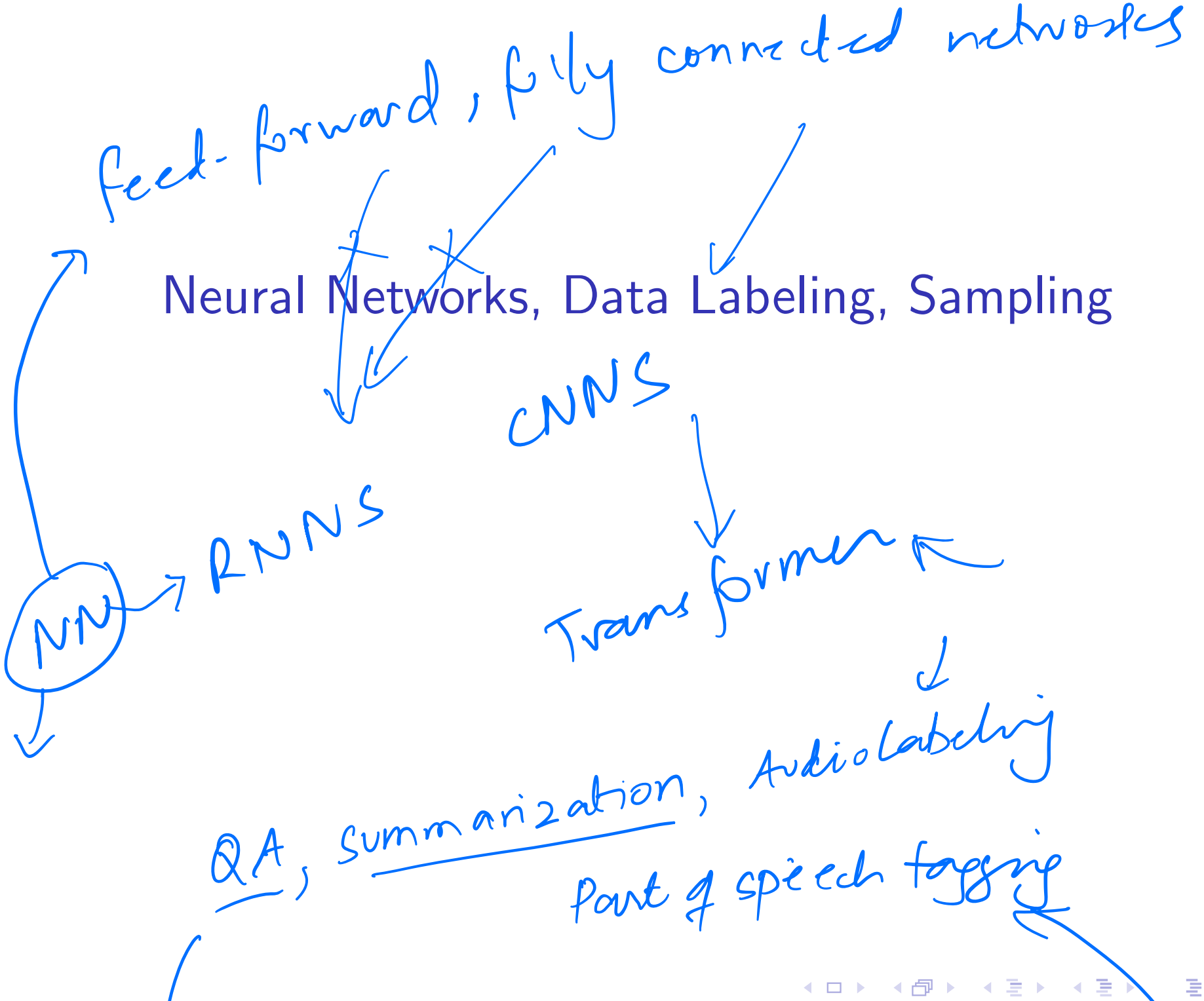
Neural Networks, Data Labeling, Sampling



Modern ML is based on neural networks

- The performance of the state of the art solutions for almost all ML tasks dramatically improved in the last decade.
- Almost all the solutions are based on deep neural networks.
- Somewhat surprisingly, modern solutions to very different tasks are based on the same architecture — derived from *transformers*.
- The best models have been evolving: RNNs, Transformers, ...
- So it is important to understand the fundamental concepts of neural networks.
- Reference texts: (i) "Artificial intelligence: a modern approach" by Stuart Russell and Peter Norvig. (ii) `deeplearningbook.org` by Goodfellow, Bengio, and Courville.

Extractive Question Answering

Text: Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion . . . third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California. As this was the 50th Super Bowl, the league emphasized the "golden anniversary" with various gold-themed initiatives, as well as temporarily suspending the tradition of naming each Super Bowl game with Roman numerals (under which the game would have been known as "Super Bowl L"), so that the logo could prominently feature the Arabic numerals 50.

Question: Where did Super Bowl 50 take place?

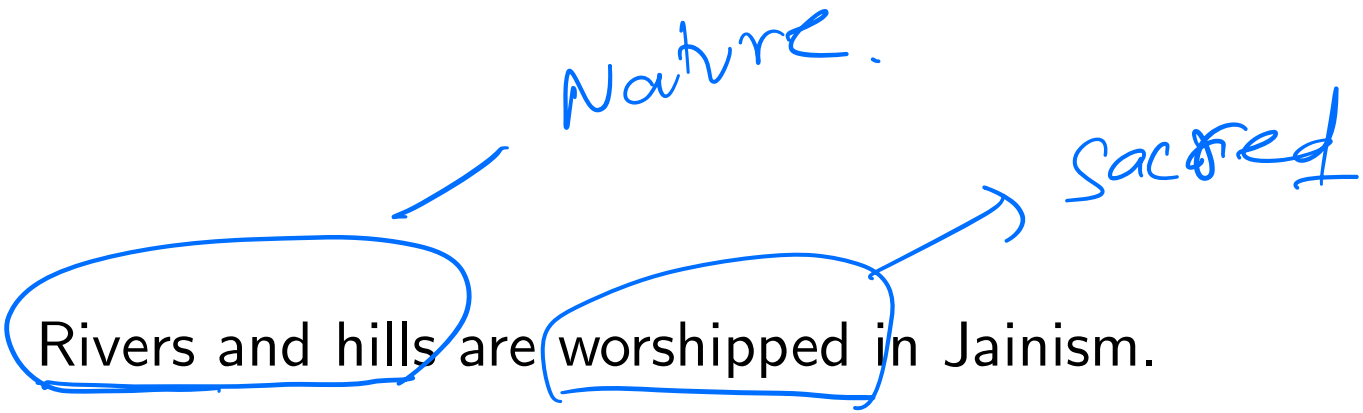
GT Answer: Levi's Stadium in the San Francisco Bay Area at Santa Clara, California.

Co-reference resolution

Location

Location

Natural Language Inference



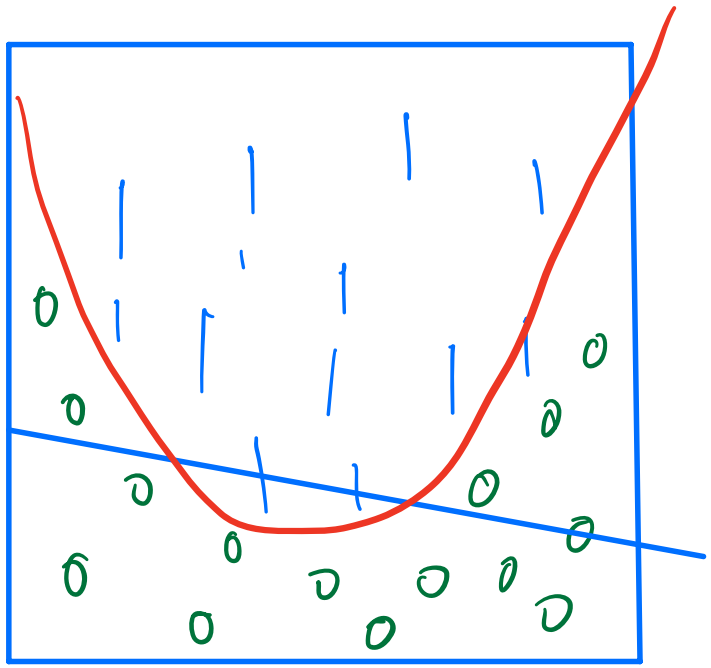
Premise: Rivers and hills are worshipped in Jainism.

