# Machine Learning (CE 40717)

## Fall 2024

Ali Sharifi-Zarchi

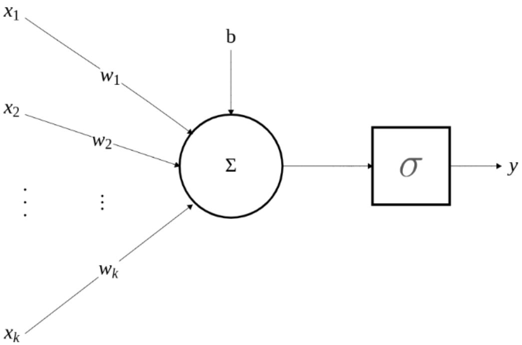
CE Department
Sharif University of Technology

December 29, 2024

# Self-Supervised Learning

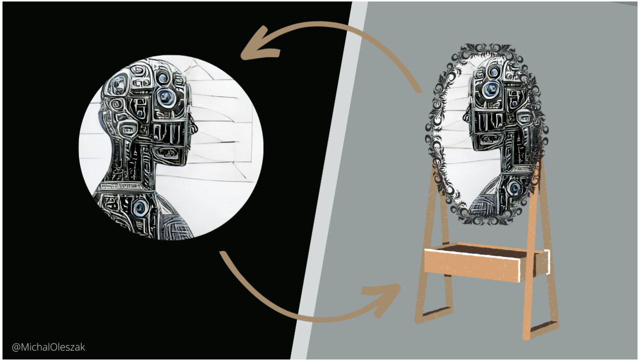
"the dark matter of intelligence" [1]

- $\{x_1, x_2, \ldots, x_k\}$ : input features
- $\{w_1, w_2, \ldots, w_k\}$ : feature weights
- $b$ : bias term
- $\sigma(\cdot)$ : activation function
- $y$ : output of the neuron




[1] https://ai.meta.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/

# Self-Supervised Learning




*"the dark matter of intelligence"*[2]

[2] https://ai.meta.com/blog/self-supervised-learning-the-dark-matter-of-intelligence/

## Why Neural Networks?

- Self-supervised learning defines a **pretext** task based on unlabeled inputs to produce descriptive and intelligible representations [Hastie et al., 2009, Goodfellow et al., 2016]
    - Learn with supervised learning objectives, e.g., classification, regression.
    - Labels of these pretext tasks are generated *automatically*
    - Can be used in other downstream tasks.

# Example Workflow

- Training objective: predicting the context surrounding a word
- encourages the model to capture relationships among words
- The same SSL model representations can be used across a range of downstream tasks. e.g.
    - translating text across languages
    - summarizing
    - generating text

# Motivation

- Problem: Supervised Learning is Expensive!
  - Labeling data is costly
  - SSL: Use signals that can be created automatically from data.
- Labled data is harder to find. There is much more unlabled data.
- Supervised Learning is not how **we** learn
  - Babies don't get supervision for everything they see!

## Comparison

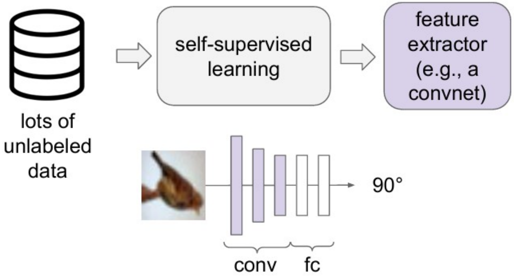
Methods that learn from data without annotations.

