DATA304 Big Data Analysis

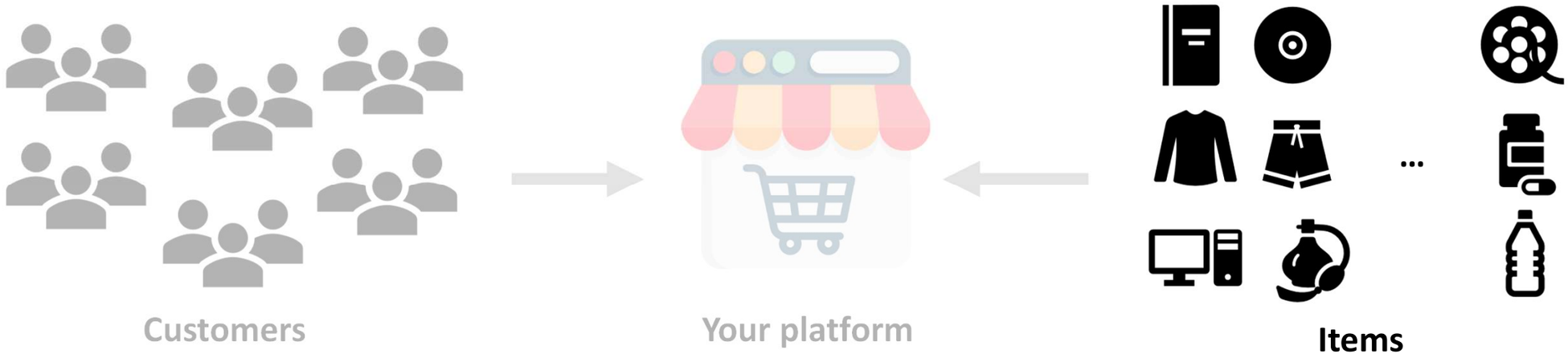
6. Text Classification 2:
Learning with Limited Labels 1

SeongKu Kang

Korea University

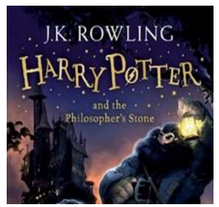
https://www.idea.korea.ac.kr/

We are opening an e-commerce platform!



We have a wide variety of items, with new arrivals every day!

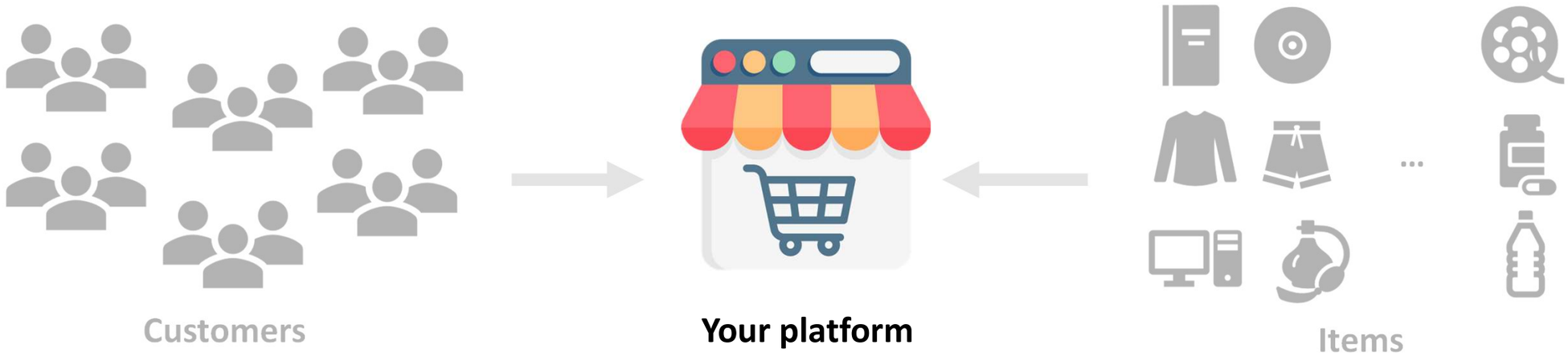
Nike Air Force is a range of athletic shoes made by Nike. It was created by designer Bruce Kilgore and was the first basketball shoe …



Harry Potter and the Philosopher's Stone is a fantasy novel written by British author J. K. Rowling. Harry Potter lives with his abusive uncle and aunt, Vernon …

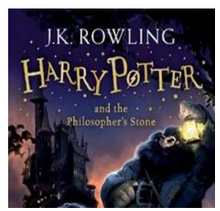
Lover is the seventh studio album by the American singer-songwriter Taylor Swift. Singles from Lover: "Me!", … ,"Lover", "The Man", "Cruel Summer " …

We are opening an e-commerce platform!



Your platform should be able to categorize a wide range of items (**classification**).

Clothing, Shoes & Jewelry
> Men > Shoes
> Fashion Sneakers



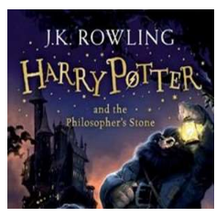
Books > Teen & Young Adult
> Literature & Fiction
> Action & Adventure
> Fantasy

CDs & Vinyl
> Pop > Vocal Pop

Text classification

- **Why Important?**
  - Core task to analyze your data: assign predefined labels (e.g., brand, sentiment, topic).
    - Widely used in search engines, recommender systems, spam detection, and more.
    - Forms the basis for many advanced applications such as QA, dialogue systems, and personalization.

Clothing, Shoes & Jewelry
> Men > Shoes
> Fashion Sneakers



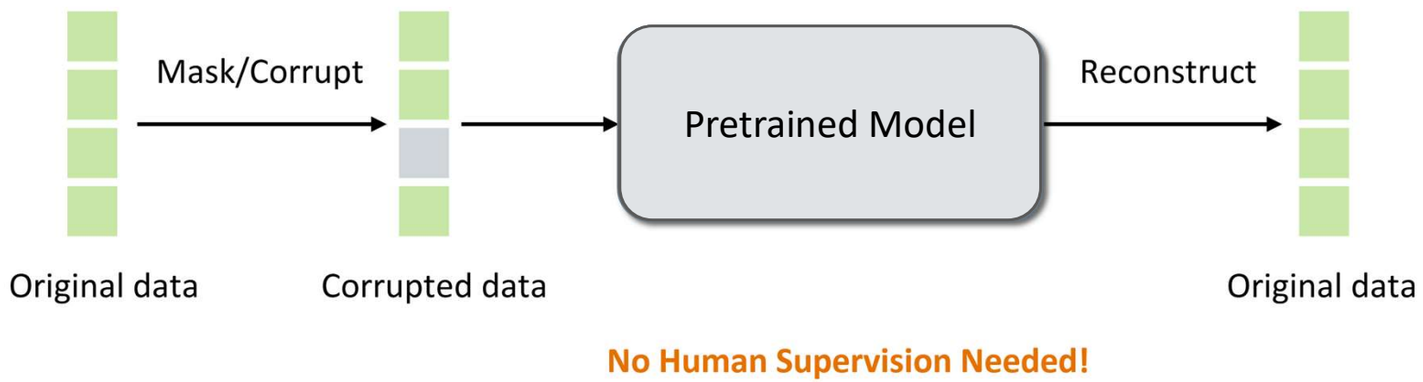
Books > Teen & Young Adult
> Literature & Fiction
> Action & Adventure
> Fantasy

CDs & Vinyl
> Pop > Vocal Pop

- **Our learning path**:
  - We have represented texts as vectors.
  - Next, we will conduct classification using these representations.
  - We will mainly follow the pretrain-and-fine-tune paradigm to adapt *general language knowledge* to *specific classification tasks*.

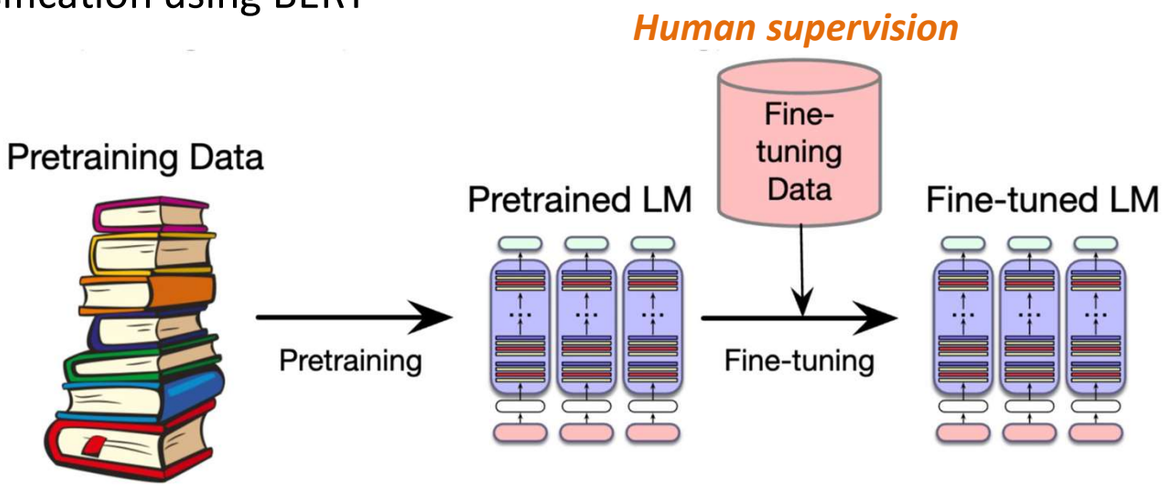
Recap: Pretraining + Fine-tuning

- **Pretraining**
  - **Motivation**: there are abundant text data on the web, with rich information of linguistic patterns and knowledge about the world
  - **Goal**: Learn general-purpose representations without human labeling
  - **Example**:
    - Word2Vec (predict neighboring words)
    - BERT (masked language modeling, next sentence prediction)



