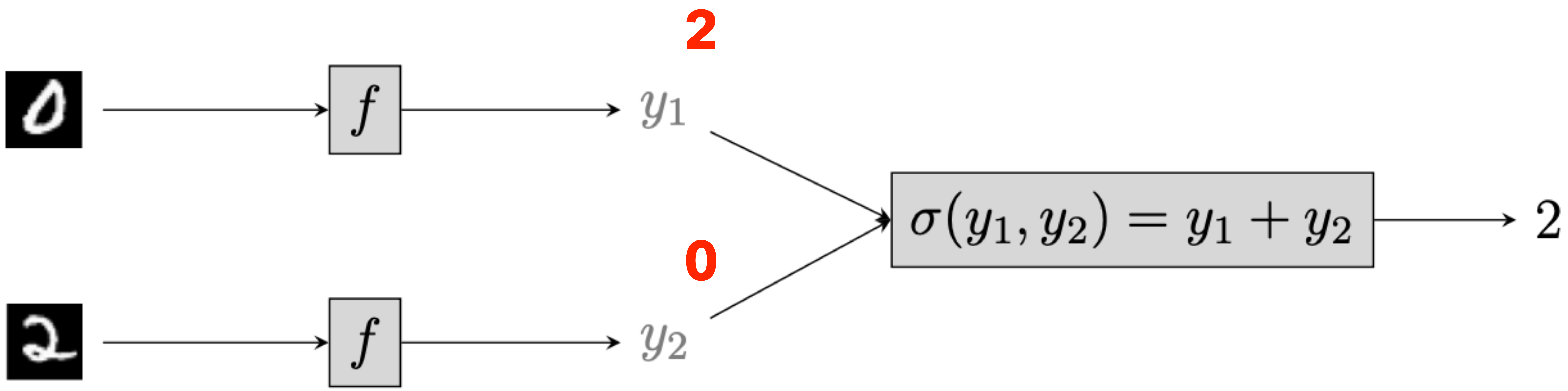
✓**Learning via differentiation through argmax**

PAC-Learnability: Known and deterministic σ

Notation

Supervised learning	NeSy	Meaning
x (given)	$\boldsymbol{x} = x_1, \ldots, x_M$ (given)	input(s)
y (given)	$\boldsymbol{y} = y_1, \ldots, y_M$ (unknown)	gold label(s)
-	$s = \sigma(\boldsymbol{y})$ (given)	partial label
-	σ (given)	transition function
$\mathcal{D}$	$\mathcal{D}_{\mathsf{P}}$	training distribution (drawing M independent samples from $\mathcal{D}$)
$[f](x)$	$[f](x)$	prediction
$\ell^{01}(y, y') := 1\{y \neq y'\}$	$\ell^{01}_{\sigma}(\boldsymbol{y}, s) := 1\{\sigma(\boldsymbol{y}) \neq s\}$	zero-one (partial) loss
$\mathcal{R}^{01}(f) :=$ $E_{(X,Y)\sim\mathcal{D}}[\ell^{01}([f](X), Y)]$	$\mathcal{R}^{01}_{\mathsf{P}}(f;\sigma) :=$ $E_{(\mathbf{X},S)\sim\mathcal{D}_{\mathsf{P}}}[\ell^{01}_{\sigma}([f](\mathbf{X}), S)]$	zero-one (partial) risk

Example



✓ $\sigma(y_1, y_2) = y_1 + y_2$

✓ $\ell_\sigma^{01}(y_1 = \mathbf{0}, y_2 = \mathbf{2}, s = \mathbf{2}) = \mathbf{0}$

✓ $\ell_\sigma^{01}(y_1 = \mathbf{2}, y_2 = \mathbf{0}, s = \mathbf{2}) = \mathbf{0}$

✓ $\ell_\sigma^{01}(y_1 = \mathbf{2}, y_2 = \mathbf{1}, s = \mathbf{2}) = \mathbf{1}$

✓ $R^{01}(f)$ denotes the probability the MNIST classifier makes a wrong prediction

✓ $R_{\mathsf{P}}^{01}(f; \sigma)$ denotes the probability the overall output is wrong

PAC-Learnability

A problem instance is PAC-learnable if there exists an algorithm $\mathscr{A}$ such that for any two user parameters ϵ and δ the following holds under any input distribution:

✓with probability at least 1-δ

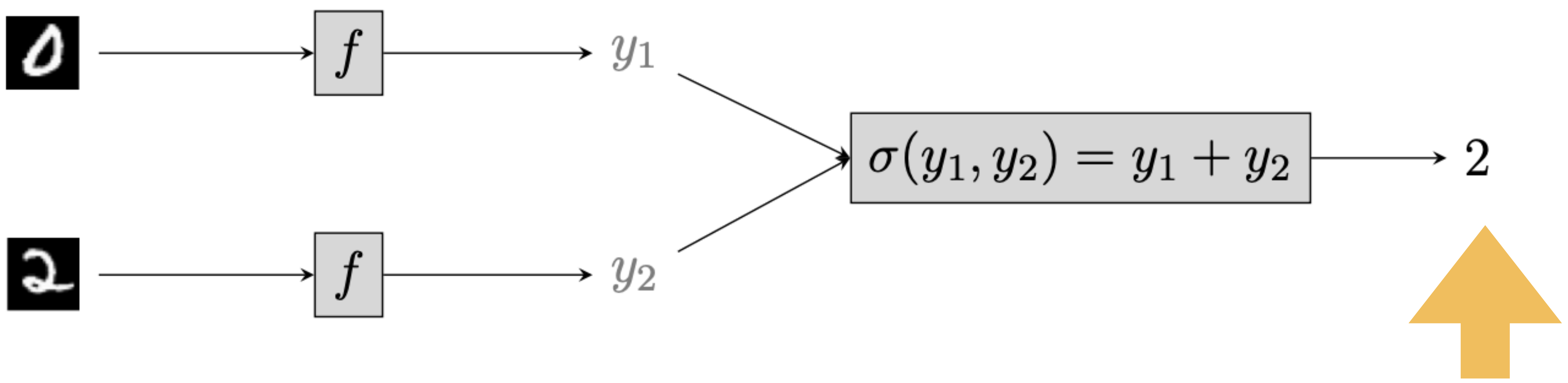
✓the learned classifier f misclassifies an input with probability $\leq \epsilon$

✓when given at least $m_{\epsilon,\delta}$ samples

Polynomial in ϵ and δ

Empirical Risk Minimizer (ERM) Learning

Algorithm $\mathscr{A}$ will work by minimizing the empirical risk $R_{\mathrm{P}}^{01}(f;\sigma)$ given a set of partial samples



$\mathscr{A}$ will minimize the error at the overall output

Question: Do you find this a reasonable decision?

