Self-supervised, contrastive learning


SUVRIT SRA

Acknowledgments: Joshua Robinson, Stefanie Jegelka

ml.mit.edu
Breakthroughs in supervised deep learning

1.2 million labeled images

![Graph showing the development of top-1 accuracy in image classification models over time, with key models like SIFT + FVs, AlexNet, SPPNet, Inception V3, NASNET-A, and Meta Pseudo Labels. The x-axis represents years from 2012 to 2020, and the y-axis represents top-1 accuracy from 40% to 100%. The graph highlights the evolution of models and their performance improvements.]
… but deep learning is really bad without labels

Performance

# labeled data

deep neural net

shallow neural net

non-neural network methods
Learning without labels
Learning without labels
Weakly-supervised learning
Learning without labels
Weakly-supervised learning

Metric Learning
Learning without labels
Weakly-supervised learning

Metric Learning

Self-supervised learning
Learning without labels

Weakly-supervised learning

Metric Learning

Self-supervised learning

Contrastive learning
**Metric / similarity driven learning**

**Idea:** Learn a representation by learning a distance metric

This idea is the forerunner of “modern representation learning”
Metric / similarity driven learning

**Idea:** Learn a representation by learning a distance metric

This idea is the forerunner of “modern representation learning”
Linear Metric learning setup

Let $x_1, \ldots, x_n$ denote training data (e.g., images, text, …)

**Assume:** Input points $x_i$ are vectors
Linear Metric learning setup

Let $x_1, \ldots, x_n$ denote training data (e.g., images, text, …)

**Assume:** Input points $x_i$ are vectors

**Goal:** Learn a linear representation so that in "embedding" space "similar" points are closer to each other and farther away from "dissimilar" ones
Linear Metric learning setup

Let \( x_1, \ldots, x_n \) denote training data (e.g., images, text, …)

**Assume:** Input points \( x_i \) are vectors

**Goal:** Learn a linear representation so that in “embedding” space “similar” points are closer to each other and farther away from “dissimilar” ones.
Linear Metric learning setup

Let $x_1, \ldots, x_n$ denote training data (e.g., images, text, …)

**Assume:** Input points $x_i$ are vectors

**Goal:** Learn a linear representation so that in “embedding” space “similar” points are closer to each other and farther away from “dissimilar” ones.
Let $x_1, \ldots, x_n$ denote training data (e.g., images, text, …)

**Assume:** Input points $x_i$ are vectors

**Goal:** Learn a linear representation so that in "embedding" space "similar" points are closer to each other and farther away from "dissimilar" ones.
Let \( x_1, \ldots, x_n \) denote training data (e.g., images, text, …)

**Assume:** Input points \( x_i \) are vectors

**Goal:** Learn a linear representation so that in “embedding” space “similar” points are closer to each other and farther away from “dissimilar” ones.
Linear Metric learning setup

Let $x_1, \ldots, x_n$ denote training data (e.g., images, text, ...) 

**Assume:** Input points $x_i$ are vectors

---

**Goal:** Learn a linear representation so that in “embedding” space “similar” points are closer to each other and farther away from “dissimilar” ones.
Linear metric learning setup

We assume: weakly-supervised training data

\[ S := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class}\} \]

\[ D := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes}\} \]
Linear metric learning setup

We assume: weakly-supervised training data

$\mathcal{S} := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class}\}$

$\mathcal{D} := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes}\}$

**Goal:** learn linear transformation to respect similarity
Linear metric learning setup

We assume: weakly-supervised training data

\[ S := \{ (x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class} \} \]

\[ D := \{ (x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes} \} \]

**Goal:** learn linear transformation to respect similarity

\[ x \mapsto Lx \]
Linear metric learning setup

We assume: weakly-supervised training data

\[ S := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class}\} \]
\[ D := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes}\} \]