Recap: Pretraining + Fine-tuning

- **Fine-tuning**
  - **Motivation**: pretrained models capture general knowledge, but tasks like classification require task-specific knowledge
  - **Goal**: adjust the pretrained model parameters to perform well on a target (downstream) task
    - Requires *much less labeled data* than training from scratch
    - Leverages *knowledge learned during pretraining*
  - **Examples**:
    - Sentiment classification using BERT



Downstream task

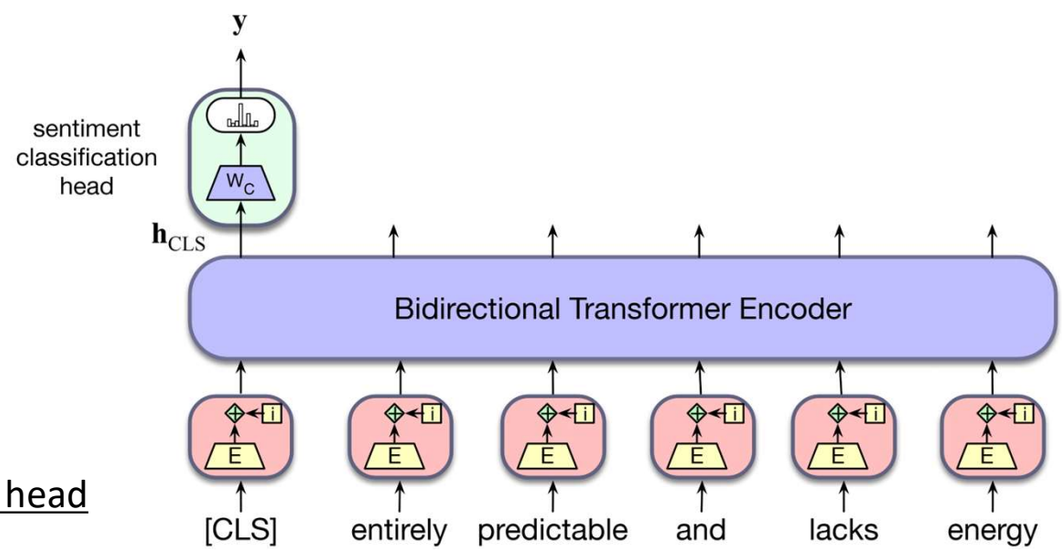
- Spam mail classification
- Sentiment classification
- Search
- Question and answering
- …

Recap: Pretraining + Fine-tuning

- **Downstream task**: sentiment classification
  - Predict whether a given sentence is positive, neutral, or negative

- **How it works**:
  - Given a pretrained BERT, we represents each sentence as a vector
    - Popular choice: CLS representation, average of all contextual embeddings
  - We add a small classification head (a small network)
  - The model is then fine-tuned with labeled data

- **Training choice**:
  A. Full fine-tuning: update all parameters
  B. Partial fine-tuning: update a subset of parameters
     - Freeze the encoder + update only the head
     - Freeze most of the encoder + update the top layers and the head



Recap: Classification task

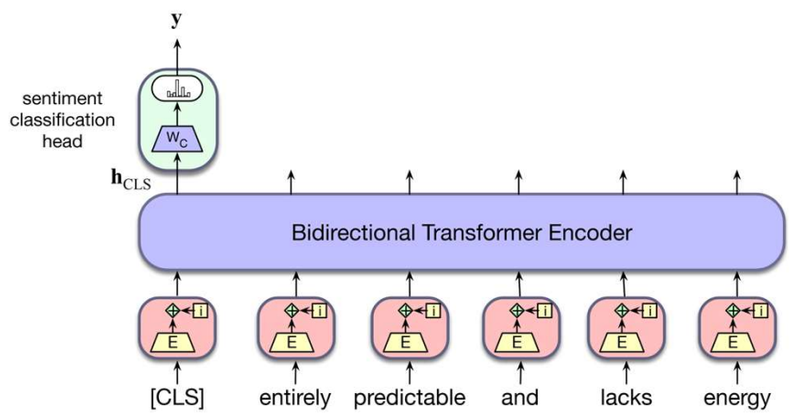
- **Classification type & loss**:
  - Binary classification
  - Multi-class classification
  - Multi-label classification

$$\mathcal{L}_{\text{BCE}}(\theta) = -\sum_{n=1}^{N}\Big[y_n \log \hat{y}_n + (1-y_n)\log(1-\hat{y}_n)\Big]$$

$$\mathcal{L}_{\text{CE}}(\theta) = -\sum_{n=1}^{N}\sum_{c=1}^{C}\mathbf{1}[y_n = c]\log \hat{y}_{n,c}$$

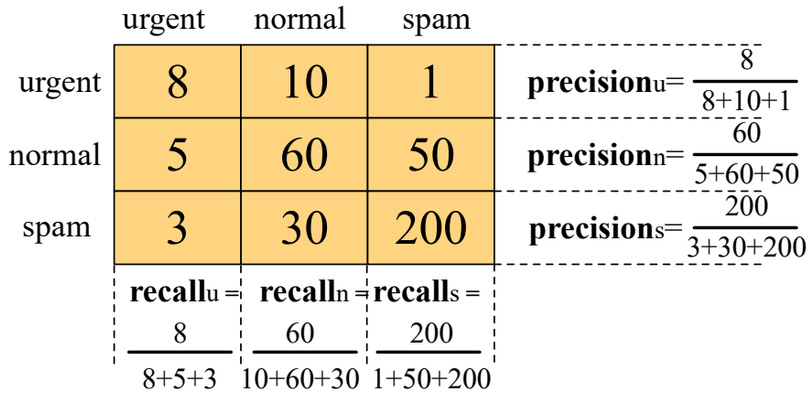
$$\mathcal{L}_{\text{Multi-label}}(\theta) = -\sum_{n=1}^{N}\sum_{c=1}^{C}\Big[y_{n,c}\log \hat{y}_{n,c} + (1-y_{n,c})\log(1-\hat{y}_{n,c})\Big]$$

- **Think about the whole picture!**
  - Each input text is represented as a vector using a pretrained model.
  - By minimizing the loss, you fine-tune the model (either full vs. partial depending on your choice)



Recap: Classification evaluation

- **Metrics**:
  - Accuracy: Overall fraction of correct predictions.
  - Precision: Among predicted positives, how many are correct.
  - Recall: Among actual positives, how many are found.
  - F1-score: Harmonic mean of Precision and Recall.



- **Averaging methods**:
  - Macro-averaging: Compute metrics per class, then average.
    - Treats all classes equally, regardless of size.
  - Micro-averaging: Aggregate TP, FP, FN across classes, then compute metrics.
    - Weighted by class frequency, favors larger classes.
  - Use Macro *when class imbalance is a concern*; Micro *when overall performance is more important*.

Course plan

Sparse representation

Dense static representation

Dense contextual representation

Representing texts as vectors

Classification task

Learning with limited labels

Text classification

Lexical retrieval

Dense retrieval

LLM-enhanced IR

Search

Node representation

Graph neural networks

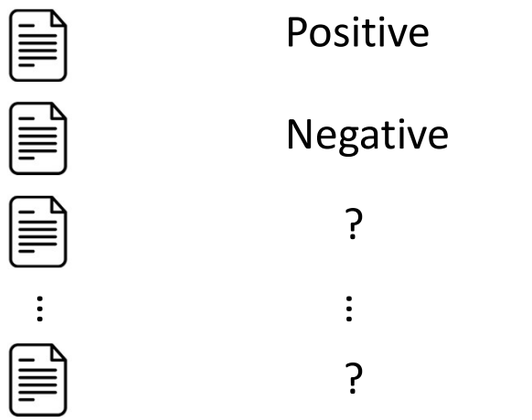
Leveraging relations

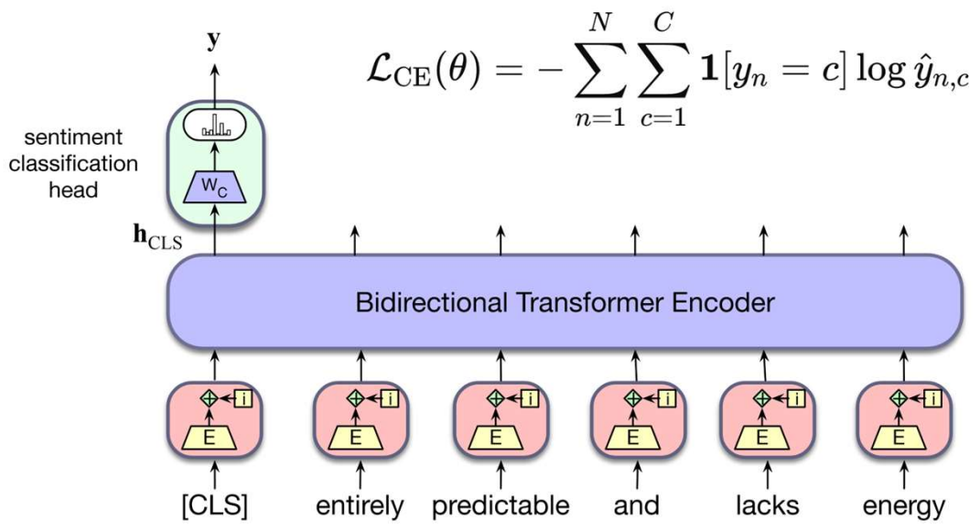
Text mining & Information retrieval

Learning with limited labels

- So far, we have discussed how to train classifiers using labeled data.
  - For every input text x, a corresponding class label y is provided.
- *But in reality, do we always have labels for all data instances?*

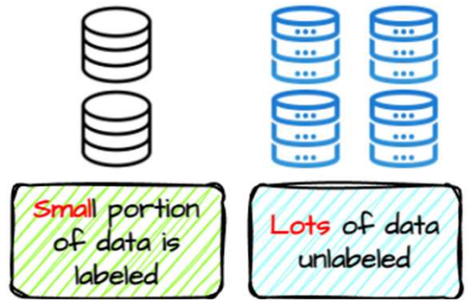
- We will explore how to deal with the scarcity of labeled data.