- **Unsupervised Learning**: Model isn't told what to predict. Older terminology, not used as much today.
- **Self-Supervised Learning**: Model is trained to predict some naturally occurring signal in the raw data rather than human annotations.
- **Semi-Supervised Learning**: Train jointly with some labeled data and (a lot) of unlabeled data

# Evaluation

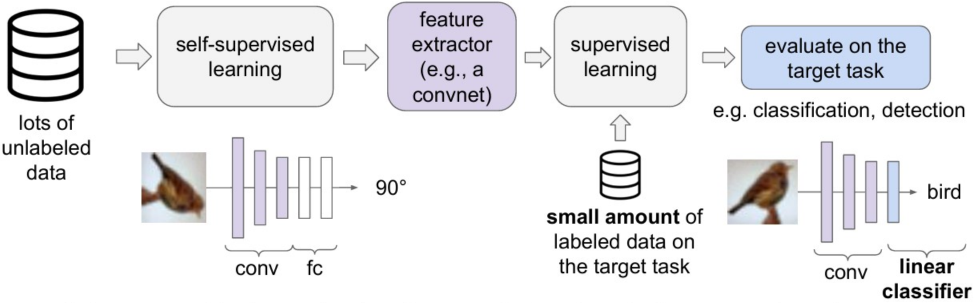
- We usually don't care about the performance of the self-supervised learning task, e.g., we don't care if the model learns to predict image rotation perfectly.
- Evaluate the learned feature encoders on downstream target tasks

# Evaluation Cont.




1. Learn good feature extractors from self-supervised pretext tasks, e.g., predicting image rotations

# Evaluation Cont.




1. Learn good feature extractors from self-supervised pretext tasks, e.g., predicting image rotations

2. Attach a shallow network on the feature extractor; train the shallow network on the target task with small amount of labeled data

# Example

- Pretext task: predict rotations
- Hypothesis: a model could recognize the correct rotation of an object only if it has the "visual commonsense" of what the object should look like unperturbed.
- The model learns to predict which rotation is applied (4-way classification)
- (This slide will be ellaborated on and expanded with diagrams)

## Applications

- **NLP**
  - Models like BERT, GPT, and T5 are based on SSL. They are pre-trained on massive text corpora using tasks like masked language modeling.
  - Improving machine translation quality with SSL-based pre-training on multilingual corpora.
- **Computer Vision**
  - Models like **SimCLR** and BYOL achieve state-of-the-art performance in image classification by learning from unlabeled images.
  - Learning representations for tasks like disease diagnosis from limited annotated medical datasets.

## Applications (Cont.)

- **Speech and Audio Processing**
  - Models like Wav2Vec and HuBERT use SSL in speech recognition task to learn representations from raw audio data. It's especially useful for languages with limited data.
  - Identifying individuals using voice features learned through SSL in speaker identification.
- **Robotics**
  - Leveraging SSL in reinforcement learning helps learn useful state representations for control tasks, enabling robots to perform complex actions.

## Applications (Cont.)

- **Healthcare**
  - Learning representations of DNA sequences to predict mutations or functional regions.
  - Identifying diseases from clinical notes, imaging data, or time-series data such as ECGs.
- **Autonomous Vehicles**
  - Integrating data from cameras, LiDAR, and radar through SSL.
  - Using SSL for anomaly detection in vehicle systems.

# Idea

- Don't learn from isolated images – take images together with some **context**
- **Video**: Image together with adjacent video frames
- **Sound**: Image with audio track from video
- **3D Image**: Image with depth map or point cloud
- **Language:** Image with natural-language text (e.g., captions or descriptions)

## Why Language?

- **Rich Semantics**
  - Just a few words give rich information.
  - Acts as a bridge between sensory data and abstract human understanding.
- **Universality**
  - Language can describe almost any concept
  - Language can act as a **universal medium** for aligning other modalities, even structured data.

# Why Language? (Cont.)

- **Large-Scale Data Availability**
  - The internet contains vast amounts of textual data.
  - Text data is relatively easier to collect, clean, and annotate (no need to experts) compared to modalities like video or audio.
  - Available datasets such as COCO (images and captions)
- **Pretrained Language Models (PLMs) as a Strong Foundation**
  - Large pretrained language models with remarkable capabilities.
  - Language models are highly transferable (transfer learning) across tasks, enabling multimodal systems to adapt to various downstream applications efficiently.