Proving PAC-Learnability: Known and Deterministic σ

To prove learnability of such a NeSY problem instance, we must **bound** $R^{01}(f)$ (**zero-one risk**) with $R_{\mathsf{P}}^{01}(f;\sigma)$ (**zero-one partial risk**), under ***any*** training distribution

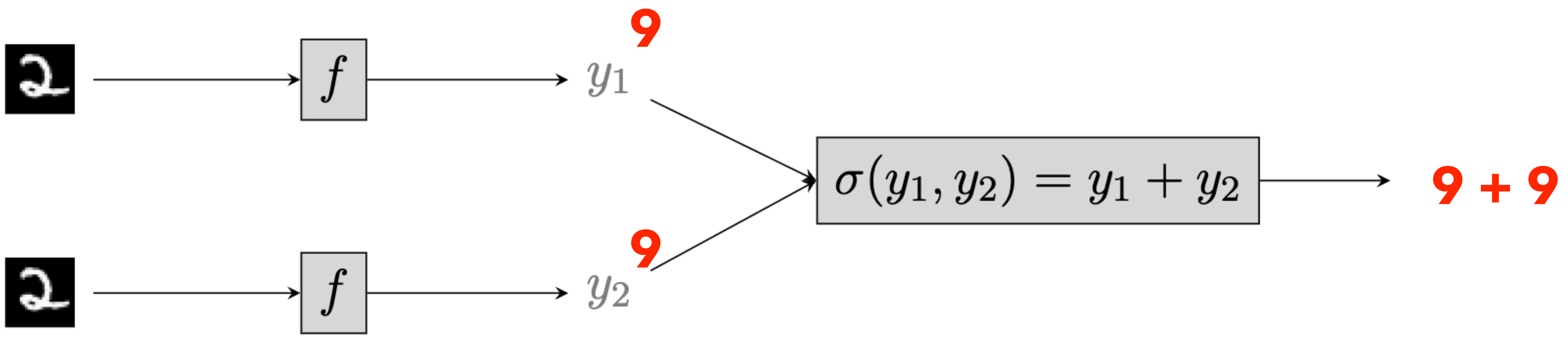
In other words, **mistakes in the overall output**, should be informative of the **classification errors** made by f, under ***any*** training distribution

Assumption: The space of classifiers $\mathscr{F}$ includes a classifier f^* that minimises $R_{\mathsf{P}}^{01}(f;\sigma)$

PAC-Learnability: Known and Deterministic σ

If an instance is learnable under **any** distribution, then it should be learnable under the ***spike***

PAC-Learnability: Known and Deterministic σ



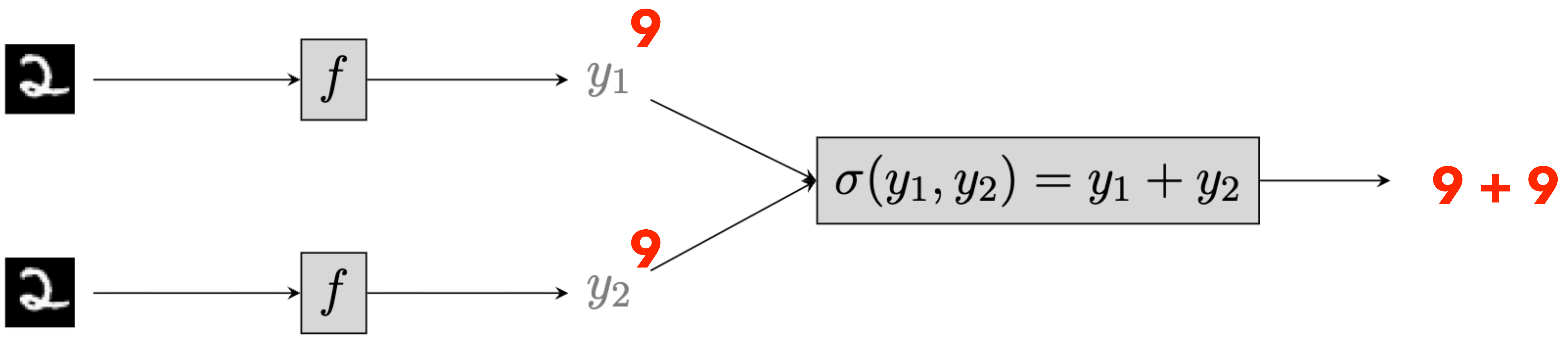
Suppose the following:

✓All mass is concentrated in  with gold label 2

✓f misclassifies  as **9**. Hence, the gold labels are **(2,2)**, but f outputs **(9,9)**

Question: Are f's classification errors concealed or not?

PAC-Learnability: Known and Deterministic σ



Suppose the following:

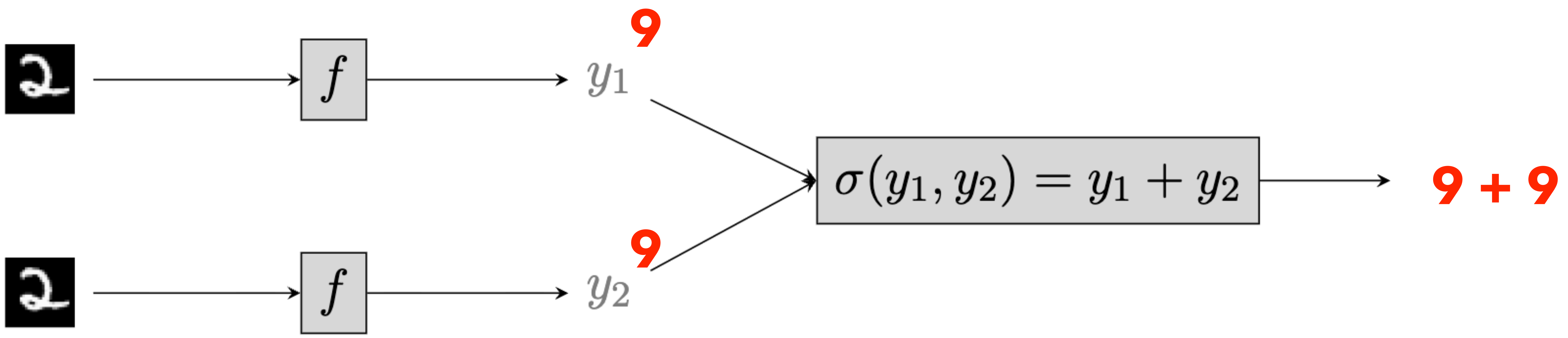
✓All mass is concentrated in  with gold label 2

✓f misclassifies  as **9**. Hence, the gold labels are **(2,2)**, but f outputs **(9,9)**

Question: Are f's classification errors concealed or not?

Answer: No, since **2 + 2** $\neq$ **9+9**

PAC-Learnability: Known and Deterministic σ



Question: Can you generalise this reasoning to a condition σ should abide by to ensure that the errors are not concealed?