Hypothesis: Nature is sacred in Jainism.

Answer: Supported.

What makes neural networks powerful?

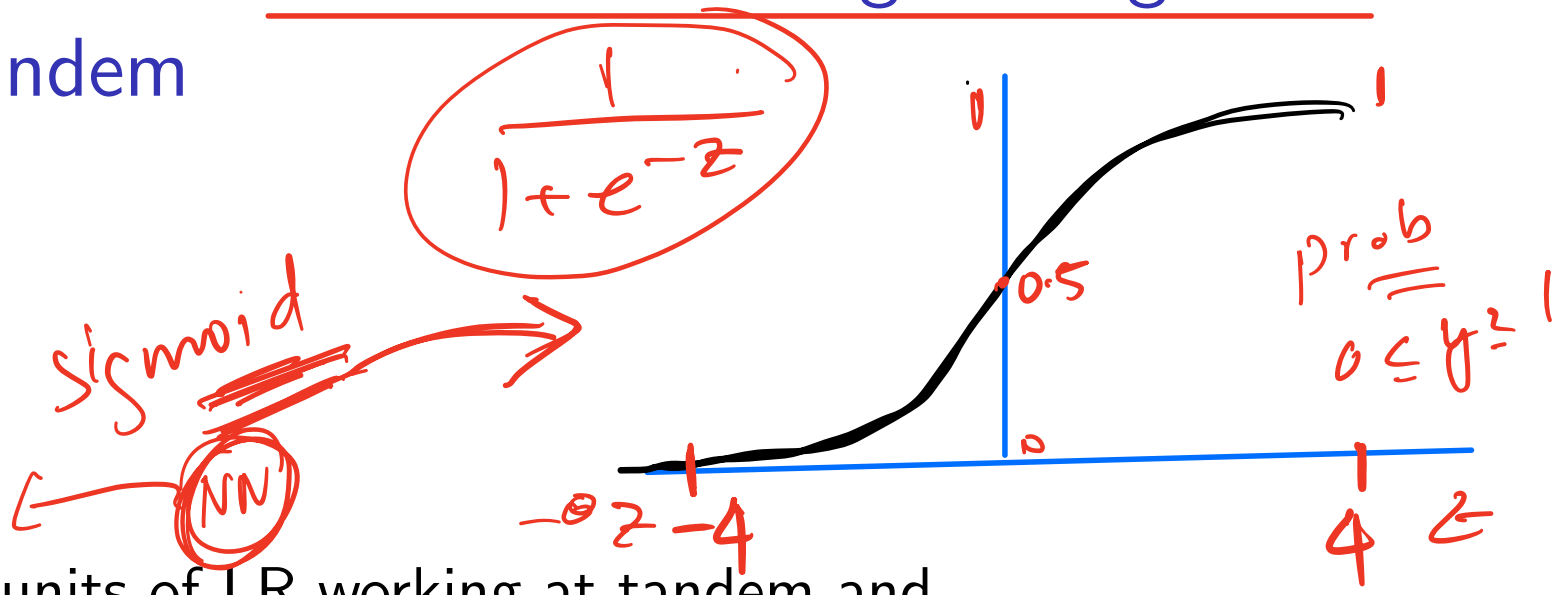
- What makes neural networks more powerful than, say, logistic regression?