# Definition

- A machine learning technique for training models to distinguish between similar and dissimilar data points.
- **Key Idea**
  - Bring similar data points closer in the embedding space.
  - Push dissimilar data points farther apart.
- **Purpose:** Learn meaningful representations for downstream tasks like classification, clustering, or retrieval
- **Widely Used In:** Representation learning across domains such as computer vision, NLP, and multi-modal tasks.

## Key Concepts

- **Embedding Space**
  - The data points are mapped into a high-dimensional space, called the embedding space.
  - Their relative positions encode similarity or dissimilarity.
- **Positive Pairs:** Data points that are semantically similar.
- **Negative Pairs:** Data points that are semantically different.
- **Objectives**
  - Minimize the distance between the embeddings of positive pairs.
  - Maximize the distance between the embeddings of negative pairs.
- **Loss Functions:** We'll discuss 2 most commonly used loss functions in contrastive learning in the following slides.

# Loss Functions - Contrastive Loss [3] [4]

- Contrastive loss was first introduced in 2005 by Yann Le Cunn et al.
- Its original application was in Dimensionality Reduction.

[3] Dimensionality Reduction by Learning an Invariant Mapping
[4] Losses explained: Contrastive Loss

# Loss Functions - Contrastive Loss (Cont.)

$$D_W\left(\vec{X}_1, \vec{X}_2\right) = \left\| G_W\left(\vec{X}_1\right) - G_W\left(\vec{X}_2\right) \right\|_2$$

- $D_W\left(\vec{X}_1, \vec{X}_2\right)$ is dissimilarity between the two data points $\vec{X}_1$ and $\vec{X}_2$.
- $G_W$ is an embedding function parameterized by $W$.
- Generally, $D_W$ can be any metric that indicates the dissimilarity between $\vec{X}_1$ and $\vec{X}_2$.

## Loss Functions - Contrastive Loss (Cont.)

$$L\left(W,(Y,\vec{X}_1,\vec{X}_2)^i\right) = (1-Y)L_S\left(D_W(\vec{X}_1^i,\vec{X}_2^i)\right) + YL_D\left(D_W(\vec{X}_1^i,\vec{X}_2^i)\right)$$

- $\left(Y,\vec{X}_1,\vec{X}_2\right)^i$ is the $i$-th labeled sample pair.
- $Y=0$ if $\vec{X}_1$ and $\vec{X}_2$ are deemed similar, and $Y=1$ if they are deemed dissimilar.
- $L_S$ is the partial loss function for a pair of similar points.
- $L_D$ is the partial loss function for a pair of dissimilar points.
- $L_S$ and $L_D$ must be designed such that minimizing $L$ with respect to $W$ would result in low values of $D_W$ for similar pairs and high values of $D_W$ for dissimilar pairs.

# Loss Functions - Contrastive Loss (Cont.)

$$\mathcal{L}(W) = \sum_{i=1}^{P} L\left(W, \left(Y, \vec{X}_1, \vec{X}_2\right)^i\right)$$

- $P$ is the number of training pairs (which may be as large as the square of the number of samples).

# Loss Functions - InfoNCE Loss[5]

- First, we'll explore this loss from a theoretical perspective which has been discussed in its original paper.
- Next, we'll discuss how it can be applied in practice.
- It's the loss in its original paper:

$$\mathcal{L}_{\mathrm{N}} = -\mathbb{E}_X \left[ \log \frac{\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}}{\sum_{x_j \in X} \frac{p(x_t|c_t)}{p(x_t)}} \right]$$

[5] Representation Learning with Contrastive Predictive Coding

# Loss Functions - InfoNCE Loss (Cont.)

- Let's start with mutual information.
- We have a set $X = \{x_1, \ldots, x_N\}$ of $N$ random samples containing one positive sample from $p(x_{t+k} \mid c_t)$ and $N-1$ negative samples from the **proposal** distribution $p(x_{t+k})$
- Our purpose is to maximize mutual information:

$$I(x_{t+k}; c_t) = \sum_{x_{t+k}, c_t} p(x_{t+k}, c_t) \log \frac{p(x_{t+k} \mid c_t)}{p(x_{t+k})}$$

- $c_t$ is context latent representation.