Suppose the following:

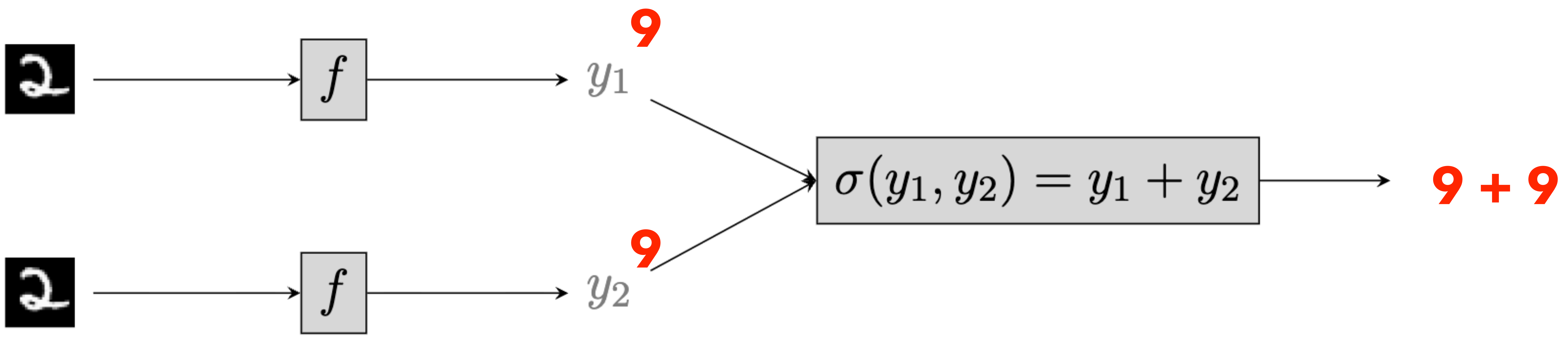
✓All mass is concentrated in  with gold label 2

✓f misclassifies  as **9**. Hence, the gold labels are **(2,2)**, but f outputs **(9,9)**

Question: Are f's classification errors concealed or not?

Answer: No, since **2 + 2** $\neq$ **9+9**

PAC-Learnability: Known and Deterministic σ



Reasoning: if for any **(y,…,y)** and **(y’,…,y’),** we have σ**(y,…,y)** $\neq \sigma$ **(y’,…,y’),** then the classification errors are **not concealed.**

Suppose the following:

✓All mass is concentrated in  with gold label 2

✓f misclassifies  as **9**. Hence, the gold labels are **(2,2)**, but f outputs **(9,9)**

Question: Are f's classification errors concealed or not?

Answer: No, since **2 + 2** $\neq$ **9+9**

PAC-Learnability: Known and Deterministic σ

Reasoning: if for any **(y,…,y)** and **(y′,…,y′),** we have σ**(y,…,y)** $\neq$ σ**(y′,…,y′),** then the classification errors are **not concealed.**

Definition (M**-unambiguity).** Component σ is M-unambiguous if for any **(y,…,y)** and **(y′,…,y′)** with **y** $\neq$ **y′,** we have σ**(y,…,y)** $\neq$ σ**(y′,…,y′).**

M-Unambiguity: Example

Question: is $\sigma(y_1, y_2) = y_1 + y_2$ M-unambiguous?

M-Unambiguity: Example

Question: is $\sigma(y_1, y_2) = y_1 \times y_2$ M-unambiguous?

M-Unambiguity: Example

Question: is $\sigma(y_1, y_2) = y_1 \oplus y_2$ M-unambiguous?

M-Unambiguity: Is It a Good Condition?

Definition (M-unambiguity). Component σ is M-unambiguous if for any **(y,…,y)** and **(y′,…,y′)** with **y** $\neq$ **y′**, we have σ**(y,…,y)** $\neq$ σ**(y′,…,y′).**

✓ M-unambiguous requires **invertibility only on inputs of the same class**

✓ This result essentially is more powerful than the results in **reasoning shortcuts** (Marconato et al., NeurIPS 2023) that suggest that **we cannot learn unless σ is invertible**

✓ Looser conditions can be obtained when the input data distribution is not a spike

M-Unambiguity: Proving Learnability

Recall: *To prove learnability of such a NeSY problem instance, we must* ***bound*** $R^{01}(f)$ *(**zero-one risk**) with* $R_{\mathsf{P}}^{01}(f;\sigma)$ *(**zero-one partial risk**), under **any** training distribution*

Theorem. If σ is M-unambiguous, then $R^{01}(f) \leq O(R_{\mathsf{P}}^{01}(f;\sigma)^{1/M})$.

M-Unambiguity: Proving Learnability

Theorem. If σ is M-unambiguous, then $R^{01}(f) \leq O(R_{\mathsf{P}}^{01}(f;\sigma)^{1/M})$.

Proof. Let $E_{i,j}(f)$ be the probability f misclassifies label i as j subject to samples from D. Then:

$$R^{01}(f) = \sum_{i \neq j} E_{i,j}(f) \quad \textbf{(*)}$$

Recall: Component σ is M-unambiguous if for any **(y,…,y)** and **(y',…,y')** with **y** $\neq$ **y'**, we have σ**(y,…,y)** $\neq \sigma$ **(y',…,y')**.

➡ *If all the M input instances have label i and are wrongly classified having label j, then the predicted partial label will be wrong.*

➡ $R_{\mathsf{P}}^{01}(f;\sigma)$ *is lower bounded by the sum of the same type of classification mistake being repeated M times:*

$$R_{\mathsf{P}}^{01}(f;\sigma) \geq \sum_{i \neq j} E_{i,j}(f)^M$$

The final result follows by combining the two equations and applying the power mean inequality.

M-Unambiguity: Proving Learnability

Theorem. If σ is M-unambiguous, then $R^{01}(f) \leq O(R_{\mathsf{P}}^{01}(f;\sigma)^{1/M})$.

To prove learnability, additionally, we need to bound the VC dimension and constructing counter-examples to show M-unambiguity is necessary

M-Unambiguity: Learnability Statement

Theorem (informal).

0-1 classification error

For any $\epsilon, \delta \in (0,1)$, with probability at least $1-\delta$, the empirical partial risk minimizer with $\widehat{\mathcal{R}}^{01}_{\mathsf{P}}(f;\sigma;\mathcal{T}_{\mathsf{P}}) = 0$ has a classification risk $\mathcal{R}^{01}(f) < \epsilon$, if

$$m_{\mathsf{P}} \geq C_0 \frac{c^{2M-2}}{\epsilon^M}\left(d_{[\mathcal{F}]} \log(6cM d_{[\mathcal{F}]}) \log\left(\frac{c^{2M-2}}{\epsilon^M}\right) + \log\left(\frac{1}{\delta}\right)\right)$$

Empirical error in getting a wrong overall output

Number of training samples

Finite Natarajan dimension

Learnability vs Training Neural Networks

Recall: To prove learnability we considered minimizing the empirical risk $R_{\mathsf{P}}^{01}(f; \sigma)$ given a set of partial samples

Minimizing $R_{\mathsf{P}}^{01}(f; \sigma)$ would be inefficient in practice

Practical Options (recap):