**Goal:** learn linear transformation to respect similarity

\[ x \mapsto Lx \]

\[ \|x_i - x_j\| \mapsto \|Lx_i - Lx_j\| \]
Linear metric learning setup

We assume: weakly-supervised training data

\[ S := \{ (x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class} \} \]

\[ D := \{ (x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes} \} \]

**Goal:** learn linear transformation to respect similarity

\[ x \mapsto Lx \]

\[ \|x_i - x_j\| \mapsto \|Lx_i - Lx_j\| \]

**Key insight**

\[ \|Lx - Ly\|^2 = (x - y)^T L^T L (x - y) \]
Linear metric learning setup

We assume: weakly-supervised training data

\[ S := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class}\} \]

\[ D := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes}\} \]

**Goal:** learn linear transformation to respect similarity

\[ x \mapsto Lx \]

\[ \|x_i - x_j\| \mapsto \|Lx_i - Lx_j\| \]

**Key insight**

\[ \|Lx - Ly\|^2 = (x - y)^T L^T L (x - y) \]

\[ A \geq 0 \]
Linear metric learning summary

We assume: weakly-supervised training data

\[ S := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class}\} \]

\[ D := \{(x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes}\} \]

**Aim:** Learn Mahalanobis distance

\[ d_A(x, y) := (x - y)^T A (x - y) \]

s.t. for pairs in \( S \), \( d_A \) is small; for pairs in \( D \), \( d_A \) large
Linear metric learning summary

We assume: weakly-supervised training data

\[ S := \left\{ (x_i, x_j) \mid x_i \text{ and } x_j \text{ are in the same class} \right\} \]

\[ D := \left\{ (x_i, x_j) \mid x_i \text{ and } x_j \text{ are in different classes} \right\} \]

**Aim:** Learn Mahalanobis distance \( d_A(x, y) := (x - y)^TA(x - y) \)

s.t. for pairs in \( S \), \( d_A \) is small; for pairs in \( D \), \( d_A \) large

Problem introduced in 2003 by Xing, Ng, Jordan, Russell
Still remains a subject of research!
Zeroth (naive) model*

\[ d_A(x, y) := (x - y)^T A (x - y) \]

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) - \lambda \sum_{(x_i, x_j) \in D} d_A(x_i, x_j)
\]

Fails empirically, but also insufficient since poor scaling or bad choice of \( D \) can drive the \( A \) to be very large and lead to a useless solution...

[skipped during lecture]
First model: MMC*

MMCC
[Xing, Jordan, Russell, Ng 2002]

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j)
\]

such that
\[
\sum_{(x_i, x_j) \in D} \sqrt{d_A(x_i, x_j)} \geq 1
\]

Semidefinite Programming (SDP): Convex problem

**Question:** Why the square-root?

*[skipped during lecture]*
Second model: LMNN*

**LMNN**
[Weinberger, Saul 2005]

widely used, successful model
made metric learning popular

Convex formulation inspired by famous SVM

\[
\begin{align*}
\min_{A \succeq 0} & \sum_{(x_i, x_j) \in S} \left[ (1 - \mu) d_A(x_i, x_j) + \mu \sum_l (1 - y_{il}) \xi_{ijl} \right] \\
\text{s.t.} & \quad d_A(x_i, x_l) - d_A(x_i, x_j) \geq 1 - \xi_{ijl} \\
\xi_{ijl} & \geq 0
\end{align*}
\]

Question: What happens without the slack variables $\xi$

term penalizes small distances between differently labeled examples

\( y_{il} = 1 \) if and only if \( y_i = y_l \), i.e., if points \( i \) and \( l \) are “similar/same”

[skipped during lecture]
Third model: ITML*

**ITML**

[Davis, Kulis, Jain, Sra, Dhillon 2007]

relative entropy b/w Gaussians

\[
\min_{A \succeq 0} D_{ld}(A, A_0)
\]

such that

\[
d_A(x, y) \leq u, \quad (x, y) \in S,
\]

\[
d_A(x, y) \geq l, \quad (x, y) \in D
\]

\[
D_{ld}(A, A_0) := \text{tr}(A A_0^{-1}) - \log \det(A A_0^{-1}) - d
\]

Convex, but nonlinear SDP. Admits efficient algorithm

widely used, successful model
popular alternative to LMNN

nonlin cost, permit
gradient-based opt.