# Loss Functions - InfoNCE Loss (Cont.)

- We know:

$$I(x_{t+k}; c_t) \leq \log N \rightarrow I(x_{t+k}; c_t) \geq \log N - \mathscr{L}_{\mathrm{N}}$$

- $\mathscr{L}_{\mathrm{N}}$ quantifies the gap between the true mutual information and the approximation.
- Minimizing $\mathscr{L}_{\mathrm{N}}$ effectively maximizes the mutual information.

# Loss Functions - InfoNCE Loss (Cont.)

- Categorical cross-entropy of classifying the positive sample correctly, with $\frac{f_k}{\sum_X f_k}$ being the prediction of the model.

$$\mathcal{L}_{\mathrm{N}} = -\mathbb{E}_X \left[ \log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k\left(x_j, c_t\right)} \right]$$

- We want to optimize it.

## Loss Functions - InfoNCE Loss (Cont.)

- Let's write the optimal probability for this loss as $p(d = i \mid X, c_t)$ with $[d = i]$ being the indicator that sample $x_i$ is the **positive** sample.
- The probability that sample $x_i$ was drawn from the conditional distribution $p(x_{t+k} \mid c_t)$ rather than the proposal distribution $p(x_{t+k})$ can be derived as follows:

$$p(d = i \mid X, c_t) = \frac{p(x_i \mid c_t) \prod_{l \neq i} p(x_l)}{\sum_{j=1}^{N} p\left(x_j \mid c_t\right) \prod_{l \neq j} p(x_l)} = \frac{\frac{p(x_i|c_t)}{p(x_i)}}{\sum_{j=1}^{N} \frac{p(x_j|c_t)}{p(x_j)}}$$

- As we can see, the optimal value for $f_k(x_{t+k}, c_t)$ in $\mathscr{L}_{\mathrm{N}}$ is proportional to $\frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$ and this is independent of the choice of the number of negative samples $N-1$.

# Loss Functions - InfoNCE Loss (Cont.)

- We can evaluate the mutual information between the variables $c_t$ and $x_{t+k}$ as follows:

$$I(x_{t+k}, c_t) \geq \log(N) - \mathscr{L}_\mathrm{N}$$

- It becomes tighter as $N$ becomes larger.
- Minimizing the InfoNCE loss $\mathscr{L}_\mathrm{N}$ maximizes a lower bound on mutual information.

# Loss Functions - InfoNCE Loss (Cont.)

- In practice, we have:

$$\mathcal{L}_{\mathrm{N}} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{\exp\left(\mathrm{sim}\left(x_i, c_i\right)/\tau\right)}{\sum_{j=1}^{N}\exp\left(\mathrm{sim}\left(x_i, c_j\right)/\tau\right)}$$

- Used in models like SimCLR, MoCo, CLIP, and others.
- Next, we want to derive this formula from the theoretical one.

# Loss Functions - InfoNCE Loss (Cont.)

- **Step 1:**

$$\frac{p(x \mid c)}{p(x)} = \exp\left(\log\left(\frac{p(x \mid c)}{p(x)}\right)\right)$$

- But in practice, we rarely know the true densities $p(x \mid c)$ and $p(x)$.
- Instead, we learn a function that approximates their log-ratio.
- A common approach is to let a neural network produce embeddings $f(x)$ and $g(c)$.