✓**Losses based on probabilistic logic semantics**
- ✓We discussed this yesterday – More will be covered in the next days

✓**Losses based on fuzzy logic semantics**
- ✓We discussed this yesterday

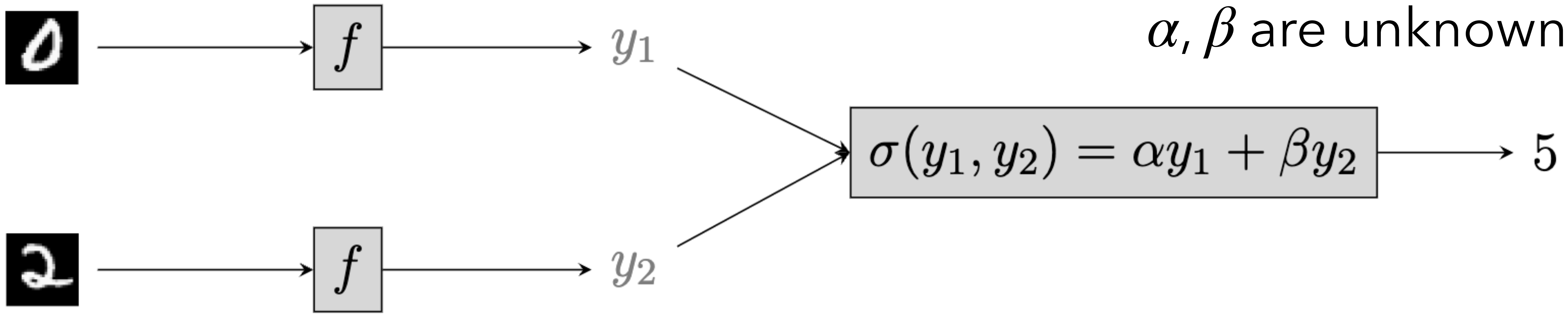
✓**Learning based on integer linear programming**
- ✓We will discuss this tomorrow

✓**Learning based on expectation maximization**

✓**Learning via differentiation through argmax**

PAC-Learnability: Unknown and Deterministic σ

Example



α, β are unknown

Question: Is this learning setting any interesting?

Learnability

Objective: Develop necessary (and sufficient) conditions that must be satisfied to ensure classifier PAC learnability. σ belongs to a family of parameterized functions $\mathscr{G}$.

Cases:

✓ Known and deterministic σ

  ✓ M-unambiguity

✓ Unknown and deterministic σ

  ✓ $\mathscr{G}$-unambiguity

Notation

Supervised learning	**NeSy**	**Meaning**
x (given)	$\boldsymbol{x} = x_1, \ldots, x_m$ (given)	input(s)
y (given)	$\boldsymbol{y} = y_1, \ldots, y_m$ (unknown)	gold label(s)
-	$s = \sigma(\boldsymbol{y})$ (given)	partial label
-	σ (unknown)	transition function
-	$\mathcal{G}$ (given)	transition space
$\mathcal{D}$	$\mathcal{D}_{\mathsf{P}}$	training distribution
$[f](x)$	$[f](x)$	prediction
$\ell^{01}(y, y') := 1\{y \neq y'\}$	$\ell^{01}_{\sigma}(\boldsymbol{y}, s) := 1\{\sigma(\boldsymbol{y}) \neq s\}$	zero-one (partial) loss
$\mathcal{R}^{01}(f) :=$ $E_{(X,Y)\sim\mathcal{D}}[\ell^{01}([f](X), Y)]$	$\mathcal{R}^{01}_{\mathsf{P}}(f;\sigma) :=$ $E_{(\mathbf{X},S)\sim\mathcal{D}_{\mathsf{P}}}[\ell^{01}_{\sigma}([f](\mathbf{X}), S)]$	zero-one (partial) risk

Recap: PAC-Learnability

A problem instance is PAC-learnable if there exists an algorithm $\mathscr{A}$ such that for any two user parameters ϵ and δ the following holds under any input distribution:

✓with probability at least 1-δ

✓the learned classifier misclassifies an input with probability $\leq \epsilon$

✓when given at least $m_{\epsilon,\delta}$ samples

Polynomial in ϵ and δ

Empirical Risk Minimizer (ERM) Learning

Known and deterministic σ

We will focus on an algorithm $\mathscr{A}$ that finds the classifier that minimizes the empirical risk $R^{01}_{\mathsf{P}}(f; \sigma)$ given a set of partial samples

Unknown and deterministic σ

We will focus on an algorithm $\mathscr{A}$ that finds the classifier that minimizes
$R^{01}_{\mathsf{P}}(f; \mathscr{G}) := \inf_{\sigma^* \in \mathscr{G}} R^{01}_{\mathsf{P}}(f; \sigma^*)$

Proving PAC-Learnability: Unknown and Deterministic σ

Known and deterministic σ

To prove learnability of a NeSY problem instance, we must **bound** $R^{01}(f)$ (**zero-one risk**) with $R_{\mathsf{P}}^{01}(f;\sigma)$ (**zero-one partial risk**), under ***any*** training distribution

In other words, **mistakes in the overall output**, should be informative of the **classification errors** made by f, under ***any*** training distribution

Unknown and deterministic σ

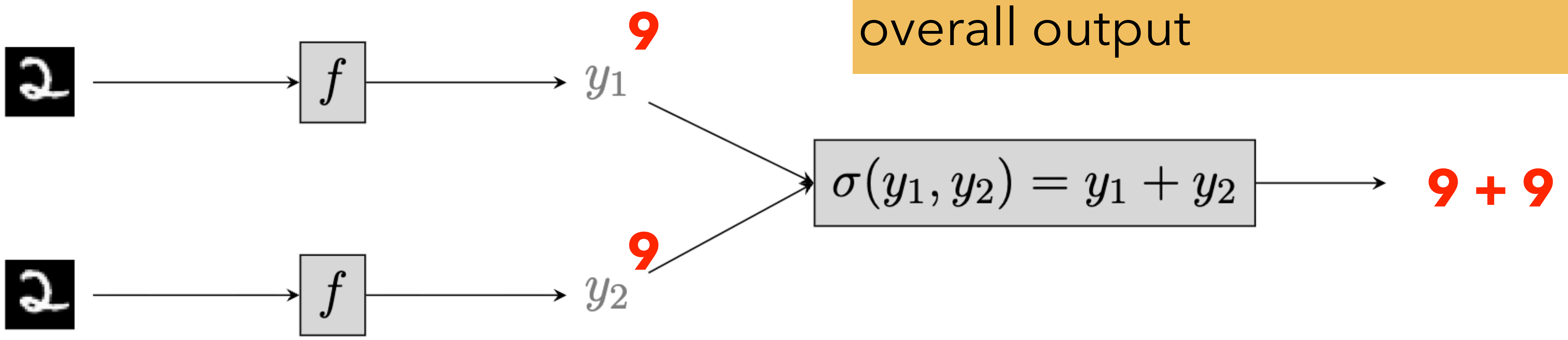
Similarly, to prove learnability of such a NeSY problem instance, we must **bound** $R^{01}(f)$ (**zero-one risk**) with $R_{\mathsf{P}}^{01}(f;\mathcal{G}) := \inf_{\sigma^* \in \mathcal{G}} R_{\mathsf{P}}^{01}(f;\sigma^*)$ under ***any*** training distribution

Assumption: The space of classifiers $\mathcal{F}$ includes a classifier f^* that minimises $R_{\mathsf{P}}^{01}(f;\sigma)$

PAC-Learnability: Unknown and Deterministic σ

Known and deterministic σ

Suffices to look at the mistakes in the overall output



Unknown and deterministic σ

Challenge. When σ is unknown, then if we replace σ with another σ' in the same family, we might get the same overall output.