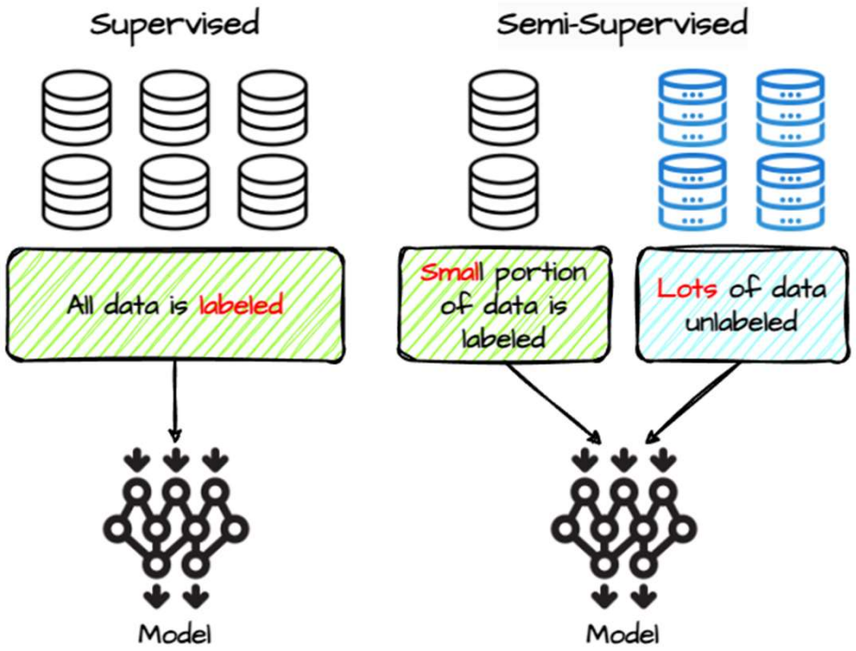


Learning with limited labels

- So far, we have discussed how to train classifiers using labeled data.
  - For every input text x, a corresponding class label y is provided.
- *But in reality, do we always have labels for all data instances?*



- We will explore how to deal with the scarcity of labeled data.
  1. **Semi-supervised learning**
     - *"How can we effectively leverage unlabeled data?"*
  2. **Multi-task learning**
     - *"If one task doesn't have enough labels, can we borrow signals from related tasks?"*
  3. **Adversarial learning**
     - *"What if labeled and unlabeled data come from different distributions?"*



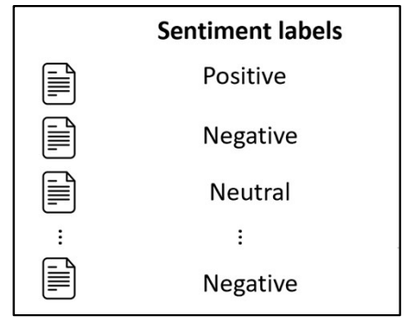
(Figure credits): many figures in this section are borrowed from Mr. Baixu Chen's slides and Dr. Kevin Clark's slides

Semi-supervised learning

Learning types according to label availability

- **Supervised learning**

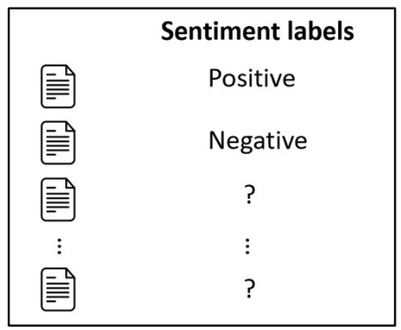
$$\mathcal{D}_l = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$



  - All training samples have ground-truth labels
  - Limitation: labeling requires human effort (often expert knowledge), making it infeasible to obtain very large labeled datasets.

- **Semi-supervised learning**

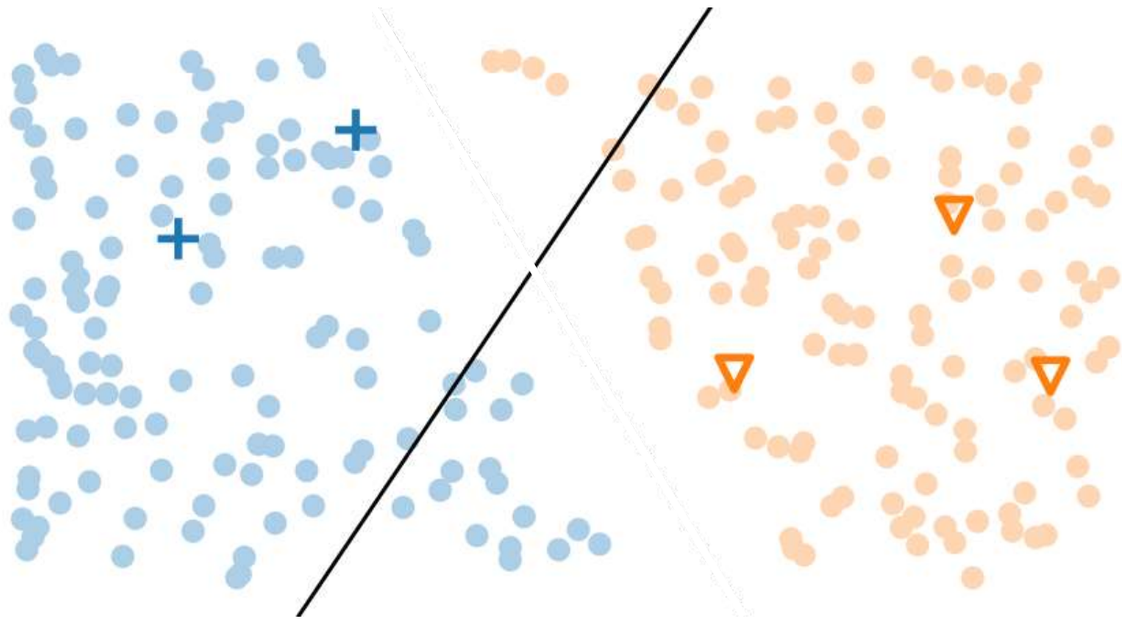
$$\mathcal{D}_l = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$$
$$\mathcal{D}_u = \{x_1, x_2, \dots, x_m\}$$



  - Only a small set of samples have ground-truth labels, i.e., $m \gg n$
  - Advantage: Collecting unlabeled samples is *much easier* compared to human-labeled data

Semi-Supervised Learning (SSL)

- **Goal**: Use both labeled and unlabeled data during training
  - Labeled data provides direct supervision
  - Unlabeled data helps the model generalize better when used with the right techniques



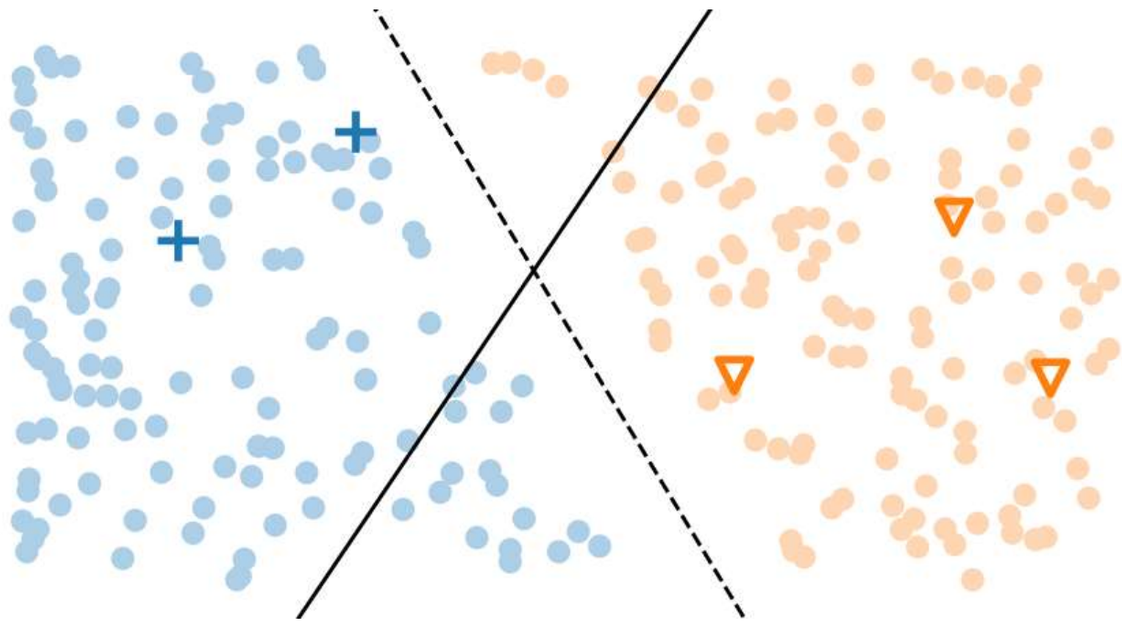
Blue, orange: class

+, ▽: labeled data

Decision boundary with supervised learning

Semi-Supervised Learning (SSL)