[skipped during lecture]
Tons of other models*

Main concerns of ITML, LMNN, etc.:

They do not scale well to the large problems, with respect to:

1. The number of constraints (how?)
2. The dimensionality of the input data (why?)

Explore: Implement solvers for LMNN and ITML and experiment

[skipped during lecture]
New model: Geometric approach

\[ d_A(x, y) := (x - y)^T A (x - y) \]

**Naive model**

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) - \lambda \sum_{(x_i, x_j) \in D} d_A(x_i, x_j)
\]
New model: Geometric approach

Naive model

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) - \lambda \sum_{(x_i, x_j) \in D} d_A(x_i, x_j)
\]

\[
d_A(x, y) := (x - y)^T A(x - y)
\]
New model: Geometric approach

Naive model

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) - \lambda \sum_{(x_i, x_j) \in D} d_A(x_i, x_j)
\]

New idea

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_A^{-1}(x_i, x_j)
\]

Intuitively: If \( a > b \), then \( a^{-1} < b^{-1} \)
New model: Geometric approach

\[d_A(x, y) := (x - y)^T A (x - y)\]

\[
\min_{A \geq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_{A^{-1}}(x_i, x_j)
\]
New model: Geometric approach

\[ d_A(x, y) := (x - y)^T A (x - y) \]

\[ \min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_{A^{-1}}(x_i, x_j) \]

Collect similar points into \( S \) and dissimilar into \( D \)

\[
S := \sum_{(x_i, x_j) \in S} (x_i - x_j)(x_i - x_j)^T,
\]

\[
D := \sum_{(x_i, x_j) \in D} (x_i - x_j)(x_i - x_j)^T
\]
New model: Geometric approach

\[ d_A(x, y) := (x - y)^T A(x - y) \]

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_A^{-1}(x_i, x_j)
\]

Collect similar points into \( S \) and dissimilar into \( D \)

\[
S := \sum_{(x_i, x_j) \in S} (x_i - x_j)(x_i - x_j)^T, \\
D := \sum_{(x_i, x_j) \in D} (x_i - x_j)(x_i - x_j)^T
\]

Equivalent problem

\[
\min_{A \succ 0} h(A) := \text{tr}(AS) + \text{tr}(A^{-1}D)
\]
New model: Geometric approach

\[ d_A(x, y) := (x - y)^T A (x - y) \]

\[
\min_{A \succeq 0} \sum_{(x_i, x_j) \in S} d_A(x_i, x_j) + \sum_{(x_i, x_j) \in D} d_A^{-1}(x_i, x_j)
\]

Collect similar points into \( S \) and dissimilar into \( D \)

\[
S := \sum_{(x_i, x_j) \in S} (x_i - x_j)(x_i - x_j)^T,
\]

\[
D := \sum_{(x_i, x_j) \in D} (x_i - x_j)(x_i - x_j)^T
\]

Equivalent problem

\[
\min_{A \succ 0} h(A) := \text{tr}(AS) + \text{tr}(A^{-1}D)
\]

Closed form solution!
Closed form solution!

\[ \nabla h(A) = 0 \iff S - A^{-1} DA^{-1} = 0 \]

\[ A = S^{-1} \# \frac{1}{2} D \]

Where we define the matrix “Geometric Mean” via

\[ X \#_t Y := X^{\frac{1}{2}} (X^{-\frac{1}{2}} Y X^{-\frac{1}{2}})^t X^{\frac{1}{2}} \]

[details skipped during lecture]
**Question:** How to handle weights?

