MAC5921 – Deep Learning

Aula 16 – 19/10/2023

Self-supervised learning

Nina S. T. Hirata

Estes slides: mix de partes de tutoriais recentes em eventos de qualidade

- NeurIPS 2021 – Tutorial Self-Supervised Learning Self-Prediction and Contrastive Learning

  https://nips.cc/media/neurips-2021/Slides/21895.pdf

- ICML 2023 Self-Supervised Learning in Vision: from Research Advances to Best Practices

  https://icml.cc/virtual/2023/tutorial/21552

  https://icml.cc/media/icml-2023/Slides/21552.pdf

- http://cs231n.stanford.edu/slides/2022/lecture_14_jiajun.pdf

- CVPR 2021 Tutorial on Leave Those Nets Alone: Advances in Self-Supervised Learning

  https://gidariss.github.io/self-supervised-learning-cvpr2021/

- https://project.inria.fr/paiss/files/2018/07/zisserman-self-supervised.pdf

Self-supervised learning

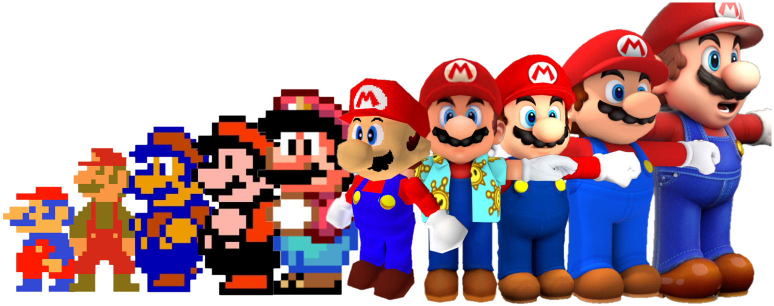
Motivação

- Temos uma abundância de dados
- Treinamento de algoritmos de ML requer dados rotulados
- Rotular dados é trabalhoso, caro
- *Self-supervised learning*: Ideia central é aprender a partir de dados não rotulados
- Informação aprendida dessa forma pode ser aproveitada/transferida para diferentes tarefas

Lembrar que as coisas mudam ao longo do tempo

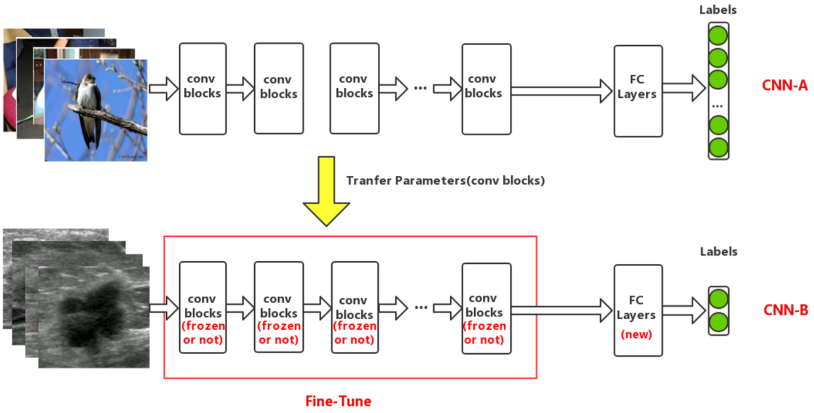
Logo, o que valia antes pode não valer mais

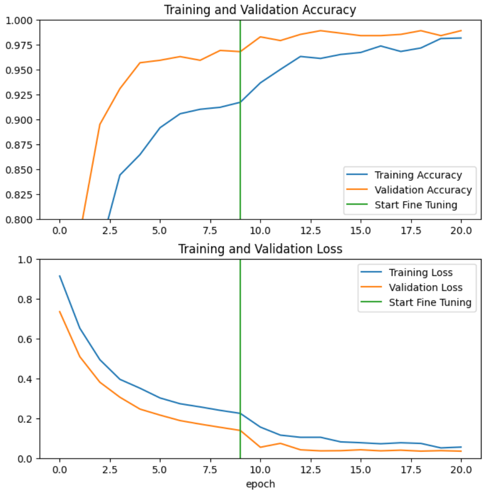
Rotular tudo de novo? NÃO!