- **Goal**: Use both labeled and unlabeled data during training
  - Labeled data provides direct supervision
  - Unlabeled data helps the model generalize better when used with the right techniques



Decision boundary with supervised learning

Decision boundary with semi-supervised learning

Blue, orange: class

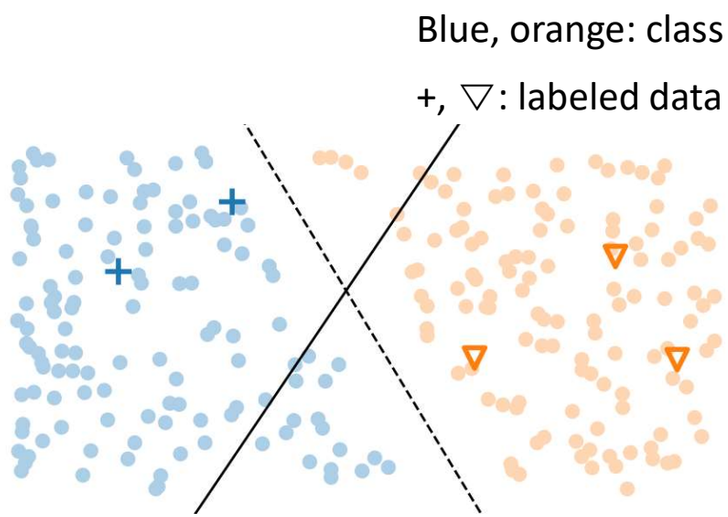
+, ▽: labeled data

Which line is a better decision boundary, and why?

Semi-Supervised Learning (SSL)

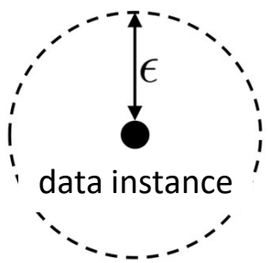
- **Goal**: Use both labeled and unlabeled data during training
  - Labeled data provides direct supervision
  - Unlabeled data helps the model generalize better when used with the right techniques

- **Key assumptions**:
  1. Smoothness assumption
     - Nearby data points in the input space should have the same label
     - *"If they look similar, they are similar"*
  2. Low-density assumption
     - A good decision boundary between classes should avoid regions with many data points.
     - *"Boundaries should pass through low-density areas"*



Semi-Supervised Learning (SSL)

- Two main (and foundational) approaches we will cover:
  1. **Pseudo labeling**
     - Let the model *assign labels to unlabeled data* using its *confident predictions*.
     - Confident predictions usually match what the model has already learned from similar labeled examples
  2. **Consistency regularization**
     - For *the same input with small changes*, the model should make *consistent prediction*.
       - E.g., Adding small noise to vectors, or a minor word change.
     - Encourages decision boundaries to lie in low-density regions.



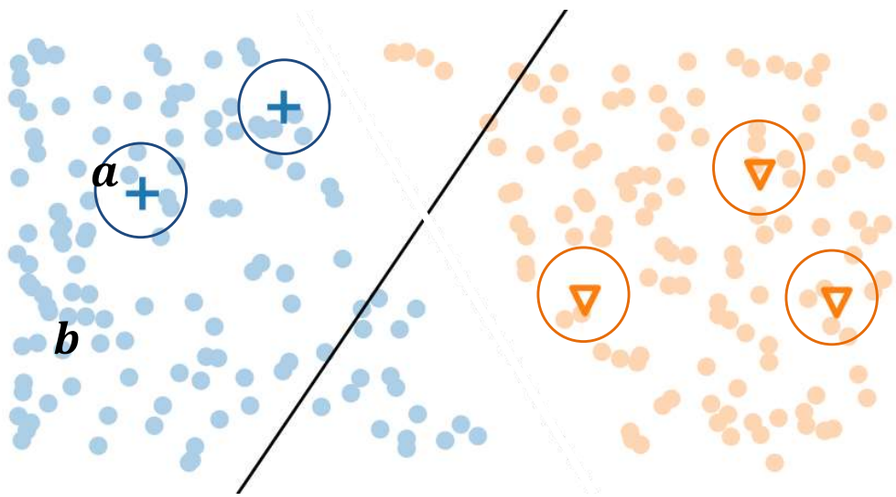
The model should make the same prediction for any point in the circle

1. Pseudo labeling

- **Key idea**: let the model *assign labels to unlabeled data* using its *confident predictions*.
- During training with labeled data, the model becomes confident about certain unlabeled samples.
  - This confidence comes from having already seen similar labeled examples.
  - Such predictions can be reused as “pseudo-labels” for training.

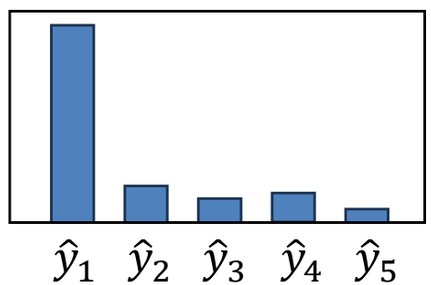
Blue, orange: class

+, ▽: labeled data

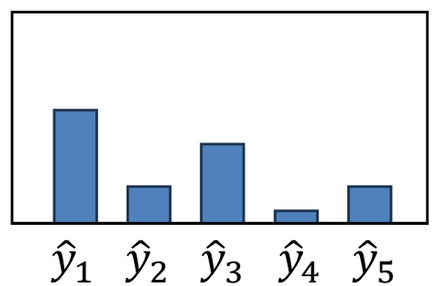


Softmax output from the model

Unlabeled data a



Unlabeled data b

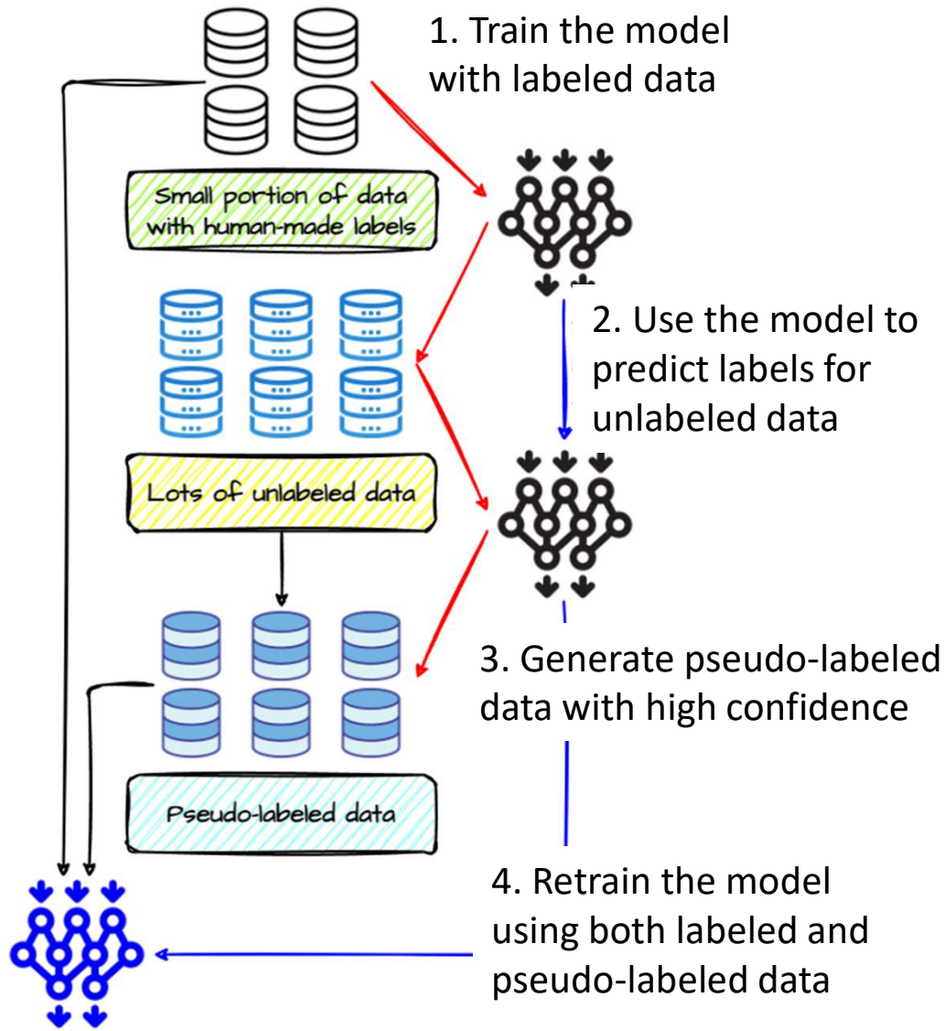


Between the two softmax outputs,
which one would you trust more, and why?