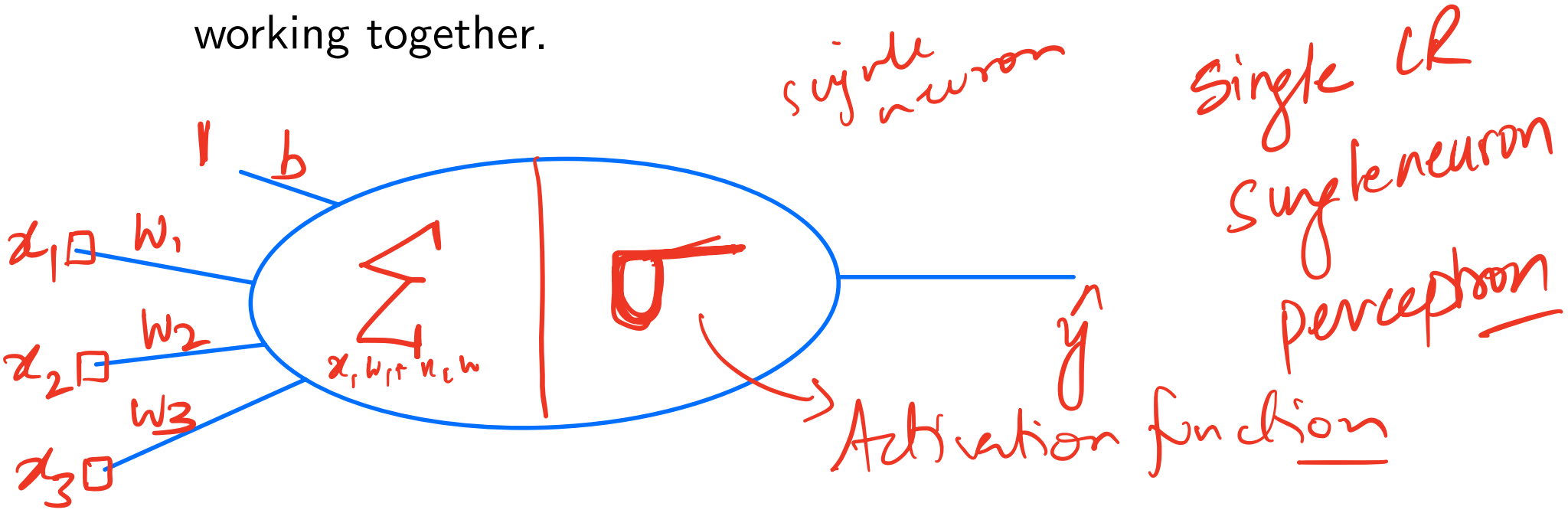
$z = wx + b$

non-linearity at the core

Neural networks are several units of logistic regression working in tandem

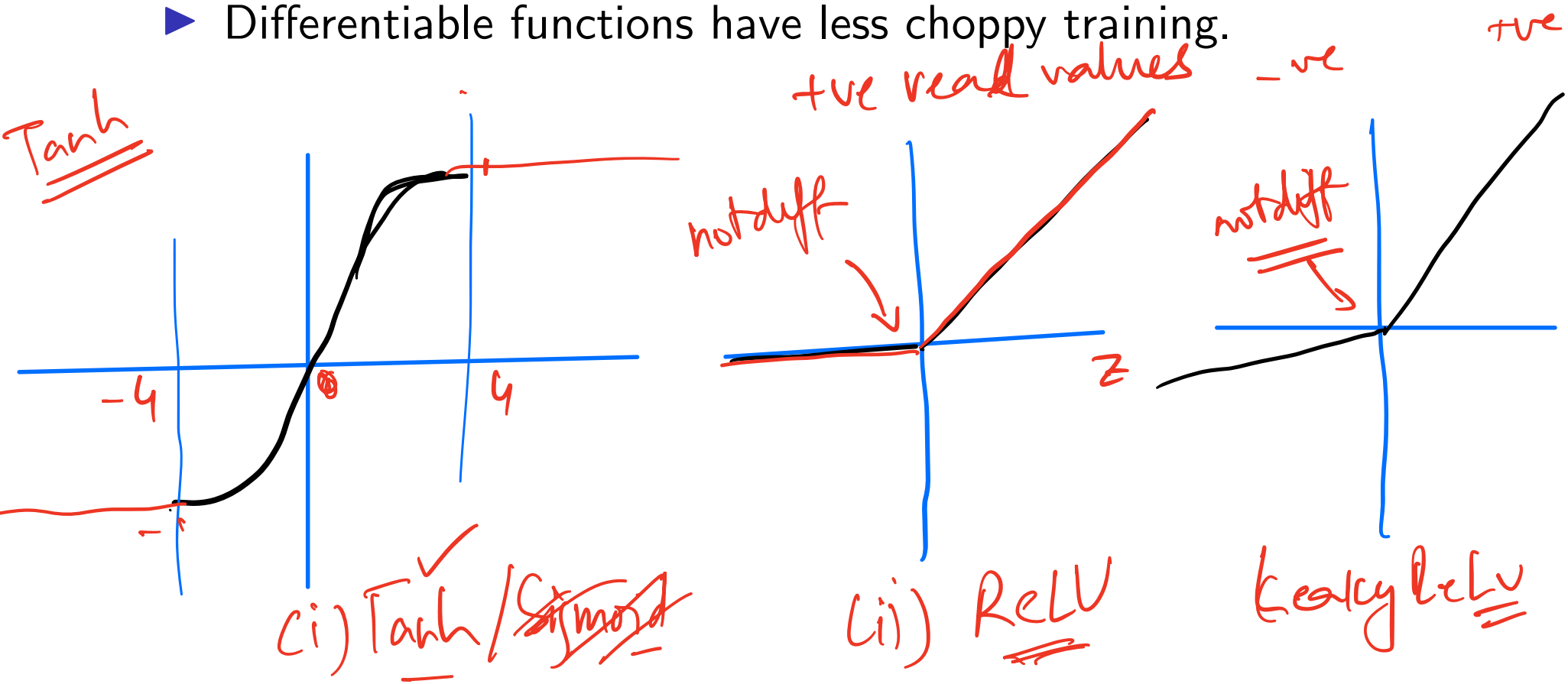


- several units of LR working at tandem and
- combine them into layers and see them as sequence of layers working together.



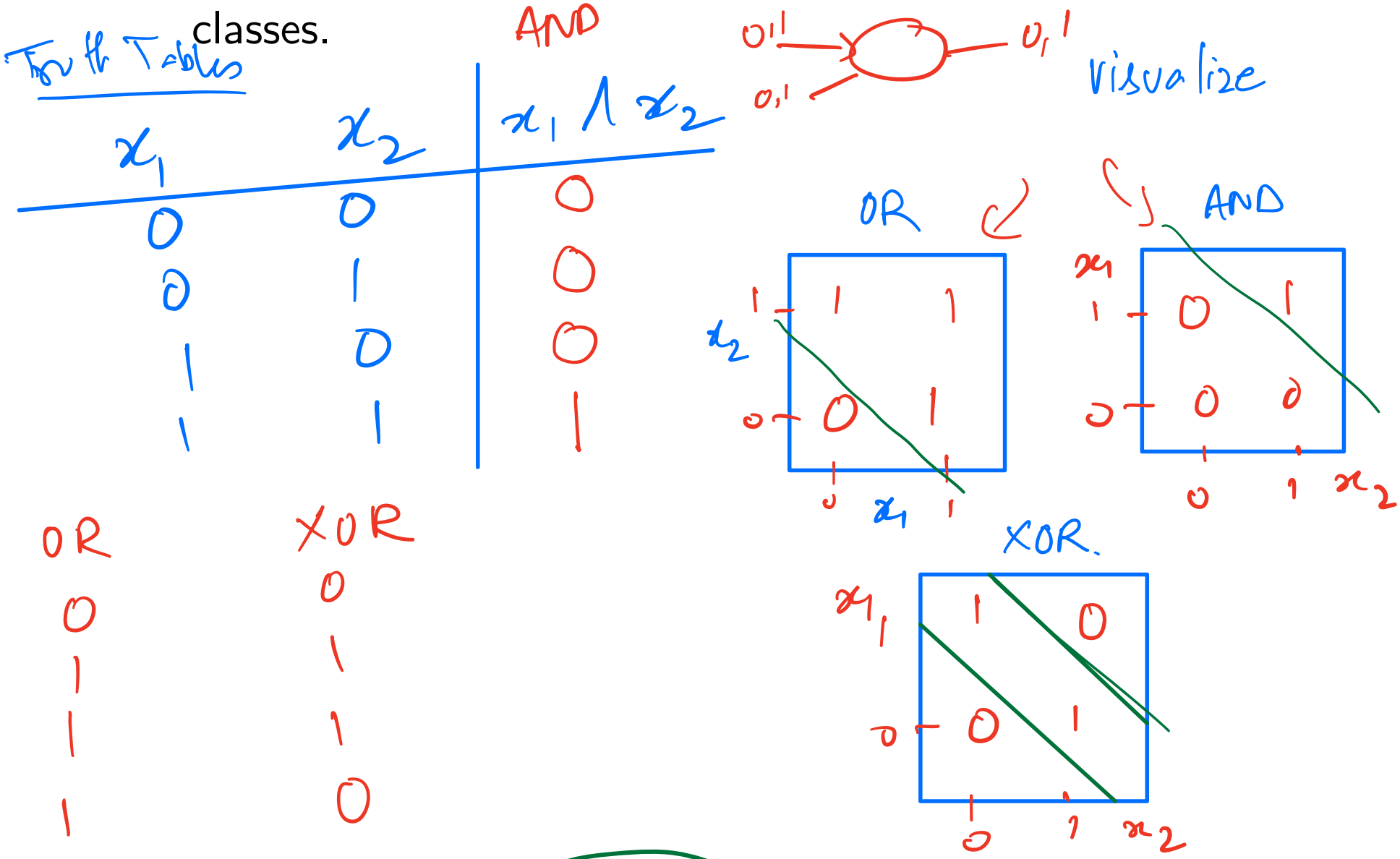
Activation functions

- There are different types of activation functions: Tanh, ReLU, Leaky ReLU. Each of them changes the output at $z = 0$ in a non-linear manner.
- The *range* is the interval of z in which the gradient in non-zero.
- Shorter ranges lead to more stable but inefficient training.
- Differentiable functions have less choppy training.



Logic gates

- AND, OR, XOR are common logic gates or Boolean functions.
- They can be viewed as classifying the inputs into 1 and 0 classes.

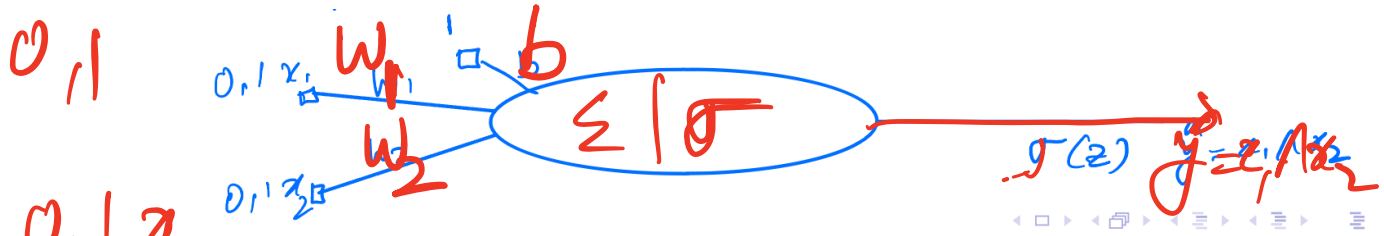


Implementing AND by a single neuron

- Can we manually determine parameter values such that a neuron behaves like an AND gate?
- Assume we have two Boolean inputs x_1, x_2. Also assume

$$\sigma(z) \approx \begin{cases} 1 \text{ when } z \geq 4 \\ 0 \text{ when } z \leq -4 \end{cases}$$

- Given data can the neuron learn to behave like an AND gate?