\[
\min_{A \succ 0} \ h(A) := \text{tr}(AS) + \text{tr}(A^{-1}D)
\]

**Question:** What’s wrong with this formulation?

\[
\min_{A \succ 0} \ \text{tr}(AS) + \lambda \text{tr}(A^{-1}D)
\]

\[
S = \lambda A^{-1}D A^{-1}
\]

\[
\Rightarrow A = S^{-1} \# (\lambda D)
\]

\[
= \sqrt{\lambda} \cdot (S^{-1} \# D)
\]

*Simply scale A. No trade-off between S & D.*
Riemannian geometry: handling weights*

- **GMML solution via alternative objective**
  \[
  \min_{A \succ 0} \delta_R^2(A, S^{-1}) + \delta_R^2(A, D)
  \]

  where we use the *Riemannian distance*
  \[
  \delta_R(X, Y) := \|\log(Y^{-1/2}XY^{-1/2})\|_F \quad \text{for } X, Y \succ 0
  \]

- **Weighted GMML formulation**
  \[
  \min_{A \succ 0} h_t(A) := (1 - t) \delta_R^2(A, S^{-1}) + t \delta_R^2(A, D)
  \]

- **Weighted geometric mean**
  \[
  A = S^{-1} \#_t D \quad \text{for } t \in [0, 1]
  \]

**Still a closed-form solution!**

*[skipped during lecture]*
Experiments

NOTE: May think of this as a “supervised whitening transform”

Code: https://github.com/PouriaZ/GMML
# Experiments

## Running time in seconds

<table>
<thead>
<tr>
<th>DATA SET</th>
<th>GMML</th>
<th>LMNN</th>
<th>ITML</th>
<th>FLATGEO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGMENT</td>
<td>0.0054</td>
<td>77.595</td>
<td>0.511</td>
<td>63.074</td>
</tr>
<tr>
<td>LETTERS</td>
<td>0.0137</td>
<td>401.90</td>
<td>7.053</td>
<td>13543</td>
</tr>
<tr>
<td>USPS</td>
<td>0.1166</td>
<td>811.2</td>
<td>16.393</td>
<td>17424</td>
</tr>
<tr>
<td>ISOLET</td>
<td>1.4021</td>
<td>3331.9</td>
<td>1667.5</td>
<td>24855</td>
</tr>
<tr>
<td>MNIST</td>
<td>1.6795</td>
<td>1396.4</td>
<td>1739.4</td>
<td>26640</td>
</tr>
</tbody>
</table>

**NOTE:** May think of this as a “supervised whitening transform”

Code: [https://github.com/PouriaZ/GMML](https://github.com/PouriaZ/GMML)
Learning without labels
Weakly-supervised learning

Metric Learning

Self-supervised learning

Contrastive learning
What if we have just a few labels?
What if we have just a few labels?

Use auxiliary data / task!
What if we have just a few labels?

Use auxiliary data / task!
What if we have just a few labels?

Use auxiliary data / task!

Medical imaging

Fibrosis

Xie & Richmond, ECCV 2018 workshop

beanding plane  brown root fungus  scalded milk
What if we have just a few labels?

Use auxiliary data / task!

Medical imaging

Sentiment analysis

Xie & Richmond, ECCV 2018 workshop

https://hackernoon.com/drafts/523x232/
What if we have just a few labels?

Medical imaging

Use auxiliary data / task!