PAC-Learnability: Unknown and Deterministic σ

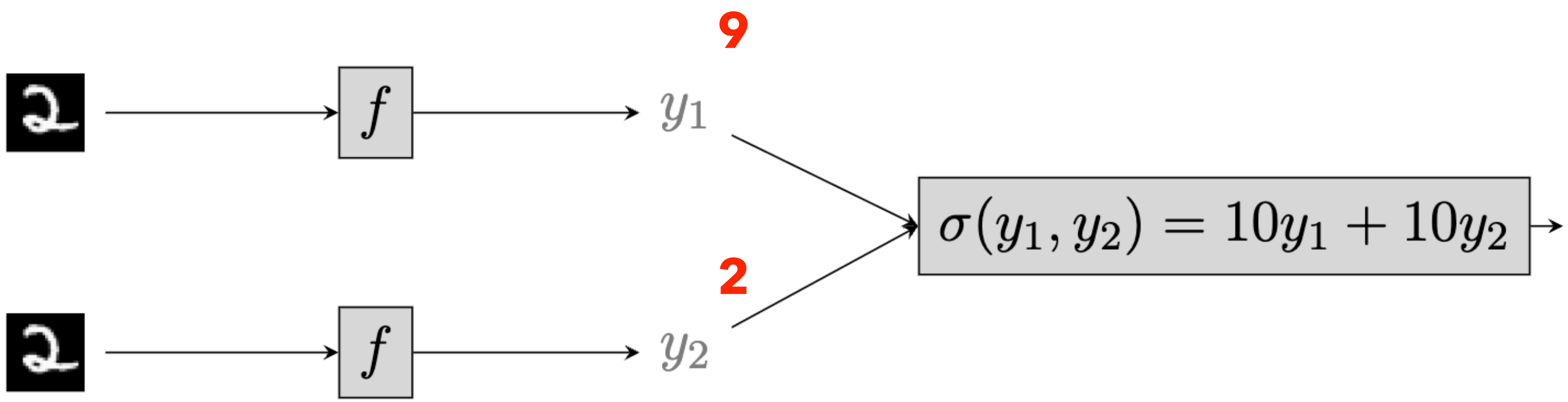
Known and deterministic σ

If an instance is learnable under **any** distribution, then it should be learnable under the ***spike***

Unknown and deterministic σ

Additional condition. Bounded classification risk: There exists $r > 0$, such that the probability each classifier predicts the correct label is $> r$

Challenges



Suppose the following:

✓All mass is concentrated in gold label **2**

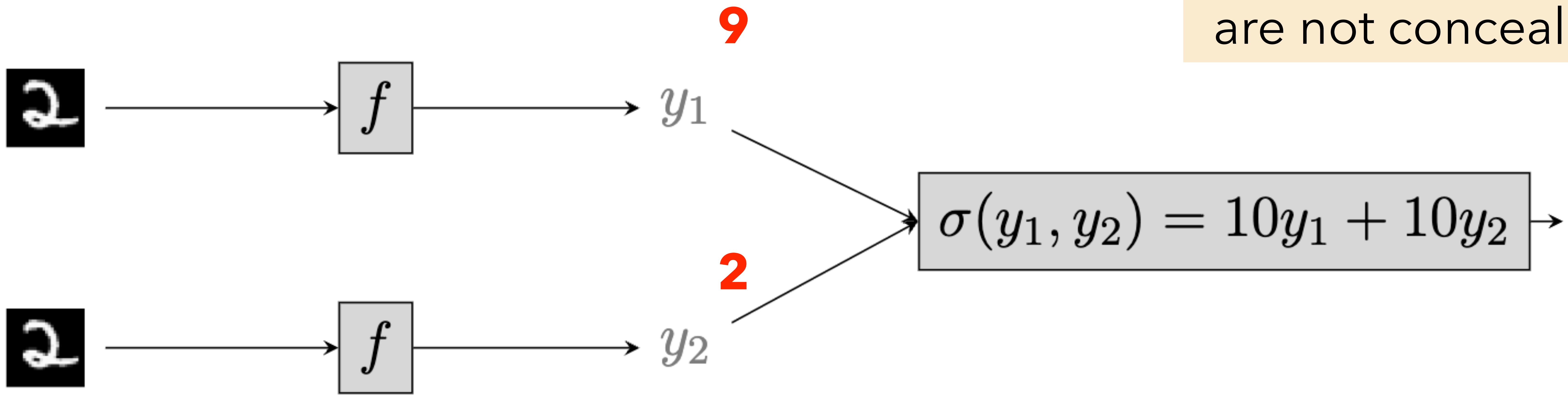
✓f misclassifies **2** as **9** with some probability. Hence, the gold labels are **(2,2)**, but f outputs **(9,2), (2,9), (9,9)**

✓**Gold** σ = **10**y_1 **+ 10**y_2, **non-gold** σ = **20**y_1 **+ 20**y_2

Question: are f's classification errors concealed or not for σ?

Challenges

Question: Can we generalize this reasoning to a condition $\mathscr{G}$ should satisfy so that the errors are not concealed?