$z = w_1 x_1 + w_2 x_2 + b$

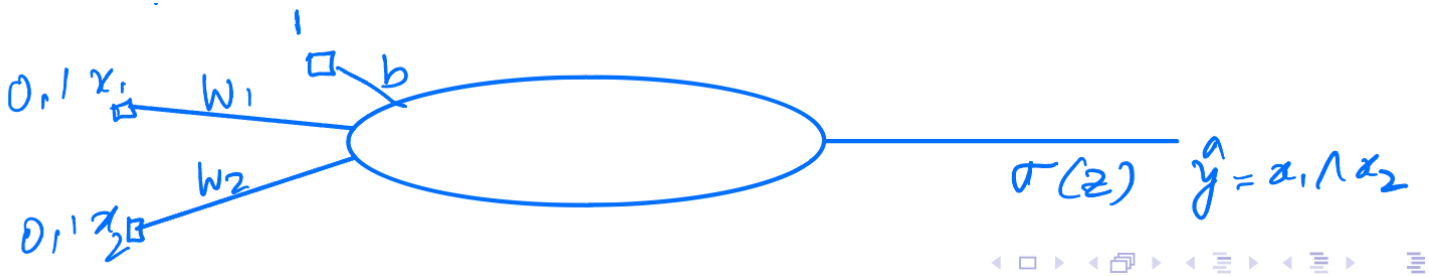
x_1	x_2	y $x_1 \wedge x_2$	z	conditions
0	0	0 ←	$z \leq -4$	$b \leq -4$
0	1	0 ←	$z \leq -4$	$w_1 + b \leq -4$
1	0	0 ←	$z \leq -4$	$w_2 + b \leq -4$
1	1	1	$z \geq 4$	$w_1 + w_2 + b \geq 4$

$b = -12$
$w_1 = 8$
$w_2 = 8$

Implementing ~~AND~~ XOR by a single neuron II

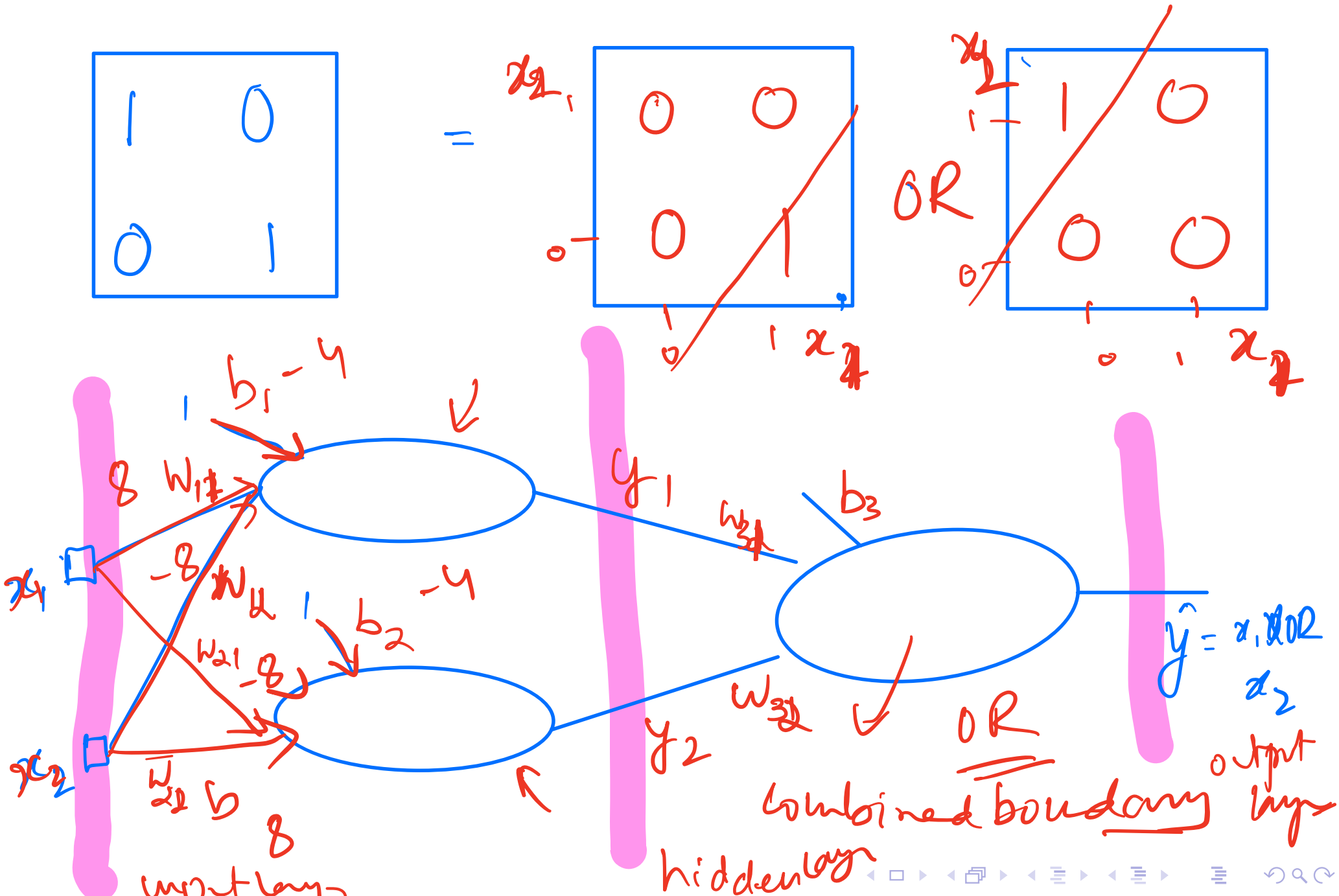
x_1	x_2	x_1 XOR x_2	z	conditions
0	0	0	$z \le -4$	$b \le -4$
0	1	1	$z \ge 4$	$w_2 + b \ge 4$
1	0	1	$z \ge 4$	$w_1 + b \ge 4$
1	1	0	$z \le -4$	$w_1 + w_2 + b \le -4$