1. Pseudo labeling: process

(Figure credit): Deep learning bible

- **Step 1: Train with labeled data**
  - Start with a small portion of labeled data
  - Train the model in a supervised way
- **Step 2: Predict labels for unlabeled data**
  - Apply the trained model to unlabeled data
  - Obtain predictions with varying confidence levels (max prob.)
- **Step 3: Generate pseudo-labeled data**
  - Keep only high-confidence predictions as pseudo-labels
  - Treat them as if they were true labeled samples
- **Step 4: Retrain the model**



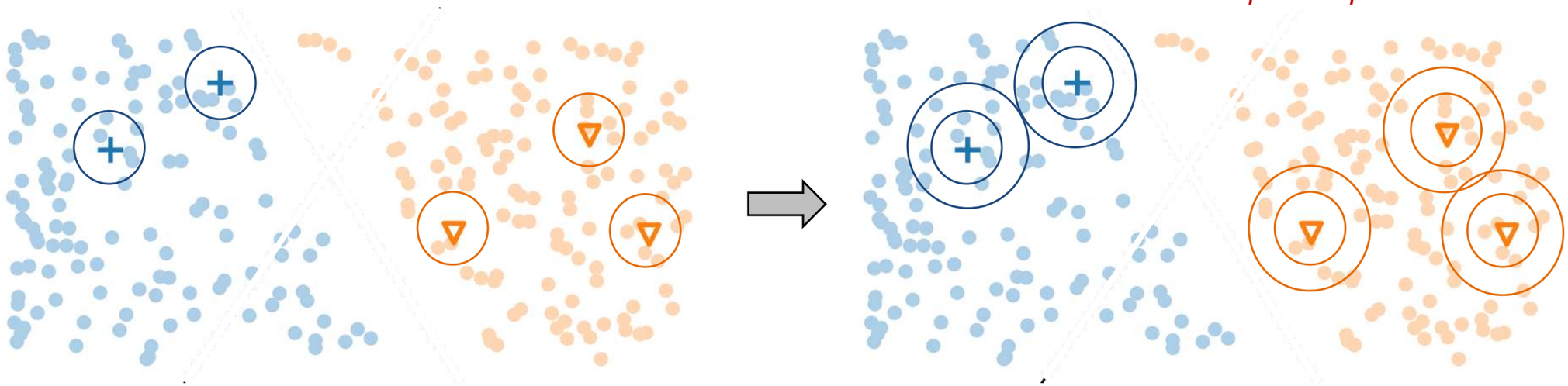
1. Pseudo labeling: self-training

- By learning with pseudo-labels, the model gradually gains more knowledge about unlabeled data.
- **Self-training**: Repeat the pseudo-labeling process to propagate this knowledge and expand supervision.
  - Self-training == Iterative pseudo-labeling

Blue, orange: class

+, ▽: labeled data

more pseudo-labeled samples
expand supervision



One step of pseudo labeling

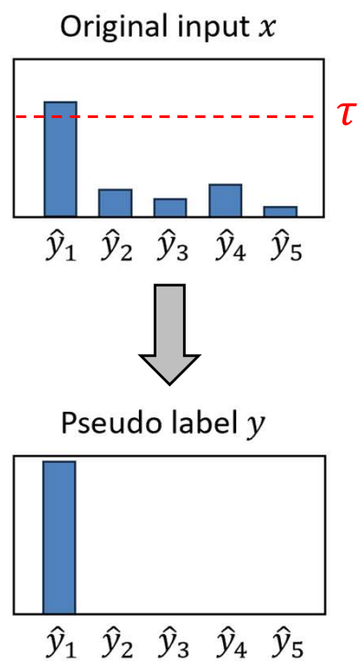
Two steps of pseudo labeling

1. Pseudo labeling: self-training

- By learning with pseudo-labels, the model gradually gains more knowledge about unlabeled data.
- **Self-training**: Repeat the pseudo-labeling process to propagate this knowledge and expand supervision.
  - Self-training == Iterative pseudo-labeling

Algorithm 1 Self-training with Pseudo-Labeling

Input: Labeled dataset $\mathcal{D}_l$, Unlabeled dataset $\mathcal{D}_u$, Confidence threshold τ
Output: Trained model f
Train base model f on $\mathcal{D}_l$
while not converged **do**
  Predict pseudo-labels for samples in $\mathcal{D}_u$ using f
  Select high-confidence predictions: $\mathcal{D}_p = \{(x, \hat{y}) \mid \max f(x) \geq \tau\}$
  Update training set: $\mathcal{D}_l \leftarrow \mathcal{D}_l \cup \mathcal{D}_p$
  Update f on the expanded $\mathcal{D}_l$
end while
return f



1. Pseudo labeling: summary

(Figure credit) IceBerg: Debiased Self-Training for Class-Imbalanced Node Classification, WWW'25

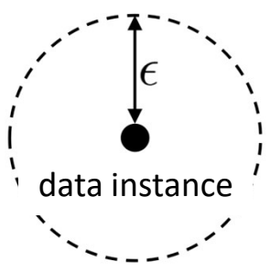
- Pseudo labeling is a simple and effective way to leverage unlabeled data.
- However, model predictions are not as accurate as human annotations.
  - In the worst case, pseudo-labeling can even perform worse than using only labeled data.

- With limited labels for minority class (orange), the model struggles to make accurate predictions across the whole space.
- Pseudo labels may only capture partial patterns of the unlabeled data.

- **Limitations**:
  1. **Error propagation**: Wrong pseudo-labels reinforce mistakes in training
  2. **Overconfidence**: The model can become overly confident, even in its wrong predictions

Semi-Supervised Learning (SSL)

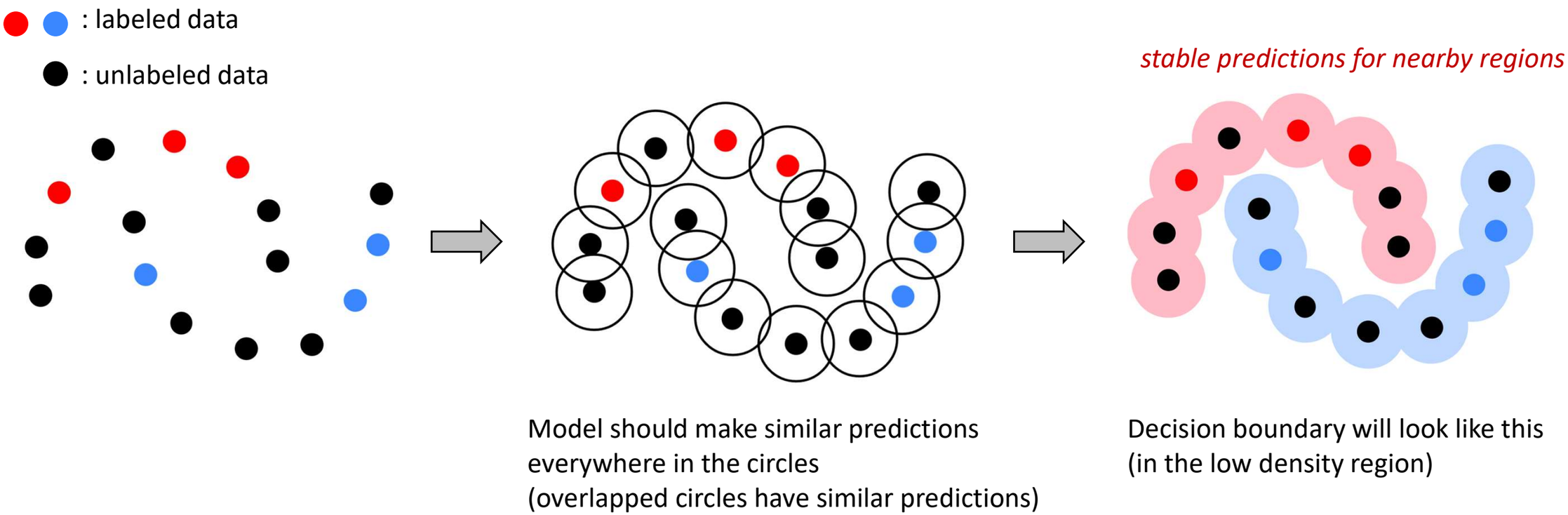
- Two main (and foundational) approaches we will cover:
  1. **Pseudo labeling**
     - Let the model *assign labels to unlabeled data* using its *confident predictions*.
     - Confident predictions usually match what the model has already learned from similar labeled examples
  2. **Consistency regularization**
     - For *the same input with small changes*, the model should make *consistent prediction*.
       - E.g., Adding small noise to vectors, or a minor word change.
     - Encourages decision boundaries to lie in low-density regions.