Sentiment analysis

Labels: [MASK] = store; [MASK] = gallon

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence
Goal: train deep network encoder $f(x)$ using lots of unlabeled data…

Key idea: We don’t have a supervised task, so “invent” one!
- No generative model
- “Pretend” labels automatically generated
Goal: train deep network encoder $f(x)$ using lots of unlabeled data…

Key idea: We don’t have a supervised task, so “invent” one!
- No generative model
- “Pretend” labels automatically generated
Self-supervised representation learning

**Goal:** train deep network encoder $f(x)$ using lots of *unlabeled data*…

- **Input:** $x$
- **Encoder:** $f(x)$
- **Representation:** $z$
- **Linear classifier:** $g(z)$
- **Label:** $y$

… so that training a simpler (e.g. linear) classifier on the representation suffices

requires less labeled data than supervised deep network

**Key idea:** We don’t have a supervised task, so “invent” one!
- No generative model
- “Pretend” labels automatically generated
Pre-training and “Pretext” tasks
Pre-training and “Pretext” tasks

Where does the pretext task come from?
Where does the pretext task come from?

Think of it as a form of “global supervision”, modeling consistency, invariances, stability, etc.
Pre-training and “Pretext” tasks

Where does the pretext task come from?

Think of it as a form of “global supervision”, modeling consistency, invariances, stability, etc.

input $X$  \hspace{5cm} embedding $f(X)$  \hspace{5cm} output $W$

Pretext task $m$ data points

$Z$
Pre-training and “Pretext” tasks

Where does the pretext task come from?

Think of it as a form of “global supervision”, modeling consistency, invariances, stability, etc.

Pretext task

$m$ data points

input $X$

embedding $Z$

output $W$

Fix & transfer $f(X)$
Pre-training and “Pretext” tasks

Where does the pretext task come from?

Think of it as a form of “global supervision”, modeling consistency, invariances, stability, etc.

Fix & transfer \( f(X) \)

Pretext task

\( m \) data points

\[
\begin{align*}
\text{input} & \quad \text{embedding} & \quad \text{output} \\
X & \quad f(X) & \quad W
\end{align*}
\]

Target task

\( n \ll m \) data points

\[
\begin{align*}
\text{input} & \quad \text{embedding} & \quad \text{label} \\
X & \quad f(X) & \quad Y \\
g(Z)
\end{align*}
\]
Example Pretext Tasks: Vision
Example Pretext Tasks: Vision

Target Tasks:
object detection,
visual data mining, ...
Target Tasks:
object detection, visual data mining, ...
Example Pretext Tasks: Vision

Example:

(Doersch, Gupta, Efros 2015)

Target Tasks:
object detection, visual data mining, ...


Predict relative patch locations

(Doersch et al. 2016)
Example Pretext Tasks: NLP

Target Task:
Question answering
Sentiment analysis

ELMo, GPT, BERT, T5, ...

Example Pretext Tasks: NLP

**ELMo, GPT, BERT, T5, ...**

- **Sentence A** = The man went to the store.
- **Sentence B** = He bought a gallon of milk.
- **Label** = IsNextSentence

- **Sentence A** = The man went to the store.
- **Sentence B** = Penguins are flightless.
- **Label** = NotNextSentence


**Target Task:**
- Question answering
- Sentiment analysis

Example Pretext Tasks: NLP

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence

Target Task:
Question answering
Sentiment analysis

ELMo, GPT, BERT, T5, ...


More general pretext task: Contrastive

- Positive pair
- Negative sample
More general pretext task: Contrastive

(Invariance to perturbations)

positive pair

negative sample
More general pretext task: Contrastive

(Invariance to perturbations)

... despite the simplicity, this idea works really well!
The advances of self-supervised learning

linear fine tuning on ImageNet

- Colorization (AlexNet)
- Rotation (RevNet50-4w)
- LocalAgg (ResNet-50)
- CMC (ResNet-50x2)
- SimCLR (ResNet-50 4x)
- SimCLRv2 (ResNet-152 x3, SK)

Other models vs. State-of-the-art models
Why do pre-trained representations help?

Example:

Which Sesame Street ? is your favorite

Suvrit Sra (suvrit@mit.edu) 6.867 (Fall 2021). Lect 12: Self supervised learning
Why do pre-trained representations help?

![Diagram showing input, embedding, and output with arrows](image)

- **Input**: $X$
- **Embedding**: $Z$
- **Output**: $Y$

**Example:**
- Which
- Sesame
- Street
- ?
- is
- your
- favorite

![Example image with text boxes](image)
Why do pre-trained representations help?