inconsistent

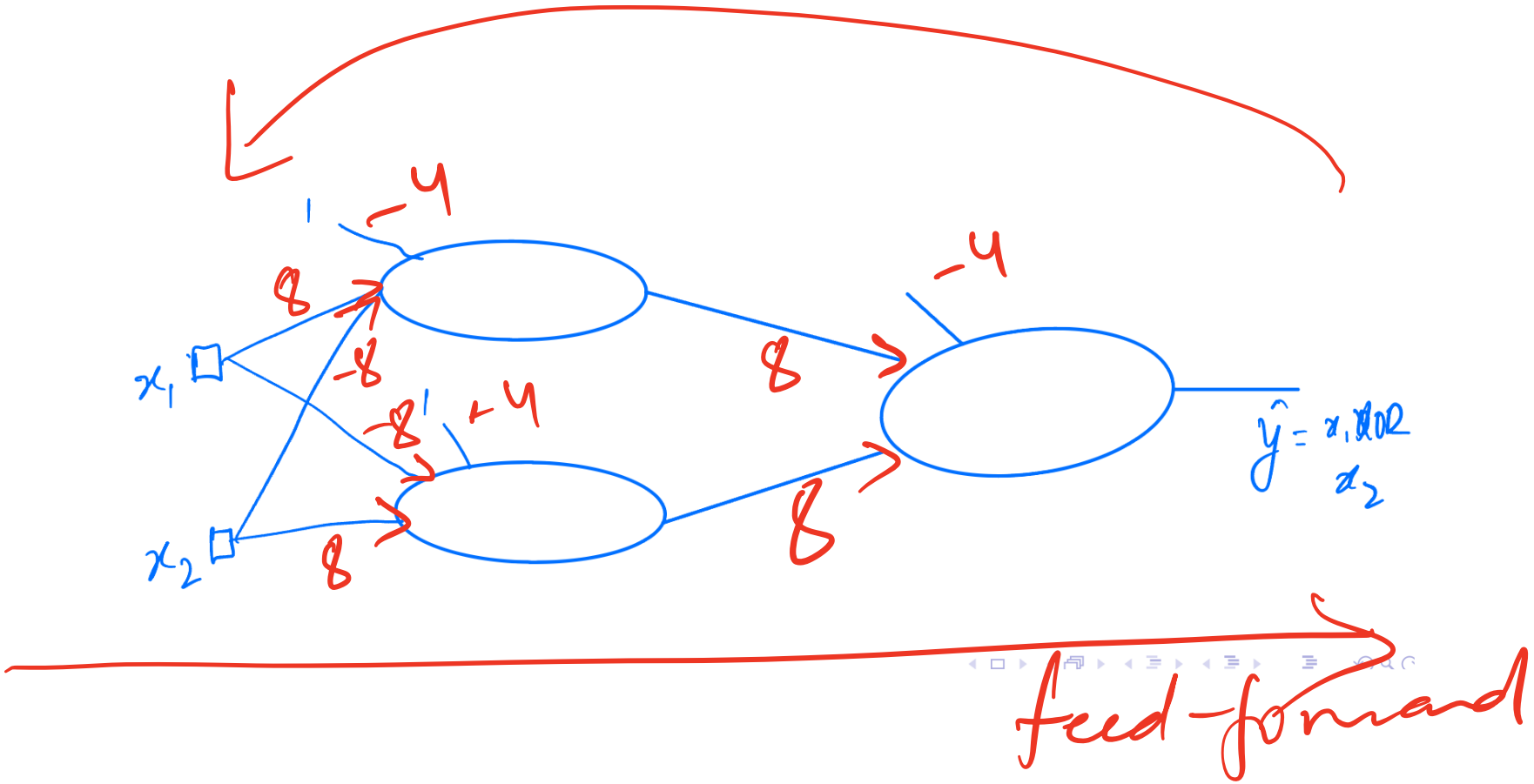


$b = -12$; $w_1 = 8$ $w_2 = 8$

Implementing XOR by a single neuron



Implementing XOR by a multiple neurons



Values:

-4	-4	-4
8	-8	8
-8	8	8

Implementing XOR by a multiple neurons

Input Layer

Hidden Layer

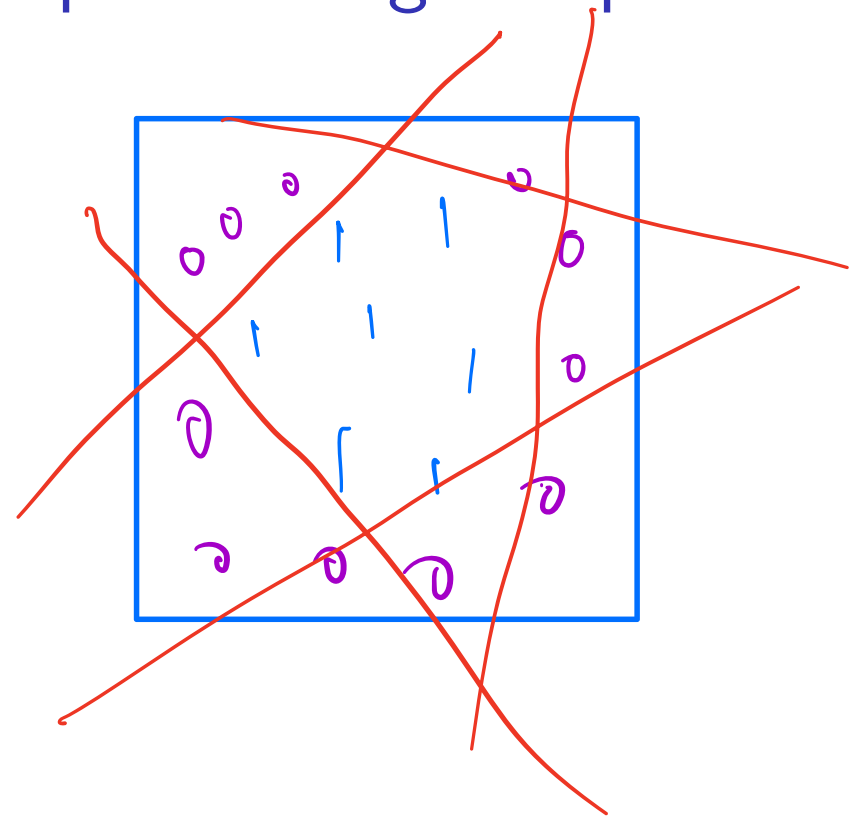
Output Layer

# of variables.

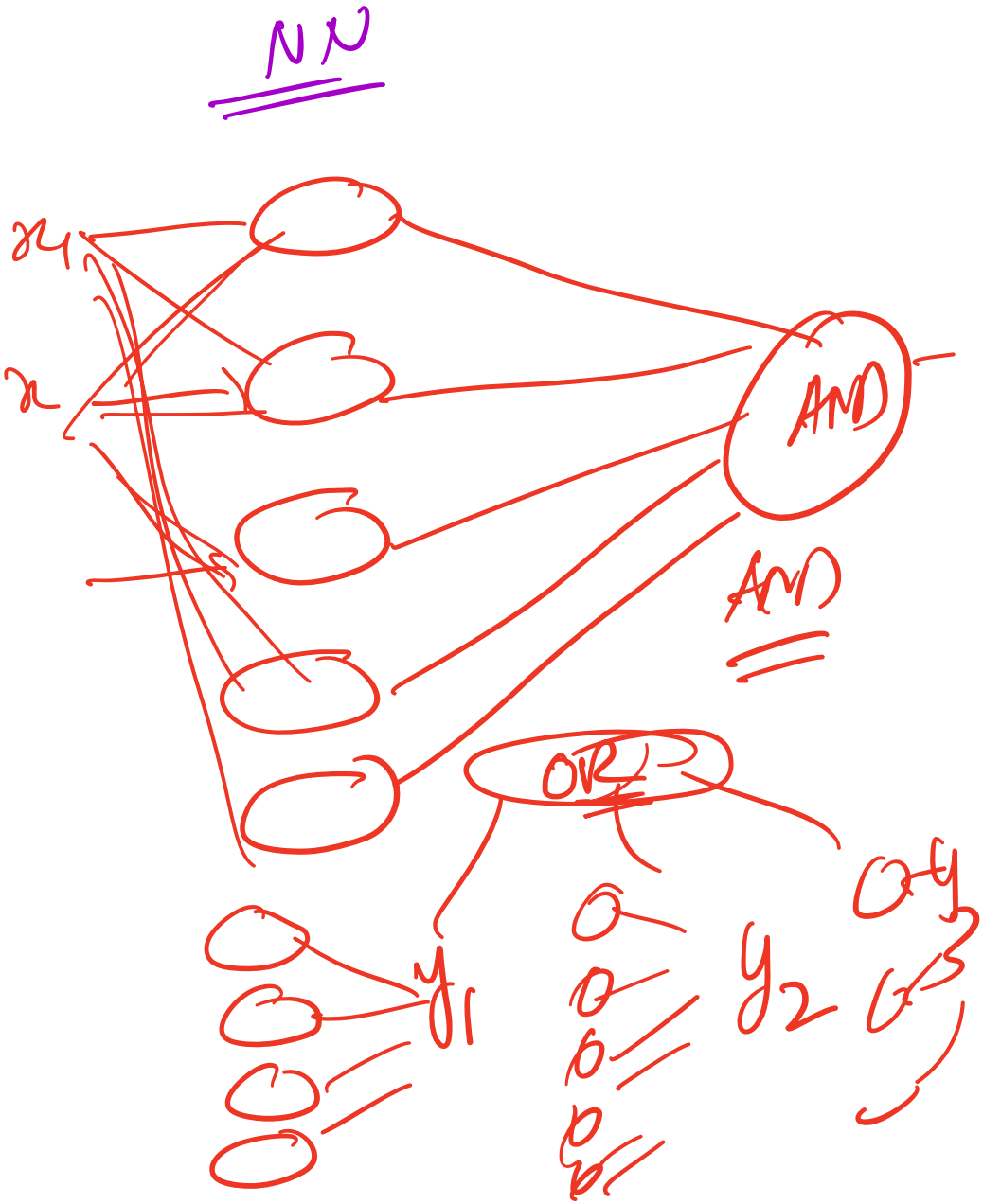
Feed-forward, fully connected.