The model should make the same prediction for any point in the circle

2. Consistency regularization

- **Key idea**: for *the same input with small changes*, the model should make *consistent prediction*.
  - Add small perturbations (e.g., noise, augmentation) → prediction should remain stable
  - Equivalently, nearby data points should get similar predictions



2. Consistency regularization

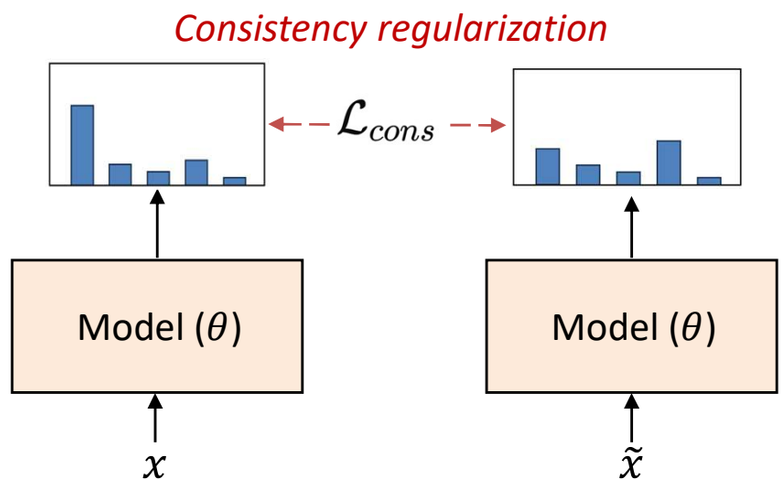
- **Key idea**: for *the same input with small changes*, the model should make *consistent prediction*.
  - Add small perturbations (e.g., noise, augmentation) → prediction should remain stable
- **Instantiation**:

$$\mathcal{L} = \mathcal{L}_{sup}(D_l) + \lambda \mathcal{L}_{cons}(D_u)$$

$$\mathcal{L}_{cons} = \mathbb{E}_{x \in D_l \cup D_u} \left[\|f(x;\theta) - f(\tilde{x};\theta)\|^2 \right]$$

x: original input, $\tilde{x}$: perturbed input

- **How to add perturbations?**
  - Add small random noise to the embeddings
  - Word dropout or masking
  - Data augmentation (e.g., rotation, crop in vision)
  - … and many other ways



2. Consistency regularization: insights

✓ Regularization prevents overfitting by discouraging models from memorizing data

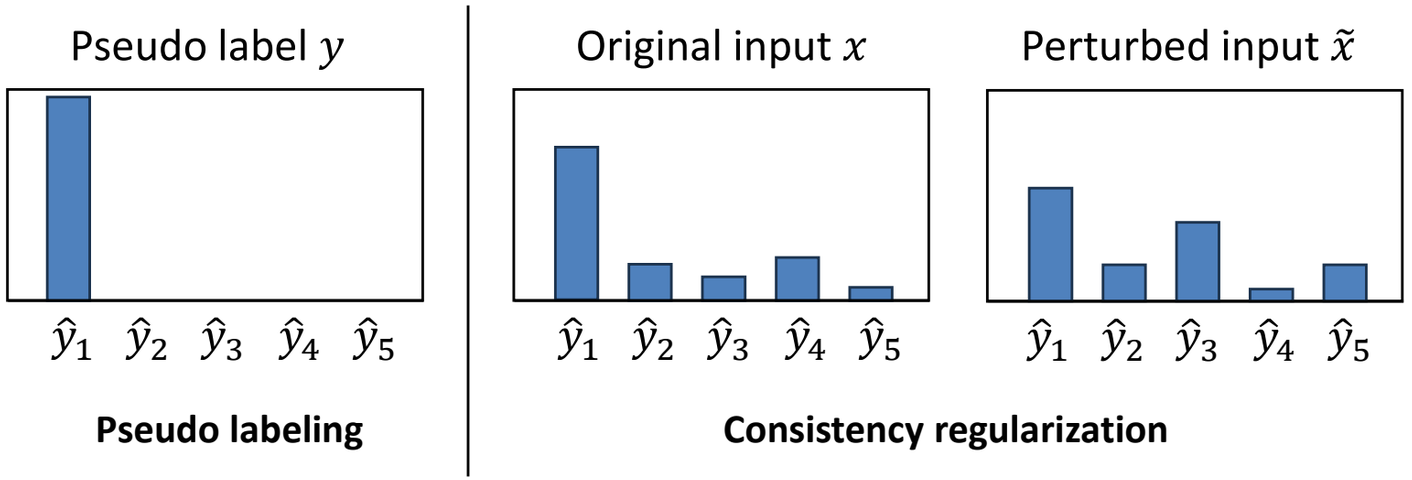
- **Instantiation**:

$$\mathcal{L} = \mathcal{L}_{sup}(D_l) + \lambda \mathcal{L}_{cons}(D_u)$$

$$\mathcal{L}_{cons} = \mathbb{E}_{x \in D_l \cup D_u} \left[\| f(x;\theta) - f(\tilde{x};\theta) \|^2 \right]$$

x: original input, $\tilde{x}$: perturbed input

- **Insight 1**: it forces the *predicted probability distributions* to be similar for original and perturbed inputs.



- The model tends to be naturally less confident on perturbed input
- As a result, the model has lower confidence overall

✓ **Regularization effect**: prevents the model from becoming overconfident, which happens when it memorizes the data (overfitting)

2. Consistency regularization: insights

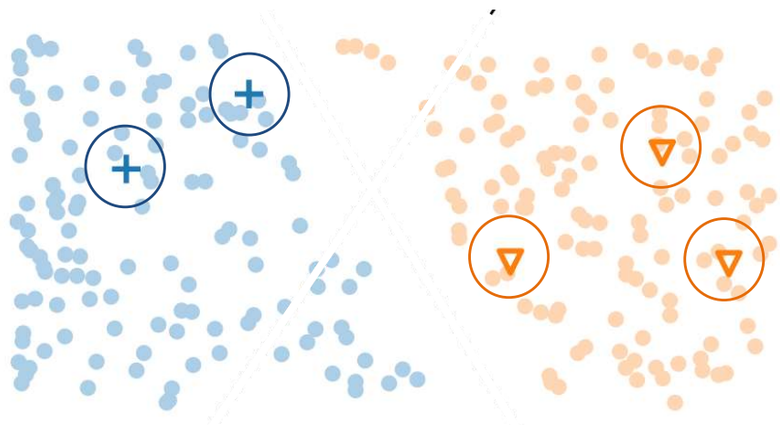
- **Instantiation**:

$$\mathcal{L} = \mathcal{L}_{sup}(D_l) + \lambda \mathcal{L}_{cons}(D_u)$$

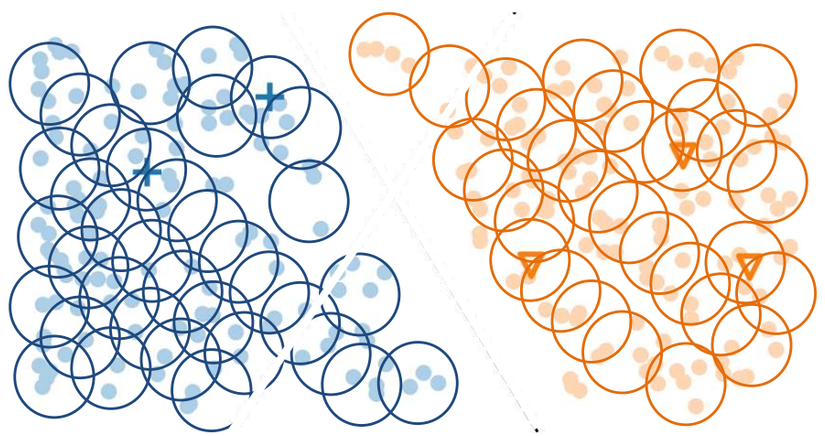
$$\mathcal{L}_{cons} = \mathbb{E}_{x \in D_l \cup D_u} \left[\| f(x;\theta) - f(\tilde{x};\theta) \|^2 \right]$$

x: original input, $\tilde{x}$: perturbed input

- **Insight 2**: unlike pseudo labeling that relies only on *high-confidence samples*, it uses *all unlabeled data*.
  - This allows leveraging much larger portions of the dataset.



Pseudo labeling



Consistency regularization

+, ▽: labeled data

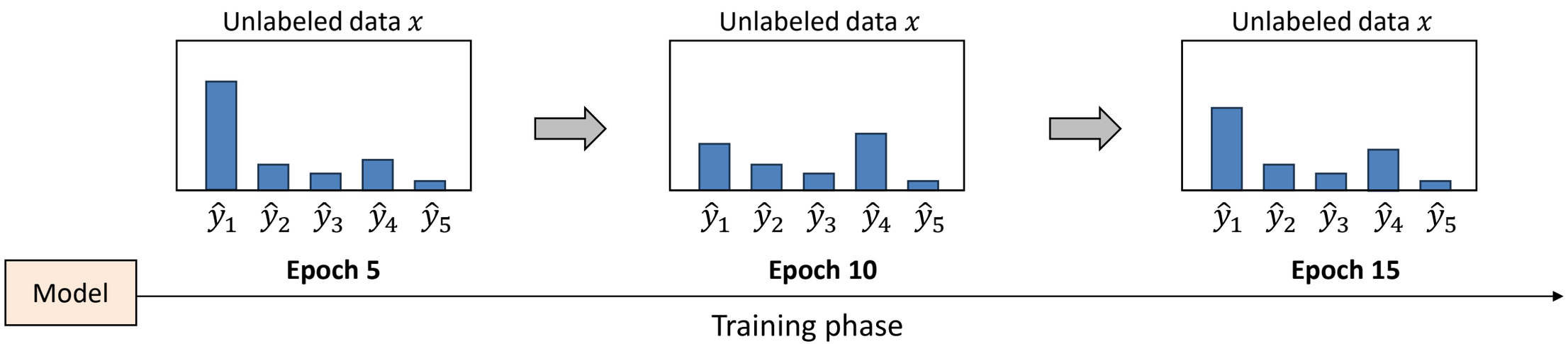
2. Consistency regularization: temporal ensemble

- **One practical issue**: during training, the model often makes unstable prediction for unlabeled data.
  - Predictions fluctuate across epochs, even without perturbations
  - With perturbations, instability becomes even worse → This makes whole training unstable!

$$\mathcal{L}_{cons} = \mathbb{E}_{x \in D_l \cup D_u} \left[\| f(x;\theta) - f(\tilde{x};\theta) \|^2 \right]$$

For unlabeled data, we don't have gold labels.
Model prediction is often highly unstable!