Common intuition: same “semantic knowledge”
the only way to solve the jigsaw is to understand that it is a picture of a cat

Formalization?

How much can pre-training help generalization?
Self-supervision can accelerate learning

**Theorem** *(Robinson et al 2020)*. If central condition holds and the pretext task has learning rate $O(1/m^\alpha)$, we use $m = \Omega(n^\beta)$ pretext samples, then with probability $1 - \delta$, the target task has excess risk
Self-supervision can accelerate learning

**Theorem** *(Robinson et al 2020).* If central condition holds and the pretext task has learning rate $O(1/m^\alpha)$, we use $m = \Omega(n^\beta)$ pretext samples, then with probability $1 - \delta$, the target task has excess risk

$$O\left(\frac{\alpha \beta \log n + \log(1/\delta)}{n} + \frac{1}{n^{\alpha \beta}}\right)$$
Self-supervision can accelerate learning

**Theorem (Robinson et al 2020).** If central condition holds and the pretext task has learning rate $O(1/m^\alpha)$, we use $m = \Omega(n^\beta)$ pretext samples, then with probability $1 - \delta$, the target task has excess risk

$$O \left( \frac{\alpha \beta \log n + \log(1/\delta)}{n} + \frac{1}{n^{\alpha \beta}} \right)$$

Rate: $O(n^{-\gamma})$

Learning without labels
Weakly-supervised learning

Metric Learning

Self-supervised learning

Contrastive learning
Setting up contrastive learning: the loss function

\[
f(f) = f(f)
\]
Setting up contrastive learning: the loss function

Learn “similarity” scores so that positives much more similar to each other than to negatives
Setting up contrastive learning: the loss function

Learn "similarity" scores so that positives much more similar to each other than to negatives.
Setting up contrastive learning: the loss function

Learn “similarity” scores so that positives much more similar to each other than to negatives

\[
\min_f \mathbb{E}_{x,x^+,\{x_i^\}^N_{i=1}} \left[ -\log \frac{e^{f(x)^\top f(x^+)}}{e^{f(x)^\top f(x^+)} + \sum_{i=1}^N e^{f(x)^\top f(x_i^-)}} \right]
\]
Setting up contrastive learning: the loss function

Learn “similarity” scores so that positives much more similar to each other than to negatives

\[
\min_f \mathbb{E}_{x,x^+,\{x_i^\}^N_{i=1}} \left[ -\log \frac{e^{f(x)\top f(x^+)}}{e^{f(x)\top f(x^+) + \sum_{i=1}^N e^{f(x)\top f(x_i^-)}}} \right]
\]
Learn “similarity” scores so that positives much more similar to each other than to negatives

\[
\min_f \mathbb{E}_{x,x^+,\{x^-\}_{i=1}^N} \left[ -\log \frac{e^{f(x)\top f(x^+)} \sum_{i=1}^N e^{f(x)\top f(x^-)}}{e^{f(x)\top f(x^+)} + \sum_{i=1}^N e^{f(x)\top f(x^-)}} \right]
\]
Setting up contrastive learning: the loss function

Learn “similarity” scores so that positives much more similar to each other than to negatives

\[
\min_f \mathbb{E}_{x,x^+, \{x^-\}_{i=1}^N} \left[ -\log \frac{e^{f(x)^\top f(x^+)} \mathcal{L}_{x^+} \sum_{i=1}^N e^{f(x)^\top f(x^-)}}{\mathcal{L}_{x^+} e^{f(x)^\top f(x^+)} + \sum_{i=1}^N \mathcal{L}_{x^-} e^{f(x)^\top f(x^-)}} \right]
\]
Setting up contrastive learning: the loss function