# Loss Functions - InfoNCE Loss (Cont.)

$$\log\left(\frac{p(x \mid c)}{p(x)}\right) \approx \operatorname{sim}\left(f(x), g(c)\right) \xrightarrow{\text{we annotate it as}} \operatorname{sim}(x, c) \rightarrow \frac{p(x \mid c)}{p(x)} \approx \exp\left(\operatorname{sim}(x, c)\right) \tag{1}$$

- $\operatorname{sim}(x, c)$ is similarity function (e.g., cosine similarity or dot product).
- Replacing unknown densities with a similarity function, yielding a **softmax** function (which we'll discuss).
- It's straightforward to implement using standard deep-learning toolkits.

## Loss Functions - InfoNCE Loss (Cont.)

- Why $\text{sim}(x, c)$ works?
    - It becomes large (positive) for the true **positive** pair $(x, c)$.
    - It becomes relatively small (negative) for **negative** pairs $(x, c')$.

$$\text{sim}(x, c) \gg \text{sim}(x, c') \longleftrightarrow p(x, c) \gg p(x, c')$$

- This is the property required to approximate the ratio $p(x \mid c) / p(x)$.

# Loss Functions - InfoNCE Loss (Cont.)

- **Step 2:**
- In practice, we don't have the full distribution $X$ or its expectations.
- Instead, we approximate this using batches of size $N$.
- Each $x_{t+k}$ is treated as the **positive sample**, and the other $x_j$s in the batch are treated as **negative samples**.
- The expectation becomes a summation over batches:

$$\mathcal{L}_{\mathrm{N}} = -\mathbb{E}_X \left[ \log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \right] \approx -\frac{1}{N} \sum_{i=1}^{N} \log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} \tag{2}$$

# Loss Functions - InfoNCE Loss (Cont.)

- **Step 3:**
- To control the sharpness of the similarity distribution, a temperature parameter $\tau$ is introduced:

$$\mathrm{sim}\,(x, c) \rightarrow \frac{\mathrm{sim}\,(x, c)}{\tau} \tag{3}$$

- $\tau$ helps balance gradients during training:
  - With no $\tau$, large similarity scores might dominate the gradients, leading to unstable updates.
  - A carefully chosen $\tau$ scales the scores appropriately, ensuring stable convergence.

# Loss Functions - InfoNCE Loss (Cont.)

- $\tau$ affects the distribution of similarity scores after applying the softmax function; in other words, it influences the sharpness of the softmax.
- Low $\tau$:
  - High sharpness.
  - The softmax heavily favors the largest score.
  - The distribution becomes more concentrated on the top-scoring pair.
  - Encourages the model to focus strongly on the positive sample while ignoring negatives.
  - The loss becomes more sensitive to small differences in scores.
- High $\tau$:
  - Low sharpness.
  - The softmax smooths the distribution, making it more uniform.
  - This encourages the model to consider a broader set of samples, not just the top-scoring pair.
  - Useful when the data is noisy or when the model needs to generalize better.

# Loss Functions - InfoNCE Loss (Cont.)

- **Finally:** From equations (1) to (3), we derive:

$$\mathscr{L}_{\mathrm{N}} = -\mathbb{E}_X \left[ \log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k\left(x_j, c_t\right)} \right] \approx -\frac{1}{N} \sum_{i=1}^{N} \log \frac{\exp(\mathrm{sim}(x_i, c_i) / \tau)}{\sum_{j=1}^{N} \exp\left(\mathrm{sim}\left(x_i, c_j\right) / \tau\right)}$$

## Common Components

- Dataset :
  - supervised: $D_m = \{(x_1^1, \cdots, x_M^1, y^1), \cdots, (x_1^n, \cdots, x_M^n, y^n)\}$
  - self-supervised: $D_m = \{(x_1^1, \cdots, x_M^1), \cdots, (x_1^n, \cdots, x_M^n)\}$
- The psudo-label or signal generated for SSL can be denoted as $z = P(x_1, \cdot, x_M)$.
- Modality Encoder(s): $c = e_k(x_k^i; \theta_k)$ for each modality $k$.
- Fusion Module: $f_\psi$ to integrate the encoded information of different modalities
- Pretext task head (like a predictive head) : $g_\gamma$ and some SSL loss $\mathcal{L}_{SSL}$