Suppose the following:

✓All mass is concentrated in gold label **2**

✓f misclassifies **2** as **9** with some probability. Hence, the gold labels are **(2,2)**, but f outputs **(9,2), (2,9), (9,9)**

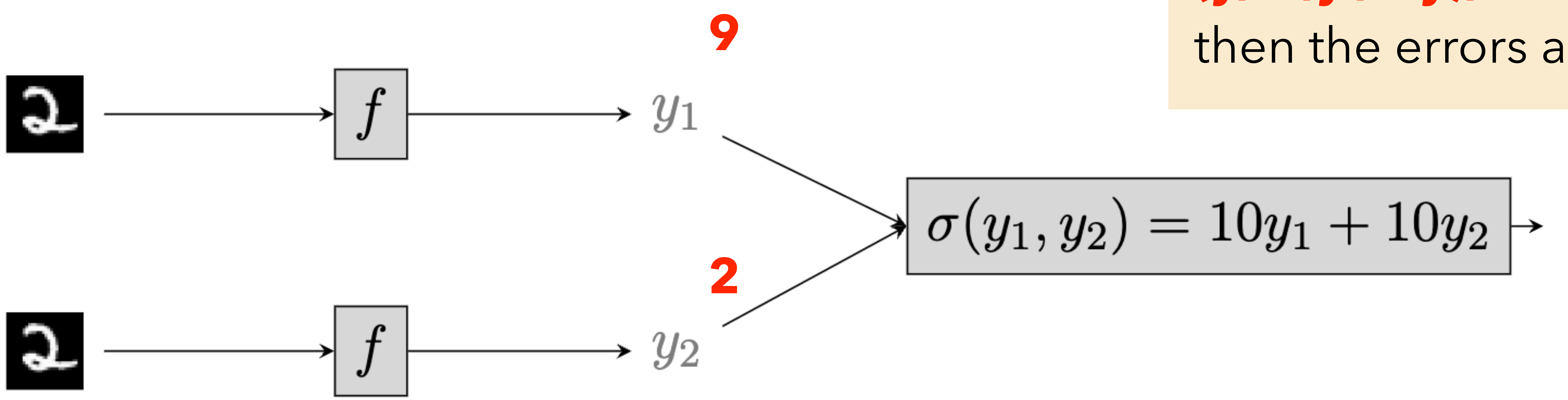
✓**Gold** σ = **10**y_1 **+ 10**y_2, **non-gold** σ = **20**y_1 **+ 20**y_2

Question: are f's classification errors concealed or not for σ?

Answer: no, since **10 * 2 + 10 * 2** $\neq$ **20 * 9 + 20 * 2**

Challenges

Reasoning: Given **gold σ,** if for **any label vector (y,…,y), any non-gold σ'** and **any y',** σ **(y,…,y)** $\neq \sigma'$ **(y,…,y',…y),** for **some (y,…,y',…y),** then the errors are not **concealed**



Suppose the following:

✓All mass is concentrated in gold label **2**

✓f misclassifies **2** as **9** with some probability. Hence, the gold labels are **(2,2)**, but f outputs **(9,2), (2,9), (9,9)**

✓**Gold $\sigma = 10y_1 + 10y_2$, non-gold $\sigma' = 20y_1 + 20y_2$**

Question: are f's classification errors concealed or not for σ?

Answer: no, since **10 * 2 + 10 * 2** $\neq$ **20 * 9 + 20 * 2**

PAC-Learnability: Known and Deterministic σ

Reasoning: Given **gold σ,** if for **any label vector (y,…,y), any non-gold σ'** and **any y', σ (y,…,y) $\neq$ σ' (y,…,y',…y),** for **some (y,…,y',…y),** then the errors are not **concealed**

Definition ($\mathscr{G}$-unambiguity). Space $\mathscr{G}$ is unambiguous if for a given σ**,** we following hold: for **any label vector (y,…,y), any non-gold σ'** and **any y', σ (y,…,y) $\neq$ σ' (y,…,y',…y),** for **some (y,…,y',…y).**

$\mathcal{G}$-Unambiguity: Example

Question: is $\mathcal{G} = \{(y_1, y_2) \mapsto \alpha y_1 + \beta y_2 \,|\, (\alpha, \beta) \in \mathbb{R}^2 - \{(0,0)\}\}$ unambiguous?

Recall. Space $\mathcal{G}$ is unambiguous if for a given σ**,** we following hold: for **any label vector (y,…,y), any non-gold** σ' and **any y',** σ **(y,…,y)** $\neq \sigma'$ **(y,…,y',…y),** for **some (y,…,y',…y).**

$\mathcal{G}$-Unambiguity: Example

Question: is $\mathcal{G} = \{(y_1, y_2) \mapsto \alpha y_1 + \beta y_2 \,|\, (\alpha, \beta) \in \mathbb{R}^2 - \{(0,0)\}\}$ unambiguous?

Recall. Space $\mathcal{G}$ is unambiguous if for a given σ**,** we following hold: for **any label vector (y,…,y), any non-gold** σ' and **any y',** σ **(y,…,y)** $\neq \sigma'$ **(y,…,y',…y),** for **some (y,…,y',…y).**

Answer. No

✓$\sigma = (y_1, y_2) \mapsto y_1 - y_1^2 + y_2 - y_2^2$

✓$\sigma' : (y_1, y_2) \mapsto y_1 - y_1^2 - y_2 + y_2^2$

✓**y = 0**

✓**y' = 1**

✓σ **(0,0)** $= \sigma'$ **(1,0)** $= \sigma'$ **(1,1)** $= \sigma'$ **(1,1)**

$\mathscr{G}$-Unambiguity: Proving Learnability

Recall: To prove learnability of such a NeSY problem instance, we must **bound** $R^{01}(f)$ (**zero-one risk**) with $R^{01}_{\mathsf{P}}(f; \mathscr{G}) := \inf_{\sigma^* \in \mathscr{G}} R^{01}_{\mathsf{P}}(f; \sigma^*)$ under ***any*** training distribution

Theorem. If σ is $\mathscr{G}$-unambiguous, then $R^{01}(f) \leq O(R^{01}_{\mathsf{P}}(f; \mathscr{G})^{1/M})$.

$\mathcal{G}$-Unambiguity: Learnability Statement

Theorem (informal).

0-1 classification error

For any $\epsilon, \delta \in (0,1)$, with probability at least $1-\delta$, the empirical partial risk minimizer with $\widehat{\mathcal{R}}^{01}_{\mathsf{P}}(f;\sigma;\mathcal{T}_{\mathsf{P}}) = 0$ has a classification risk $\mathcal{R}^{01}(f) < \epsilon$, if

$$m_{\mathsf{P}} \geq C_4 \frac{c^{2M-2}}{r^M \epsilon^M}\left(\left((d_{[\mathcal{F}]} + d_{\mathcal{G}}) \log(6M(d_{[\mathcal{F}]} + d_{\mathcal{G}})) + d_{[\mathcal{F}]} \log c\right) \log\left(\frac{c^{2M-2}}{r^M \epsilon^M}\right) + \log\left(\frac{1}{\delta}\right)\right)$$