Antes, porém, um parênteses

Transfer Learning & Fine-tuning


Labels
conv blocks
conv blocks
conv blocks
...
conv blocks
FC Layers
CNN-A
Tranfer Parameters(conv blocks)
Labels
conv blocks (frozen or not)
conv blocks (frozen or not)
conv blocks (frozen or not)
...
conv blocks (frozen or not)
FC Layers (new)
CNN-B
Fine-Tune



1. Treinar as novas camadas FC, mantendo a parte pré-treinada congelada

2. Descongelar as últimas camadas da parte pré-treinada e fazer o fine-tuning (usar lr menor)

Self-supervised learning

Como aprender a partir de dados não rotulados?

Sabemos que os dados rotulados são importantes para calcular a perda e a partir dela otimizar os parâmetros do modelo

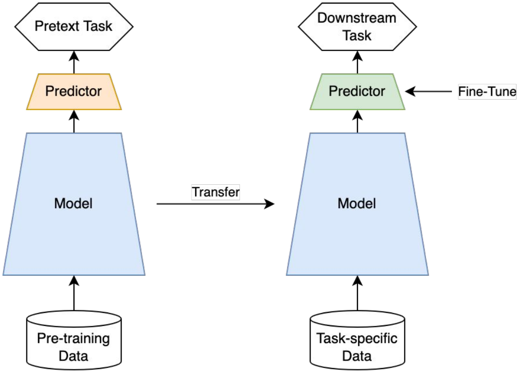
Essa otimização "guiada" é que leva o modelo a "aprender"

Abordagem: vamos então criar tarefas que funcionem como **pretexto**, um guia para o processo de aprendizado

Recap: Pretext Tasks

Step 1: Pre-train a model for a pretext task

Step 2: Transfer to applications



Já vimos tarefas pretexto!

- No caso de RNNs, treinamos ela para predizer o próximo elemento da sequência.

  Por exemplo, em NLP, textos existem aos montes, sem custo.

- No caso do GPT, idem

- No caso do BERT, vimos *masking*

O que GPT e/ou BERT aprendem?

- GPT – aprende a gerar texto

  Ou seja, a distribuição dos dados ($p(\mathbf{x})$)

- BERT – aprende uma representação dos dados

  que possa ser útil para diferentes *downstream tasks*

  (masked auto-encoding)

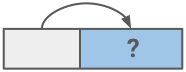
Tipos de ***self-supervised tasks***

- Self-prediction — intra-sample
- Contrastive-learning — inter-sample

Methods for Framing Self-Supervised Learning Tasks

Self-prediction: Given an individual data sample, the task is to predict one part of the sample given the other part.

The part to be predicted pretends to be missing.

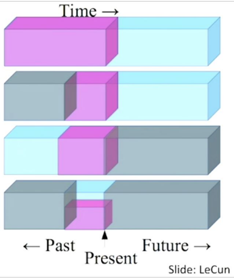


“Intra-sample” prediction

Self-Prediction

Self-prediction construct prediction tasks within every individual data sample: to predict a part of the data from the rest while pretending we don't know that part.

- ▶ **Predict any part of the input from any other part.**
- ▶ **Predict the future from the past.**
- ▶ **Predict the future from the recent past.**
- ▶ **Predict the past from the present.**
- ▶ **Predict the top from the bottom.**
- ▶ **Predict the occluded from the visible**
- ▶ **Pretend there is a part of the input you don't know and predict that.**

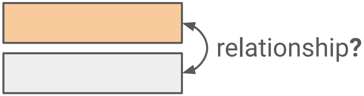


(Famous illustration from Yann LeCun)

Methods for Framing Self-Supervised Learning Tasks

Contrastive learning: Given multiple data samples, the task is to predict the relationship among them.

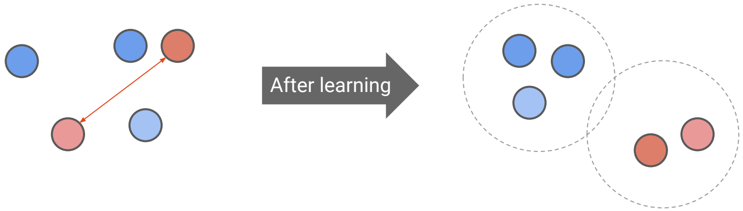
The multiple samples can be selected from the dataset based on some known logics (e.g. the order of words / sentences), or fabricated by altering the original version.