Learn “similarity” scores so that positives much more similar to each other than to negatives

$$\min_f \mathbb{E}_{x, x^+, \{x_i^-\}_{i=1}^N} \left[ -\log \frac{e^{f(x)^\top f(x^+)} \sum_{i=1}^N e^{f(x)^\top f(x_i^-)}} {e^{f(x)^\top f(x^+)} + \sum_{i=1}^N e^{f(x)^\top f(x_i^-)}} \right]$$
Setting up contrastive learning: the loss function

Learn “similarity” scores so that positives much more similar to each other than to negatives

\[
\min_f \mathbb{E}_{x, x^+, \{x_i^\}^N_{i=1}} \left[ -\log \frac{e^{f(x)^\top f(x^+)} }{e^{f(x)^\top f(x^+)} + \sum_{i=1}^N e^{f(x)^\top f(x_i^-)}} \right]
\]
Learn “similarity” scores so that positives much more similar to each other than to negatives

\[
\min_f \mathbb{E}_{x,x^+,\{x_i^-\}} \left[ -\log \frac{e^{f(x)^\top f(x^+)} + \sum_{i=1}^N e^{f(x)^\top f(x_i^-)}}{e^{f(x)^\top f(x^+)} + \sum_{i=1}^N e^{f(x)^\top f(x_i^-)}} \right]
\]

Can outperform supervised pretraining \( (He \ et \ al \ 2020, \ Misra \ & \ van \ der \ Maaten \ 2020) \)

**BUT:** how get positive / negative pairs without labels?
Generating “positive” and “negative” examples

Generating positive & negative samples

**positives:** a random combination of data augmentations

Image from (Chen et. al. 2020)
Generating “positive” and “negative” examples

Generating positive & negative samples

positives: a random combination of data augmentations

Image from (Chen et. al. 2020)
Generating “positive” and “negative” examples

Generating positive & negative samples

**positives:** a random combination of data augmentations

Image from (Chen et. al. 2020)

**negatives:** uniformly sampled at random from dataset

Uniformly sampled from STL10 dataset
Key factors driving performance

(SimCLR, MoCo)

aggressive use of augmentation
Key factors driving performance

(SimCLR, MoCo)

- Aggressive use of augmentation
- Momentum encoding enabling large negative batches

\[ \theta_k \leftarrow m \theta_k + (1 - m) \theta_q. \]
Key factors driving performance

(SimCLR, MoCo)

- Aggressive use of augmentation
- Momentum encoding enabling large negative batches
- The projection head

(Also key: normalization onto sphere)

\[ \theta_k \leftarrow m\theta_k + (1 - m)\theta_q. \]
Impact of augmentation / transformations

Impact of progressively removing transformations

(Figure from Grill et al. 2020)
Large batches of negative examples help

Figure 9. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar is a single run from scratch.¹⁰

(Figure from Chen et al. 2020)
Scoring the similarity: so-called projection head

Pretrain using $g \circ f$, finetune using just $f$ (throw away $g$)

$f$ is a deep encoder (e.g. ResNet), $g$ a 2 or 3 layer MLP

(Figure from Chen et al. 2020)
How can you generate negative samples?

negatives are typically sampled uniformly at random from training data.
negatives are typically sampled uniformly at random from training data

Some pros of uniform sampling

- it is easy to implement
- no supervision to required guide sampling
- large negative batches get good coverage
Uniform sampling: what could go wrong?

1) false negatives
Uniform sampling: what could go wrong?

1) false negatives

false negative sample
Uniform sampling: what could go wrong?

1) false negatives

2) easy negatives
Uniform sampling: what could go wrong?

1) false negatives

2) easy negatives

- false negative sample
- model already knows are different
- no useful gradient signal
False negatives: do they matter?

Do they even matter? What can we do about them?
Removing *false negatives* improves generalization.
Impact of false negatives and debiased CL

Removing *false negatives* improves generalization

Embeddings evaluated by fixing the parameters and training a linear model on the learned features.