[Softmax output from the model]

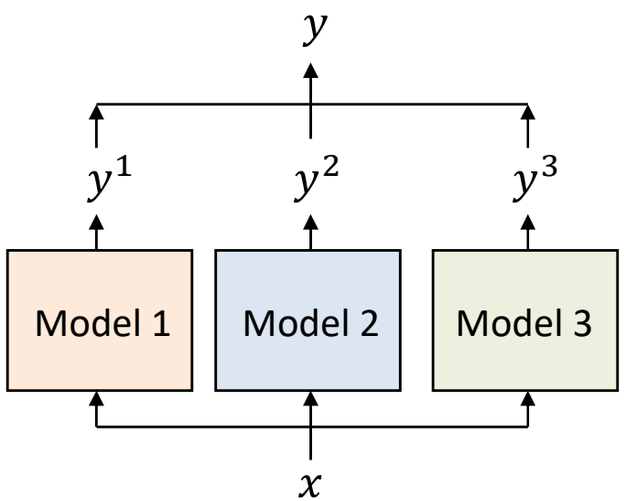


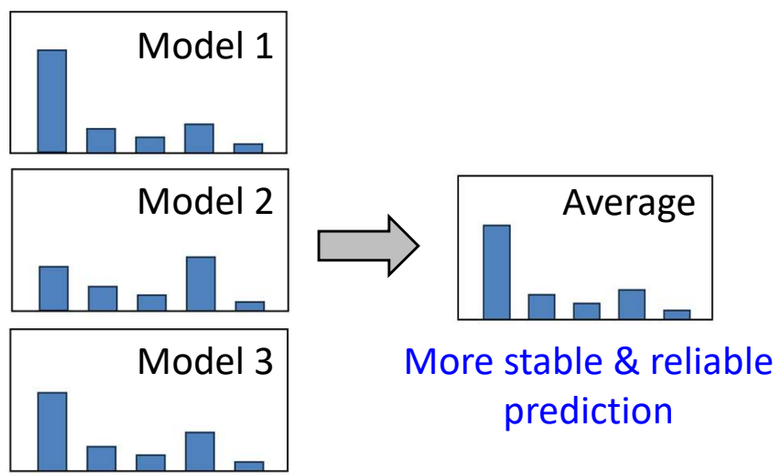
2. Consistency regularization: temporal ensemble

- How to stabilize the model prediction?

✓ **Model ensemble**

- Instead of using a single model, train multiple models and aggregate their predictions
- **Why?**
  - Different initializations → different converged parameters.
  - By averaging predictions, variance is reduced, making more stable & reliable prediction!





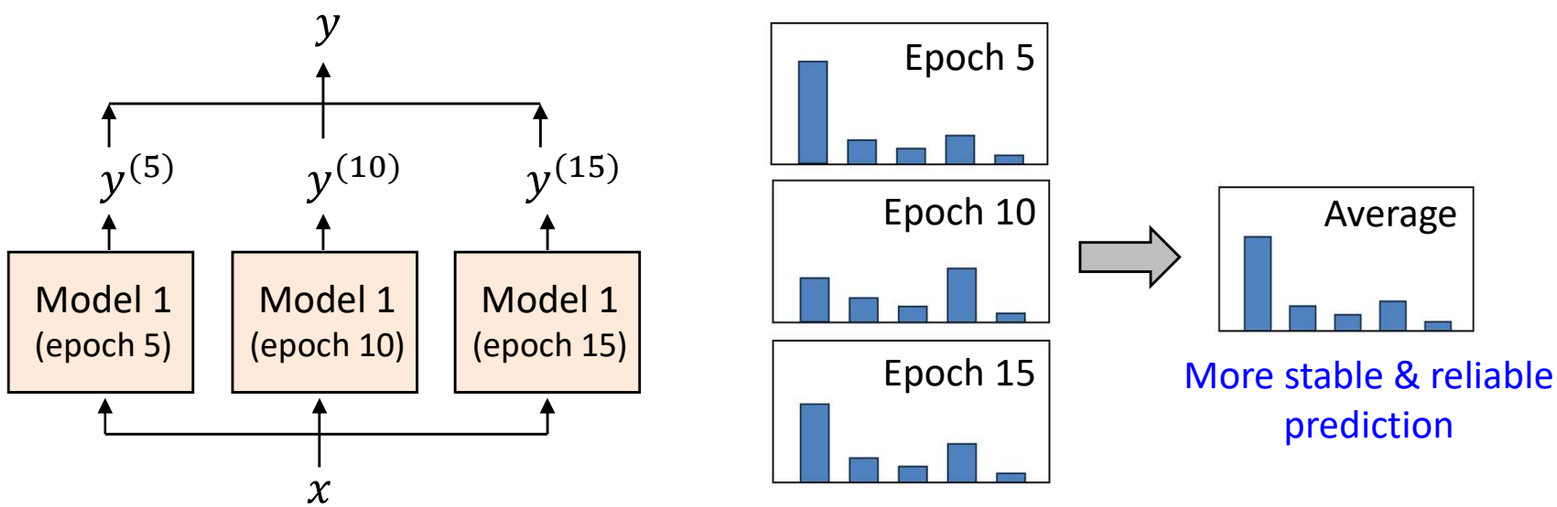
2. Consistency regularization: temporal ensemble

- How to stabilize the model prediction?

✓ **Temporal ensemble**

- Independently training multiple models is expensive. *Can we get the similar effect with just one model?*
- **Key idea:** Aggregate predictions of the same model over different training epochs

- Averaging over time reduces fluctuations. The prediction becomes more stable & reliable!



2. Consistency regularization: temporal ensemble

✓ **Temporal ensemble**

- **Key idea:** Aggregate predictions of the same data over different training epochs
- **Instantiation:** Conduct ensemble by exponential moving average (EMA) of model parameters

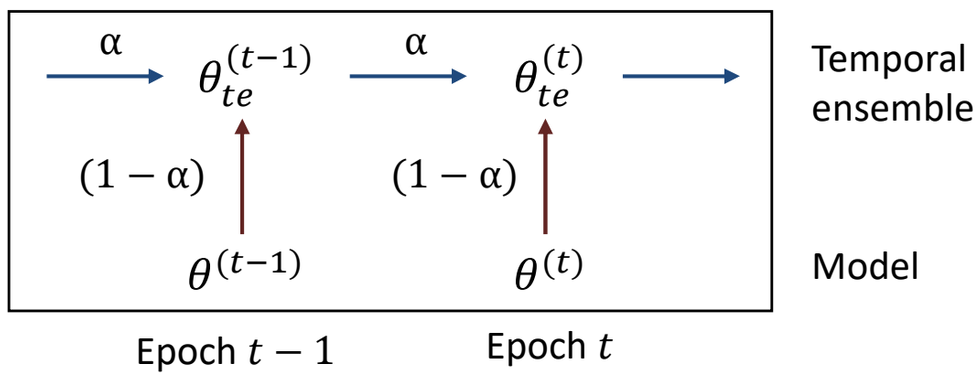
$$\theta_{te}^{(t)} \leftarrow \alpha\,\theta_{te}^{(t-1)} + (1-\alpha)\theta^{(t)}$$

$\theta^{(t)}$: model parameter at epoch t

$\theta_{te}^{(t)}$: temporal ensemble at epoch t

α: a hyperparameter that controls the degree of past knowledge

- Temporal ensemble aggregates past model parameters during training.
- It *follows the latest model slowly*, while remaining much more *stable by averaging predictions across epochs*.