# Architectures

- There many veriations and structures

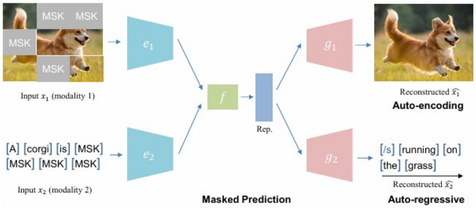



Figure 1: Figure 1 masked prediction frameworks

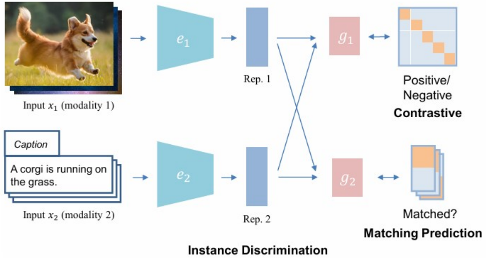



Figure 2: Figure 2 instance discrimination objectives

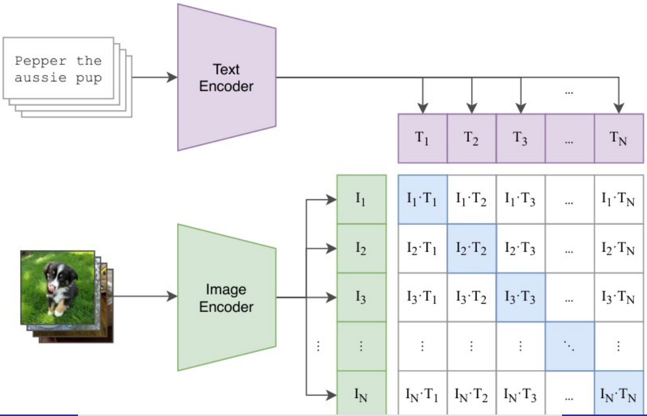
# CLIP

- Connecting text and images
- Contrastive Language–Image Pre-training
- CLIP $\implies$ a shared representation(embedding) between two modalities (text and images) by training on a large dataset of image-text pairs.

# CLIP Cont.

- Image Encoder: a Vision Transformer (ViT) or a ResNet.
- Text Encoder: A Transformer model

(1) Contrastive pre-training



## CLIP Goals

- CLIP was designed to mitigate a number of major problems:
- Costly datasets: Deep learning needs a lot of data, manually labeled datasets are expensive to construct.
  - CLIP learns from text–image pairs that are already publicly available on the internet
- Narrow: An ImageNet model excels at predicting the 1000 ImageNet categories but requires additional data and fine-tuning for other tasks.
  - CLIP can be adapted to perform a wide variety of visual classification tasks without needing additional training examples.

# Applications

- CLIP enables searching for images by interpreting natural language descriptions.
- It classifies images into categories defined at inference using textual prompts (e.g., "This is a photo of a [label]").
- Visual Question Answering (VQA) becomes possible with CLIP's text-image alignment.
- Integrated into DALL·E or Stable Diffusion, it improves image generation with scoring and feedback loops.
- CLIP enhances robot-human interaction by understanding instructions about objects or scenes.

# Applications

- It matches user queries with product images for seamless search and discovery in e-commerce.
- The model enables zero-shot classification to identify new or rare medical conditions.
- CLIP detects anomalies in manufacturing processes or equipment visuals to ensure quality.
- It personalizes digital experiences based on user preferences described in natural language.

# Contributions

**These slides are authored by:**

- Hooman Zolfaghari
- Amir Mohammad Fakhimi