Implementing XOR by a multiple neurons

Implementing complex class boundaries







Visualizing neural networks



http://playground.tensorflow.org

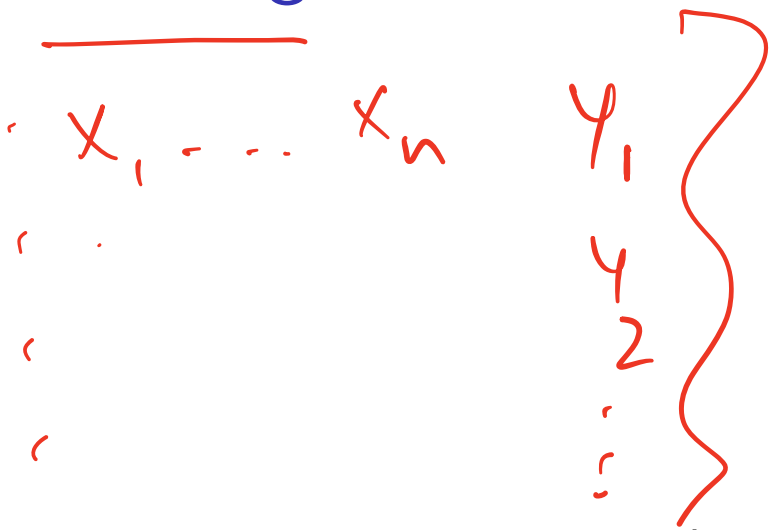
Viewing a neural network as a sequence of layers



- In mathematical notation:

$$
\begin{aligned}
\boldsymbol{z}^1_{1\times d_2} &= \boldsymbol{x} W^1_{d_1\times d_2} + b^1_{1\times d_2} \\
\boldsymbol{h}^1_{1\times d_2} &= g(\boldsymbol{z}^1) \\
\boldsymbol{z}^2_{1\times d_3} &= \boldsymbol{h}^1 W^2_{d_2\times d_3} + b^2_{1\times d_3} \\
\boldsymbol{h}^2_{1\times d_3} &= g(\boldsymbol{z}^2)
\end{aligned}
$$

Labeling



- All ML models (studied so far) are trained using labeled data.
- Often labels are provided by humans.

Human Challenges in Labeling

- Humans are not very good at labeling data consistently.
- Different humans may label the same data differently.
- Even the same human may label the same data differently at different times.
- Humans may be biased in their labeling.
- Humans may make mistakes in their labeling.
- Humans may not understand the data they are labeling.

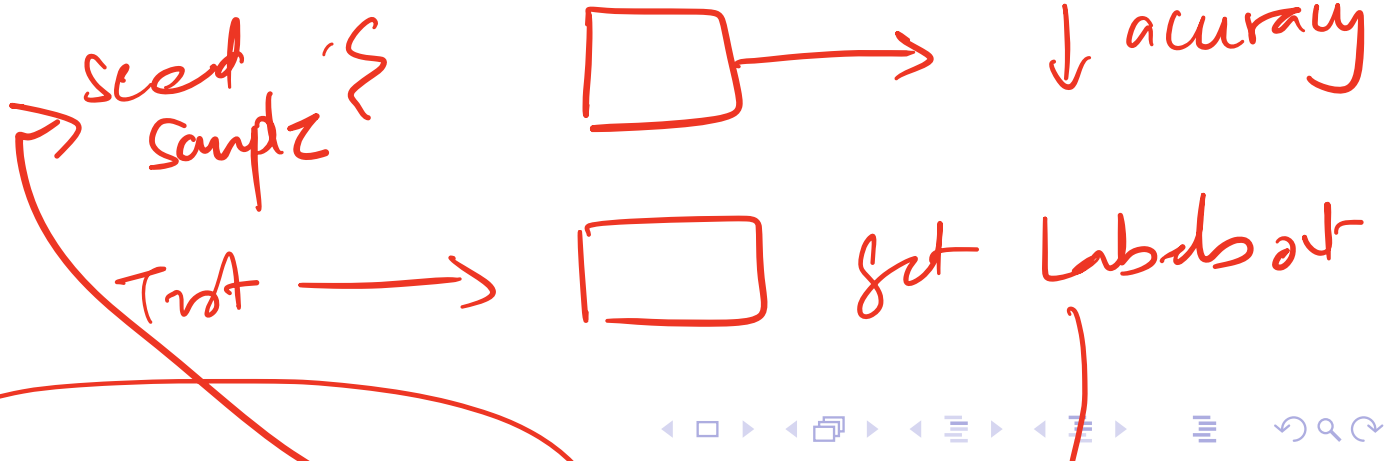
Establishing a baseline: Human Level Performance

First task in ML System design.

- Human level performance (HLP) is the performance of a human on a given task.
- HLP is often used as a baseline for evaluating the performance of ML models.
- Example: Labelers on 4 of 6 examples and disagreed on 2 out of 6. And so Human Level Performance is 66.7%.
- HLP is also useful for understanding the limitations of ML models.

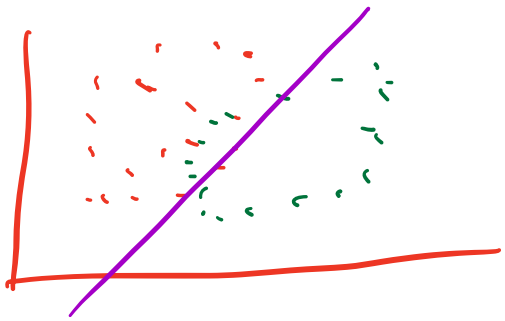
GT L is give

Data Leakage



Techniques for Lack of Labeled Data

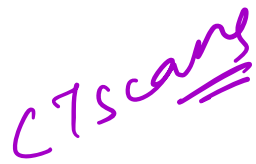
- Weak Supervision: Leverages (often noisy) heuristics to generate labels; e.g., Snorkel. No/Some GT labels needed.
- Transfer Learning: Leverages knowledge from a related task with abundant labeled data. No GT labels needed.
- Data Augmentation: Creates new training examples by modifying existing ones.
- Semi-supervised Learning: Leverages structural assumptions to generate labels for unlabeled data. Yes GT labels needed.
- Active Learning: Labels data samples that are most useful to your model. Need GT labels.



Weak Supervision

- Weak supervision is a way to reduce the dependence on labeled data.
- It uses (often noisy) heuristics to generate labels.
- The heuristics can be based on domain knowledge, patterns in the data, or other sources of information.
- The generated labels are often noisy and may not be accurate.
- The model is trained on the noisy labels and learns to generalize from them.
- Example: Snorkel is a popular weak supervision framework.