Temporal ensemble update process

2. Consistency regularization: temporal ensemble

- **Consistency regularization with temporal ensemble**
  - We use $f(x;\theta_{te})$, the prediction from temporal ensemble
  - Provides a more reliable and stable guidance for unlabeled data

The original version

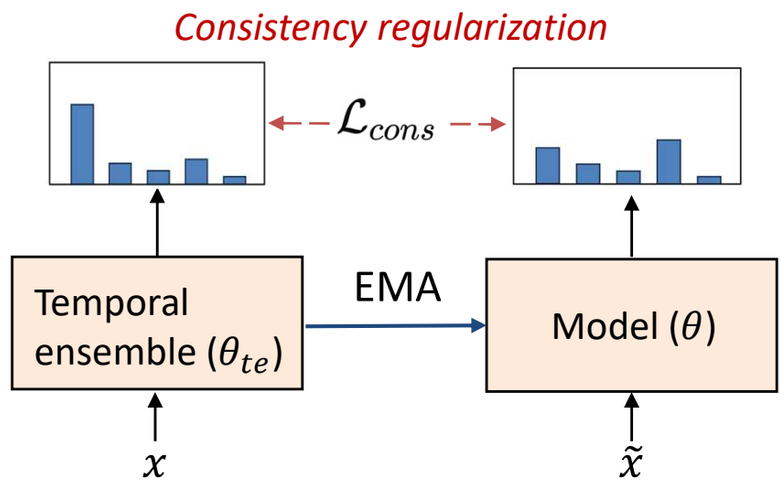
$$\mathcal{L}_{cons} = \mathbb{E}_{x \in D_l \cup D_u}\left[\|f(x;\theta) - f(\tilde{x};\theta)\|^2\right]$$

$$\mathcal{L} = \mathcal{L}_{sup}(D_l) + \lambda\,\mathcal{L}_{cons}(D_u)$$

$$\mathcal{L}_{cons} = \mathbb{E}_{x \in D_l \cup D_u}\left[\|f(x;\theta_{te}) - f(\tilde{x};\theta)\|^2\right]$$

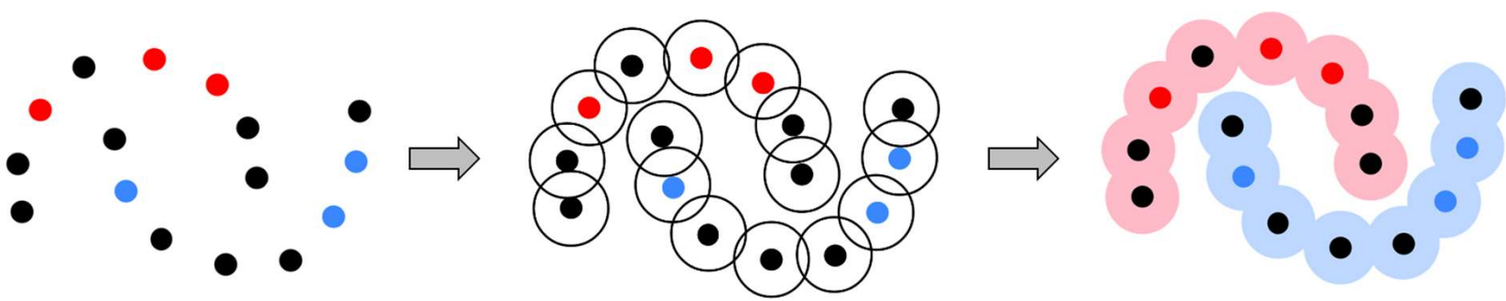
Algorithm 2 Consistency Regularization with Temporal Ensemble

Input: Labeled dataset D_l, Unlabeled dataset D_u
Output: Trained model $f(\cdot;\theta)$
Initialize parameters θ, ensemble parameters $\theta_{te} \leftarrow \theta$
for epoch $t = 1$ to T **do**
  Compute supervised loss on D_l: $\mathcal{L}_{sup}$
  Compute consistency loss on D_u: $\mathcal{L}_{cons} = \|f(x_u;\theta_{te}) - f(\tilde{x}_u;\theta)\|^2$
  Update θ by minimizing $\mathcal{L} = \mathcal{L}_{sup} + \lambda\mathcal{L}_{cons}$
  Update ensemble parameters: $\theta_{te}^{(t)} \leftarrow \alpha\theta_{te}^{(t-1)} + (1-\alpha)\theta^{(t)}$
end for
Return: $f(\cdot;\theta)$



2. Consistency regularization: summary

- For the same input with small changes, the model should make consistent predictions!
- Pros:
  - **Effective use of unlabeled data**: requires only perturbations and applied to broad unlabeled data
  - **Simple concept**: easy to add as a regularizer to existing models
- Cons:
  - **Perturbation sensitivity**: effectiveness depends on the choice of noise/augmentation.
    - Researchers still actively study this topic.
  - **Weak direct guidance**: does not provide explicit labels, so boundaries may still be uncertain.

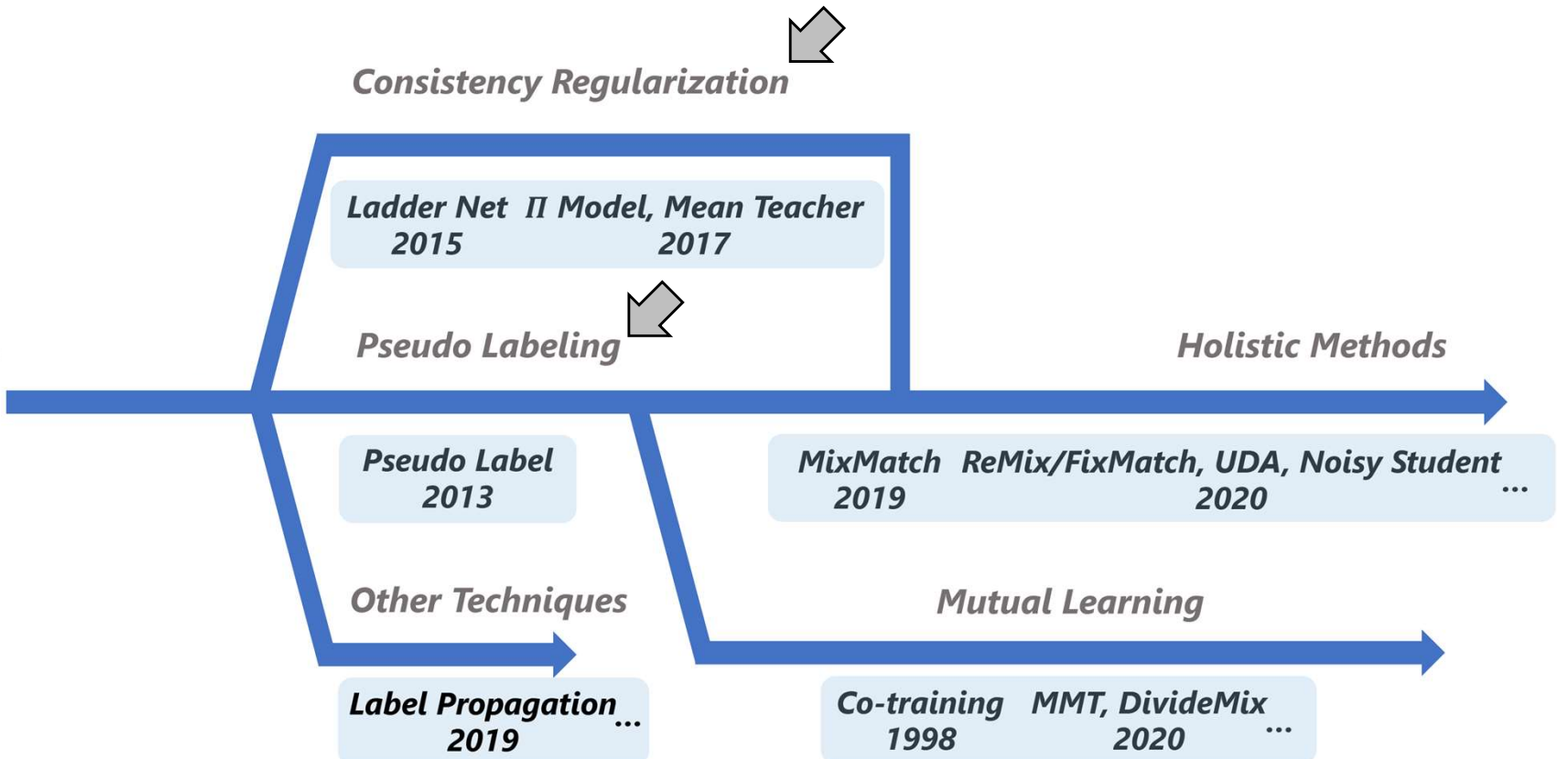


Semi-Supervised Learning (SSL): summary

- Two main (and foundational) approaches we covered:
  1. **Pseudo labeling**
     - Strength: Provides a direct training signal (like ground truth)
     - Limitation: Can amplify errors — if the model is overconfident in wrong predictions.
  2. **Consistency regularization**
     - Strength: Acts as a regularizer that prevents the model from becoming overconfident.
     - Limitation: Provides weak guidance since no explicit labels are given.
  - They have complementary relationship!
    - *Pseudo-labeling provides strong but noisy signals, while consistency regularization stabilizes them.*
  - *Almost all state-of-the-art SSL combines two approaches as a holistic framework!*

(Optional) Overview of SSL

- In this class, we covered up to Mean Teacher, published in 2017.
- Almost all subsequent work are based on two approaches we studied.



Overview of semi-supervised learning

Recommended readings

- **Articles**:
  - The Universe of AI – Semi-supervised learning
    - Korean: https://wikidocs.net/255169
    - English: https://wikidocs.net/255197

- **Papers**:
  - Consistency regularization with temporal ensemble
    - Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results, NeurIPS'17

  - (Purely optional, beyond the scope of this class)
    - Debiased Self-Training for Semi-Supervised Learning, NeurIPS'22