Empirical error in getting a wrong overall output

Number of training samples

Finite Natarajan dimension

Relationship With Other Weakly-Supervised Learning Settings

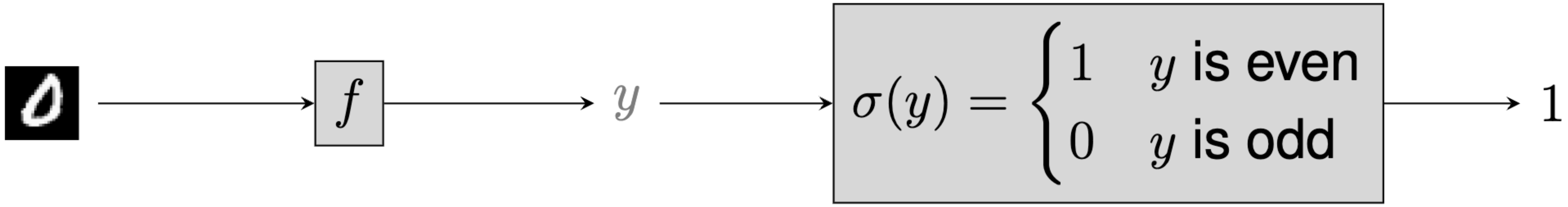
Outline of Today's Lecture

- ✓**Introduction to ML**
  - ✓Learning Paradigms
- ✓**Introduction to NeSY**
- ✓**Learnability**
  - ✓Definition
  - ✓Learnability vs Training Deep Networks
  - ✓Known and Deterministic σ
    - ✓M-Unambiguity
  - ✓Unknown and Deterministic σ
    - ✓$\mathcal{G}$-Unambiguity
- ✓**Relationship with Other Weakly-Supervised Learning Settings**
  - ✓Partial Label Learning
  - ✓Learning via Transition Matrices

Relevant Settings

- ✓ Partial label learning
- ✓ Learning via transition matrices

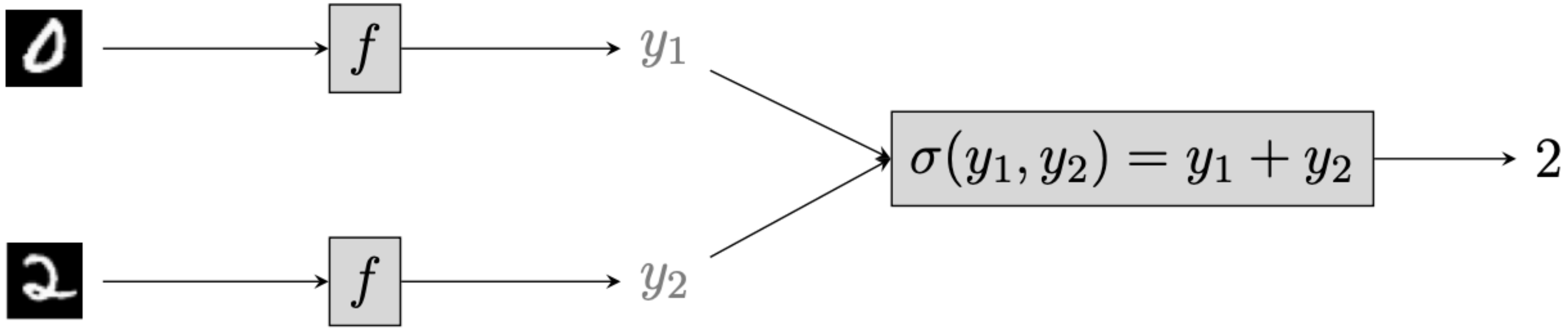
Partial Label Learning: Example

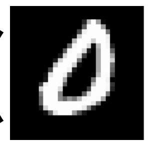


Training sample: (, {0,2,4,6,8})

Mutually exclusive set of candidate labels

NeSy Learning



Training sample: (( , ), {(0,2), (2,0), (1,1)})

Mutually exclusive set of candidate label vectors

Partial Label Learning vs NeSy Learning

✓Multiple vs single input

✓Deterministic vs non-deterministic σ, i.e.,

✓Prior learnability results rely on **small ambiguity** , i.e., that exists $\gamma < 1$, where

Gold label

$$\gamma := \sup_{D(x,y)>0 \wedge y' \neq y} \mathbb{P}_{(x,y)\sim D}(y' \in \sigma(y))$$

Density

Probability y' cooccurs with gold y in a training sample

Learnability under non-deterministic σ – proper extension of **small ambiguity** (Wang, Tsamoura, Roth, NeurIPS 2023).

Relevant Settings

- ✓ Partial label learning
- ✓ Learning via transition matrices

Learning via Transition Matrices

A **transition matrix** $\mathbf{T}$ for a learning problem with **hidden label** $Y$and **observed label** S is a **stochastic matrix**, where the element in its i-th column and j-th row is the conditional probability $\mathbb{P}(S = j \mid Y = i)$

hidden labels

$$
\mathbf{T} = \begin{array}{c} \\ s = 1 \\ \vdots \\ s = j \\ \vdots \\ s = |\mathcal{S}| \end{array}
\begin{array}{c}
\begin{array}{ccc} y = 1 & y = i & y = |\mathcal{Y}| \end{array} \\
\begin{bmatrix} \mathbb{P}(S = 1 | Y = 1) & & \\ & & \\ & \mathbb{P}(S = j | Y = i) & \\ & & \\ & & \mathbb{P}(S = |\mathcal{S}| | Y = |\mathcal{Y}|) \end{bmatrix}
\end{array}
$$

observed labels

Learning via Transition Matrices

A **transition matrix** $\mathbf{T}$ encodes the probability of getting the **observed label** S given then **hidden label** Y**.**

hidden labels

$$\mathbf{T} = \begin{array}{c} \\ s=1 \\ \vdots \\ s=j \\ \vdots \\ s=|\mathcal{S}| \end{array} \begin{array}{c} \begin{array}{ccc} y=1 & y=i & y=|\mathcal{Y}| \end{array} \\ \begin{bmatrix} \mathbb{P}(S=1|Y=1) & & \\ & & \\ & \mathbb{P}(S=j|Y=i) & \\ & & \\ & & \mathbb{P}(S=|\mathcal{S}||Y=|\mathcal{Y}|) \end{bmatrix} \end{array}$$

observed labels

Learning via Transition Matrices

If **transition matrix $\mathbf{T}$** is **invertible**, then we can compute the **hidden data distribution**. In other words, we can **estimate the classification loss.**

$$o(\mathbf{x}) = \mathbf{T}(\mathbf{x}) \; h(\mathbf{x})$$

$$[\mathbb{P}(S=1 \,|\, \mathbf{x}), \ldots, \mathbb{P}(S=|\mathcal{S}| \,|\, \mathbf{x})]^{\top} = \begin{array}{c c} & \begin{array}{ccc} y=1 & y=i & y=|\mathcal{Y}| \end{array} \\ \begin{array}{c} s=1 \\ \vdots \\ s=j \\ \vdots \\ s=|\mathcal{S}| \end{array} & \begin{bmatrix} \mathbb{P}(S=1|Y=1) & & \\ & & \\ & \mathbb{P}(S=j|Y=i) & \\ & & \\ & & \mathbb{P}(S=|\mathcal{S}||Y=|\mathcal{Y}|) \end{bmatrix} \end{array} \quad [\mathbb{P}(Y=1 \,|\, \mathbf{x}), \ldots, \mathbb{P}(Y=|\mathcal{Y}| \,|\, \mathbf{x})]$$