Snorkel: Motivation

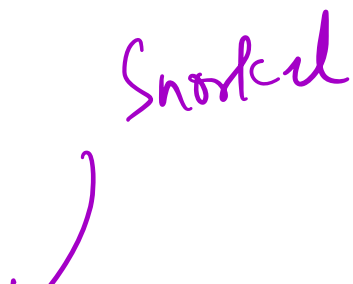




- Time to get training data approx. 8-9 months
- Time to train a model approx. 1-2 days to atmost a week.
- Slow: Time required scales linearly with number of labels needed as long as no re-labeling
- Non-adaptive: Every change requires re-labeling the dataset

Snorkel: key idea

- Can we generate labels programmatically?
- Can we generate more labels as needed?
- Labeling functions (LFs): Encode SME heuristics as functions and use them to label training data programmatically



Snorkel: example

Each LF is a rule, heuristic, or weak signal.

- Task: classify a tweet.
- example LFs:
  - LF_1: If text contains "love" $\rightarrow$ positive
  - LF_2: If text contains "hate" $\rightarrow$ negative
  - LF_3: If user clicked "like" $\rightarrow$ positive
- Each LF outputs: a label, or may abstain.

Snorkel: challenges

- LFs are noisy and may conflict
- LF_1 and LF_2 may both fire on the same example: Tweet: "I love how much I hate Mondays"
- LF_3 may be unreliable
- How to combine LF outputs to produce a single label?

LF_1 ✓ inaccurate

LF_2 ✓

LF_3 incorrect

Issues with Majority Vote

If you just take a majority vote, you're assuming all LFs are equally good and independent — which is rarely true.

$P(Y|X)$

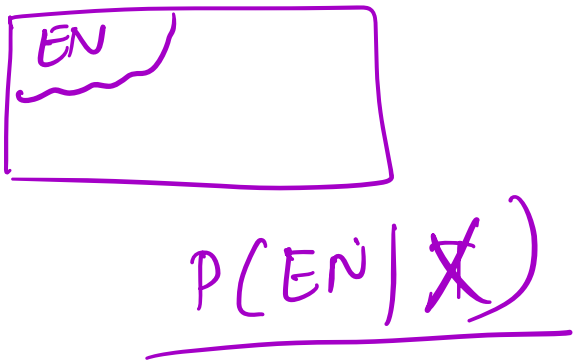
Snorkel: generative model

- Snorkel learns a generative model to estimate LF accuracies and correlations
- Uses these estimates to combine LF outputs into probabilistic labels
- Can also learn to abstain when LFs are uncertain
- LF_1 is 90% accurate, LF_2 is 80% accurate, LF_3 is 60% accurate
- On an example where LF_1 and LF_2 fire, but LF_3 abstains:
  - Majority vote: 50% positive, 50% negative
  - Snorkel: 69% positive, 31% negative (weighted by accuracies)

Discriminative Vs Generative Models

- Discriminative Goal: Directly predict the label from observed features.
- Learns: The conditional probability P(Y—X), without caring how X was generated.
- Usually higher predictive performance, captures complex patterns in raw features.
- Example: Logistic regression, SVM, neural networks.





Discriminative Vs Generative Models

- Discriminative Goal: Directly predict the label from observed features.
- Learns: The conditional probability P(Y—X), without caring how X was generated.
- Usually higher predictive performance, captures complex patterns in raw features.
- Example: Logistic regression, SVM, neural networks.
- Generative Goal: Model how the data was generated.
- Learns: The joint probability P(X,Y), which can be factored into P(X—Y)P(Y), where X is the observed data and Y is the label.
- Can generate new data points, handles missing data well.
- Example: Naive Bayes, Gaussian Mixture Models, Hidden Markov Models.

Snorkel

- Creates a generative probabilistic model is a model that assumes there is some hidden (latent) structure behind your data.
- Specifies a probability distribution over both:
  - The hidden variables (things you don't observe directly, like the true label of a data point).
  - The observed variables (things you do observe, like the outputs of labeling functions).
- "Generates" the observed data by sampling from this hidden structure.
- Formally:
  P(observed, data,hidden variables) = P(hidden variables)
  P(observed data — hidden variables)