**Graph:**
- **x**
- **x^+** ~ \( p_x^+ \) false negative sample
- **x_i^-** ~ \( p \)

**Graph:**
- 3062 126 254 510
- Top-1 Accuracy: 95 90 85 80
- Negative Sample Size (N)
- uniform negatives
- no false negatives

Embeddings evaluated by fixing the parameters and training a linear model on the learned features.
Impact of false negatives and debiased CL

Removing *false negatives* improves generalization

**Problem:** training data is unlabeled, so we cannot directly identify false negatives
Removing *false negatives* improves generalization.

**Problem:** training data is unlabeled, so we cannot directly identify false negatives.

**Idea:** use positive and uniform samples to approximate true negatives.

Embeddings evaluated by fixing the parameters and training a linear model on the learned features.
Easy negatives not useful, potentially even bad?

What if the model already knows two samples are different?

No useful gradient signal
**Hard Negatives**

Anchor:

- Willow
- Sycamore
- Maple
- Sequoia
- Oak
- Violets
- Mountain

**Embedding Space:**

- Oak
- Anchor
- Hard negatives
- Easy negatives

Sample Negatives

- Uniformly from Dataset (typical method)

Sample Hard Negatives (our method)
Hard negatives are precisely the samples that your encoder is currently “wrong” on.
What do hard negatives look like?

Anchor

Hard negatives

Uniform Negatives
What do hard negatives look like?

Question: How to sample hard negatives?
How to sample hard negatives?*

As before use positive and uniform samples to approximate true negatives. Have a tuning param to control “hardness”

**uniform negatives**

sample negatives \( \{x_i^-\}_{i=1}^N \) from marginal \( p(x^-) \)

**debiased negatives**

sample negatives \( \{x_i^-\}_{i=1}^N \) from \( p(x^- \mid x, x^- \text{ diff. class}) \)

**hard negatives**

sample negatives \( \{x_i^-\}_{i=1}^N \) from \( q_\beta(x^- \mid x, x^- \text{ diff. class}) \) where

\[
q_\beta(x^-) \propto e^{\beta f(x)^\top f(x^-)} \cdot p(x^-)
\]

avoid false hard negatives, approximated using Positive-Unlabeled learning methods

hard negatives: \( \beta \) controls the level of “hardness”

sampling from \( q_\beta \) using importance sampling with proposal \( p(x^-) \) so we can generate in-batch hard negatives
**Hard Negative Sampling: Implementation**

Implementation is simple & efficient

**sampling from** \( q_\beta \) **using importance sampling with proposal**

\[ p(x^-) \] so we can generate in-batch hard negatives

```python
# pos : exp of inner products for positive examples
# neg : exp of inner products for negative examples
# N  : number of negative examples
# t  : temperature scaling
# tau_plus: class probability
# beta : concentration parameter

#Original objective
standard_loss = -log(pos.sum() / (pos.sum() + neg.sum()))

#Hard sampling objective (Ours)
reweight = (beta*neg) / neg.mean()
Neg = max((-N*tau_plus*pos + reweight*neg).sum() / (1-tau_plus), e**(-1/t))
hard_loss = -log( pos.sum() / (pos.sum() + Neg))
```
Hard Negative Sampling: Experimental results

Comparison on vision problems

![Graph showing Top-1 Accuracy vs. Negative Sample Size N (STL10)]

- Hard negative sampling & debasing (removing false negatives) also help on **language** and **pixel-level control problems**
Summary & References

⭐ **contrastive learning**: pushes positive pairs together, negatives apart
⭐ **false negatives**: can be partly removed without supervision
⭐ **not all negatives are created equal**: harder negatives are better
⭐ **looking forward**: making SSL as easy as supervised learning?

Reading material


Blog posts


[https://lilianweng.github.io/lil-log/2021/05/31/contrastive-representation-learning.html](https://lilianweng.github.io/lil-log/2021/05/31/contrastive-representation-learning.html)