$$h(\mathbf{x}) = \mathbf{T}^{+}(\mathbf{x}) \; o(\mathbf{x})$$

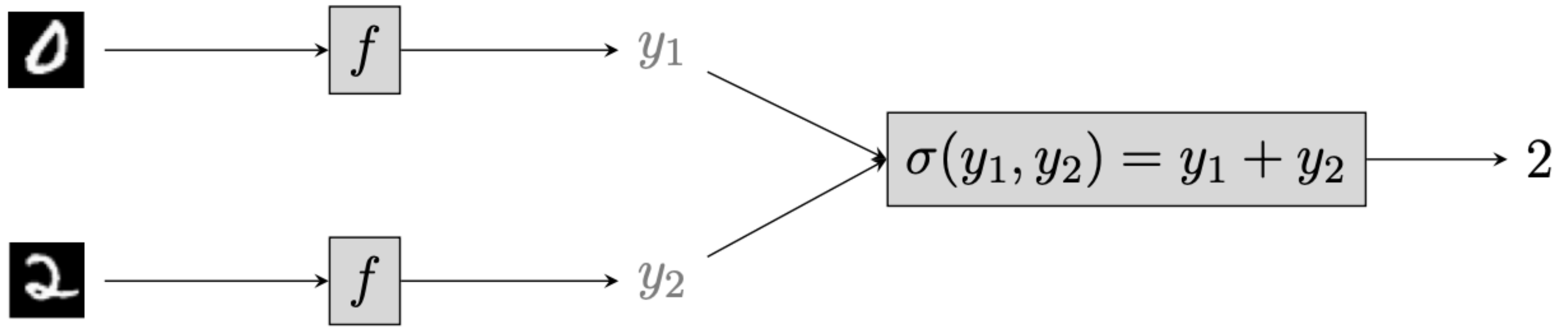
Learning via Transition Matrices

✓Supervised/semi-supervided learning

✓Noisy label learning

✓Partial label learning

Transition Matrix Formulation of 2SUM



hidden labels

observed labels

$$
\mathbf{T} = \begin{array}{c} \\ s=1 \\ \vdots \\ s=j \\ \vdots \\ s=|\mathcal{S}| \end{array}
\begin{array}{c}
\begin{array}{ccc} y=1 & y=i & y=|\mathcal{Y}| \end{array} \\
\begin{bmatrix} \mathbb{P}(S=1|Y=1) & & \\ & & \\ & \mathbb{P}(S=j|Y=i) & \\ & & \\ & & \mathbb{P}(S=|\mathcal{S}||Y=|\mathcal{Y}|) \end{bmatrix}
\end{array}
$$

Transition Matrix Formulation of 2SUM

hidden labels

$$\mathbf{T} = \begin{array}{c} \\ s=0 \\ s=1 \\ s=2 \\ \vdots \\ s=18 \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{y}=(0,0) & \boldsymbol{y}=(0,1) & \boldsymbol{y}=(1,0) \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & \vdots & \vdots \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

observed labels

Question: Is this reduction any good?

Transition Matrix Formulation of 2SUM

hidden labels

$$\mathbf{T} = \begin{array}{c} \\ s=0 \\ s=1 \\ s=2 \\ \vdots \\ s=18 \end{array} \begin{array}{c} \begin{array}{ccc} \boldsymbol{y}=(0,0) & \boldsymbol{y}=(0,1) & \boldsymbol{y}=(1,0) \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & \vdots & \vdots \\ 0 & 0 & 0 \end{bmatrix} \end{array}$$

observed labels

Question: Is this reduction any good? **No**

Answer: The reduction works only when σ is 1-1

Learning via Transition Matrices (Better Formulation)

✓The previous reduction suggests that we need **randomness**

✓Randomness can come from the **distribution of input instances**

Learning via Transition Matrices (Better Formulation)

This is a possible training technique

$$\mathbf{T}_1 = \mathbf{T}_2 = \begin{array}{c} \\ s=0 \\ s=1 \\ \vdots \\ s=9 \\ s=11 \\ \vdots \\ s=18 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & \cdots & 9 \end{array} \\ \begin{bmatrix} \mathbb{P}(Y=0) & 0 & \cdots & 0 \\ \mathbb{P}(Y=1) & \mathbb{P}(Y=0) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{P}(Y=9) & \mathbb{P}(Y=8) & \cdots & \mathbb{P}(Y=0) \\ 0 & 0 & \cdots & \mathbb{P}(Y=2) \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbb{P}(Y=9) \end{bmatrix} \end{array}$$

Consider a label $y = 0$. The probability that the sum is 1 ($s = 1$), equals the probability that other label is 1 ($\mathbb{P}(Y = 1)$)

Learning via Transition Matrices vs NeSy

✓ M-unambiguity $\not\Rightarrow \mathbf{T}$ invertibility

✓ $\mathbf{T}$ invertibility $\not\Rightarrow M$-unambiguity

$\mathbf{T}$ invertibility $\not\Rightarrow$ M-unambiguity

Encoding 2SUM

$$\mathbf{T}_1 = \mathbf{T}_2 = \begin{array}{cc} & \begin{array}{cccc} 0 & 1 & \cdots & 9 \end{array} \\ \begin{array}{c} s=0 \\ s=1 \\ \vdots \\ s=9 \\ s=11 \\ \vdots \\ s=18 \end{array} & \begin{bmatrix} \mathbb{P}(Y=0) & 0 & \cdots & 0 \\ \mathbb{P}(Y=1) & \mathbb{P}(Y=0) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbb{P}(Y=9) & \mathbb{P}(Y=8) & \cdots & \mathbb{P}(Y=0) \\ 0 & 0 & \cdots & \mathbb{P}(Y=2) \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \mathbb{P}(Y=9) \end{bmatrix} \end{array}$$

Consider the following:

- ✓Two hidden labels {1,2}
- ✓$\sigma(1,1) = \sigma(2,2) = 1$
- ✓$\sigma(1,2) = \sigma(2,1) = 0$
- ✓$\mathbb{P}(Y = 1) = 0.1$
- ✓$\mathbb{P}(Y = 2) = 0.9$

Question: Does M-unambiguity hold?

Question: Is matrix $\mathbf{T}$ invertible?

More Results

✓Better convergence rates via forcing additional conditions

✓Learnability under multiple classifiers

✓Learnability under non-deterministic σ

✓Rademacher error bounds under approximate probabilistic losses

Kaifu Wang, Efthymia Tsamoura, and Dan Roth. On Learning Latent Models with Multi-Instance Weak Supervision. In NeurIPS, 2023.

Recap of Today's Lecture

- ✓**Introduction to ML**
  - ✓Learning Paradigms
- ✓**Introduction to NeSY**
- ✓**Learnability**
  - ✓Definition
  - ✓Learnability vs Training Deep Networks
  - ✓Known and Deterministic σ
    - ✓M-Unambiguity
  - ✓Unknown and Deterministic σ
    - ✓$\mathcal{G}$-Unambiguity
- ✓**Relationship with Other Weakly-Supervised Learning Settings**
  - ✓Partial Label Learning
  - ✓Learning via Transition Matrices