Snorkel results

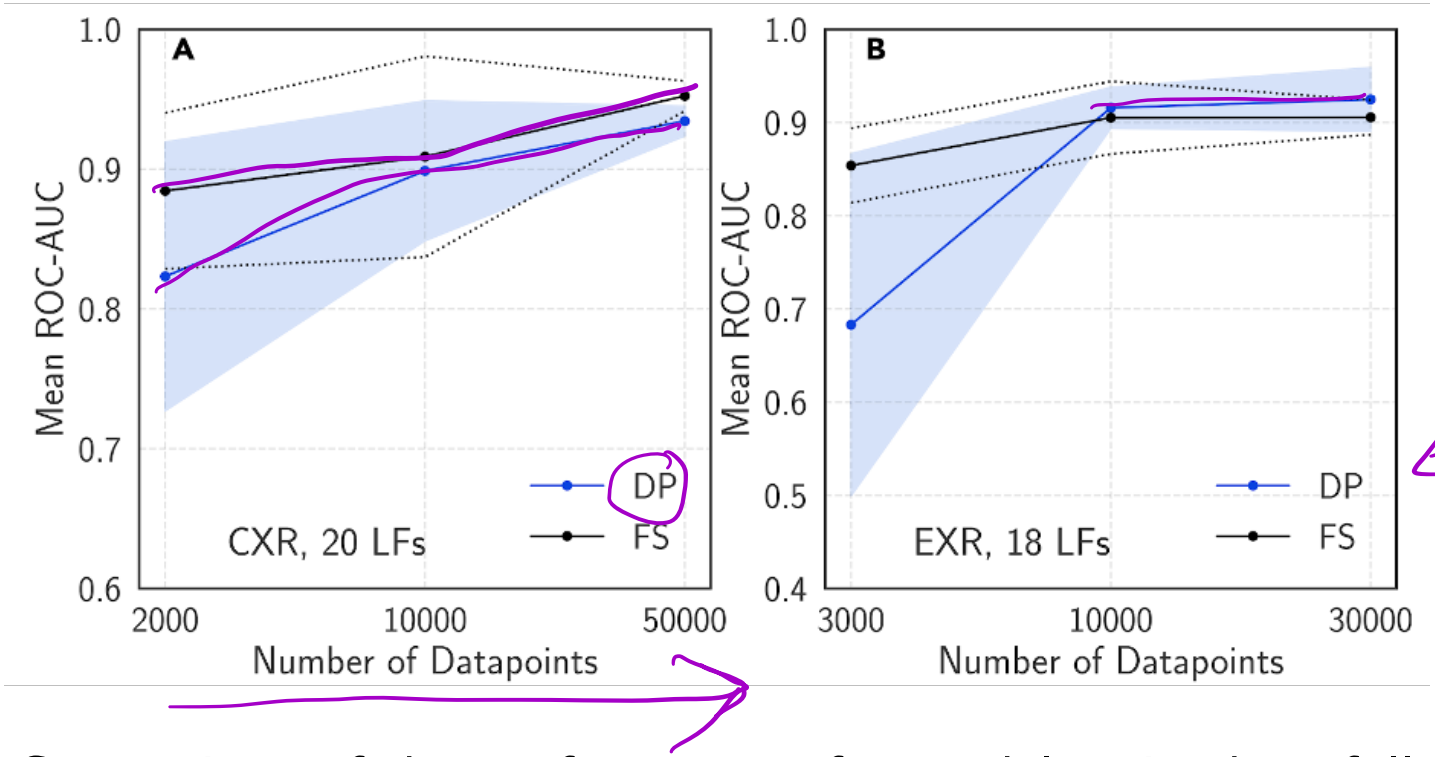


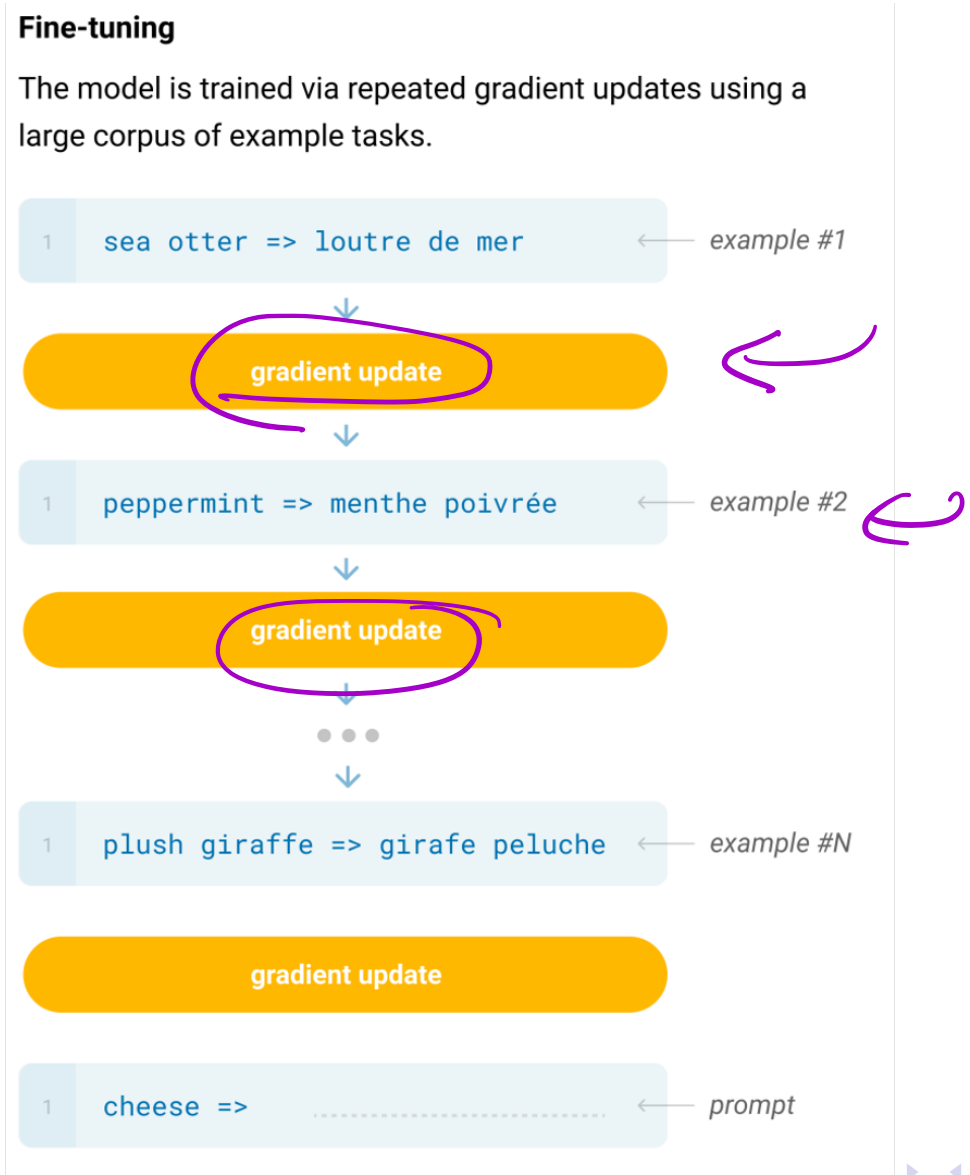
Figure: Comparison of the performance of a model trained on fully supervised labels (FS) and a model trained with programmatic labels (DP) on CXR and EXR tasks. Image by Dunnmon et al.

Transfer learning

- Transfer learning is a way to reduce the dependence on labeled data.
- It leverages knowledge from a related task with abundant labeled data.
- The model is pre-trained on the related task and then fine-tuned on the target task with limited labeled data.
- The pre-trained model has learned useful representations of the input data that can be transferred to the target task.
- Example: BERT, GPT, RoBERTa, etc.
- Alternatively, model is prompted to perform the target task.

Transfer Learning: finetuning

- Finetuning: Pretrain a model on a large dataset, then update all or some of its weights using labeled data from the target task.



Transfer Learning: prompting

- Prompting: Use a pre-trained model as is, and design input prompts to elicit desired outputs for the target task.

Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  cheese =>                           ← prompt
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← example
3  cheese =>                           ← prompt
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

```
1  Translate English to French:        ← task description
2  sea otter => loutre de mer          ← examples
```

Active Learning

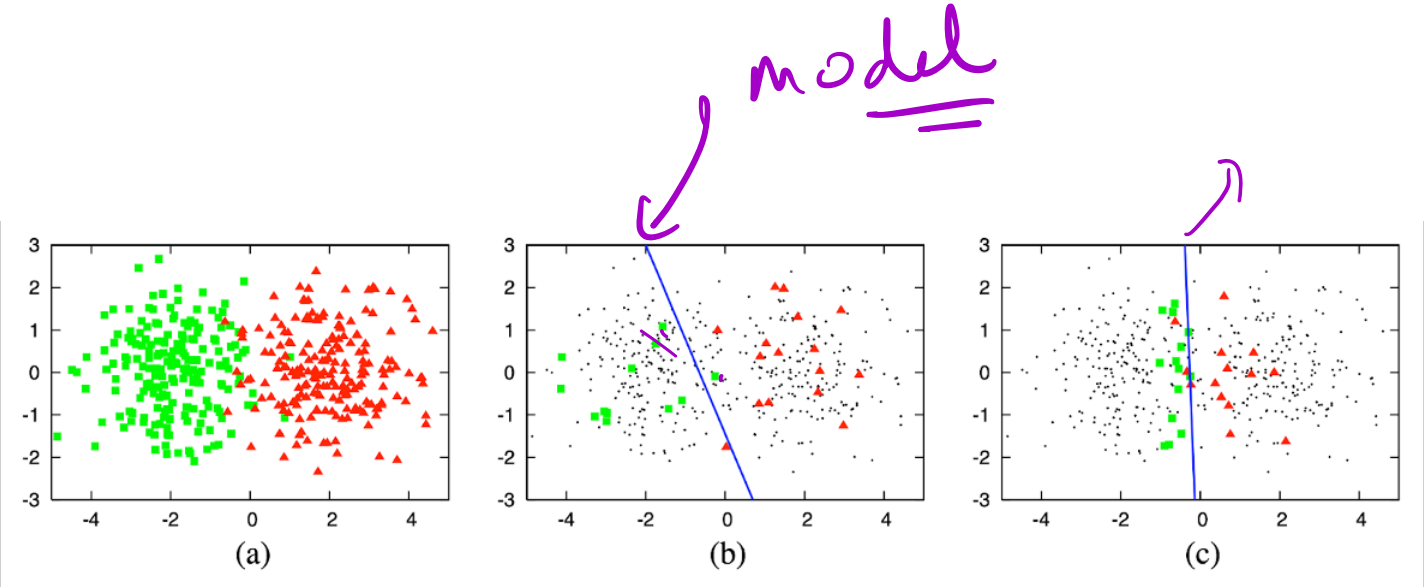


Figure: How uncertainty-based active learning works. (a) A toy data set of 400 instances, evenly sampled from two class Gaussians. (b) A model trained on 30 examples randomly labeled gives an accuracy of 70%. (c) A model trained on 30 examples chosen by active learning gives an accuracy of 90%. Image by Burr Settles.