“Inter-sample” prediction

Contrastive Learning

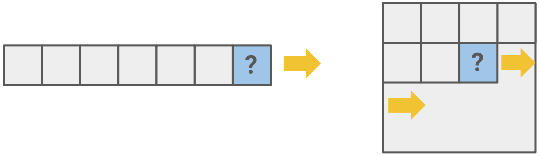
The goal of contrastive representation learning is to learn such an embedding space in which *similar* sample pairs stay *close* to each other while *dissimilar* ones are *far apart*.



Métodos de *self-prediction*

Self-Prediction: Autoregressive Generation

The autoregressive model predicts future behavior based on past behavior. Any data that comes with an innate sequential order can be modeled with regression.

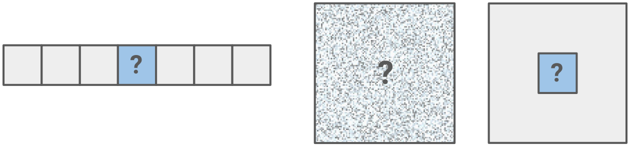


Examples:

- Audio (WaveNet, WaveRNN)
- Autoregressive language modeling (GPT, XLNet)
- Images in raster scan (PixelCNN, PixelRNN, iGPT)

Self-Prediction: Masked Generation

We mask a random portion of information and pretend it is missing, irrespective of the natural sequence. The model learns to predict the missing portion given other unmasked information.

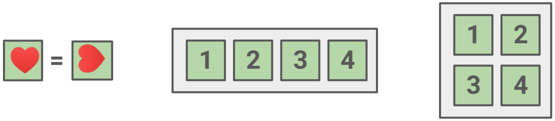


Examples:

- Masked language modeling (BERT)
- Images with masked patch (denoising autoencoder, context autoencoder, colorization)

Self-Prediction: Innate Relationship Prediction

Some transformation (e.g. segmentation, rotation) of one data sample should maintain the original information or follow the desired innate logic.



Examples:

- Order of image patches (e.g., relative position, jigsaw puzzle)
- Image rotation
- Counting features across patches

Contrastive learning

Inter-sample classification

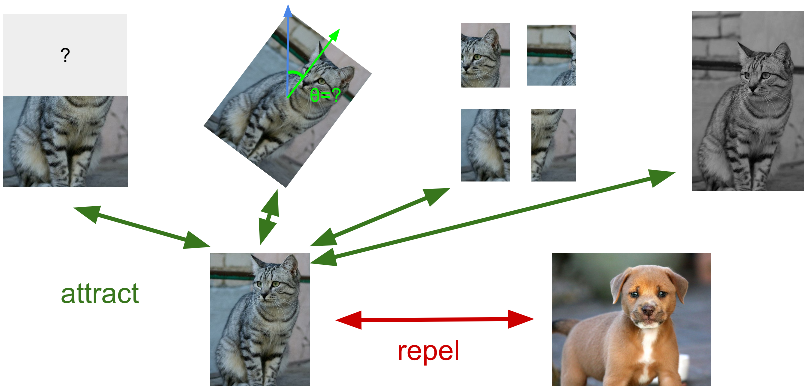
Contrastive Learning: Inter-Sample Classification

Given both similar (“positive”) and dissimilar (“negative”) candidates, to identify which ones are similar to the anchor data point is a *classification* task.

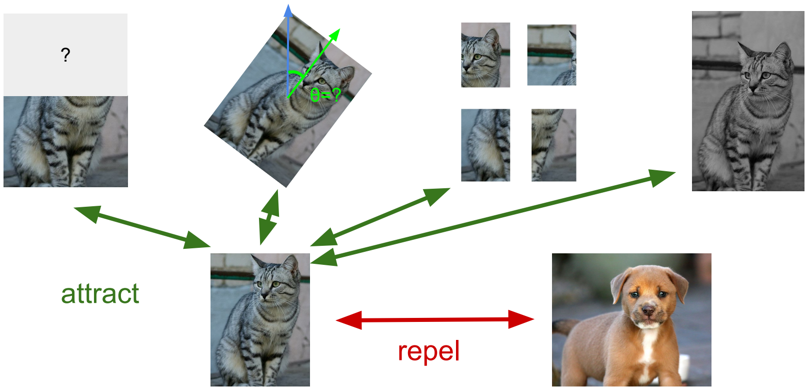
There are creative ways to construct a set of data point candidates:

1. The original input and its distorted version
2. Data that captures the same target from different views

Contrastive Representation Learning



Contrastive Representation Learning



Contrastive Learning: Inter-Sample Classification

Common loss functions:

- Contrastive loss (Chopra et al. 2005)
- Triplet loss (Schroff et al. 2015; FaceNet)
- Lifted structured loss (Song et al. 2015)
- Multi-class n-pair loss (Sohn 2016)
- Noise contrastive estimation (“NCE”; Gutmann & Hyvarinen 2010)
- InfoNCE (van den Oord, et al. 2018)
- Soft-nearest neighbors loss (Salakhutdinov & Hinton 2007, Frosst et al. 2019)

Contrastive Learning: Inter-Sample Classification

Contrastive loss (Chopra et al. 2005): Works with labelled dataset.

Encodes data into an embedding vector such that examples from the same class have similar embeddings and samples from different classes have different ones.

Given two labeled data pairs $(\mathbf{x}_i, y_i)$ and $(\mathbf{x}_j, y_j)$:

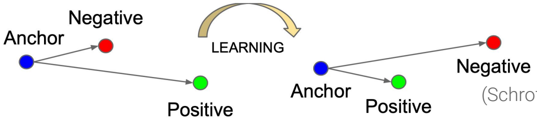
$$\mathcal{L}_{\text{cont}}(\mathbf{x}_i, \mathbf{x}_j, \theta) = \mathbb{1}[y_i = y_j] \underbrace{\boxed{\|f_\theta(\mathbf{x}_i) - f_\theta(\mathbf{x}_j)\|_2^2}}_{\text{minimize}} + \mathbb{1}[y_i \neq y_j] \max(0, \epsilon - \underbrace{\boxed{\|f_\theta(\mathbf{x}_i) - f_\theta(\mathbf{x}_j)\|_2}}_{\text{maximize}})^2$$

Contrastive Learning: Inter-Sample Classification

Triplet loss (Schroff et al. 2015): learns to minimize the distance between the anchor **x** and positive **x+** and maximize the distance between the anchor **x** and negative **x-** at the same time.

Given a triplet input $(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-)$,

$$\mathcal{L}_{\text{triplet}}(\mathbf{x}, \mathbf{x}^+, \mathbf{x}^-) = \sum_{\mathbf{x}\in\mathcal{X}} \max\left(0, \|f(\mathbf{x}) - f(\mathbf{x}^+)\|_2^2 - \|f(\mathbf{x}) - f(\mathbf{x}^-)\|_2^2 + \epsilon\right)$$



(Schroff et al. 2015)

Contrastive Learning: Inter-Sample Classification

N-pair loss (Sohn 2016) generalizes triplet loss to include comparison with multiple negative samples.

Given one positive and N-1 negative samples, $\{\mathbf{x}, \mathbf{x}^+, \mathbf{x}_1^-, \dots, \mathbf{x}_{N-1}^-\}$

$$\begin{aligned}\mathcal{L}_{\text{N-pair}}(\mathbf{x}, \mathbf{x}^+, \{\mathbf{x}_i^-\}_{i=1}^{N-1}) &= \log\Big(1 + \sum_{i=1}^{N-1} \exp(f(\mathbf{x})^\top f(\mathbf{x}_i^-) - f(\mathbf{x})^\top f(\mathbf{x}^+))\Big) \\ &= -\log \frac{\exp(f(\mathbf{x})^\top f(\mathbf{x}^+))}{\exp(f(\mathbf{x})^\top f(\mathbf{x}^+)) + \sum_{i=1}^{N-1} \exp(f(\mathbf{x})^\top f(\mathbf{x}_i^-))}\end{aligned}$$

Contrastive Learning: Inter-Sample Classification

Lifted structured loss (Song et al. 2015): utilizes all the pairwise edges within one training batch for better computational efficiency.

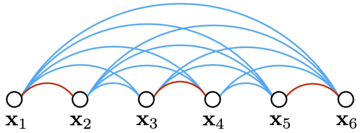
$$\mathcal{L}_{\text{struct}}^{(ij)} = D_{ij} + \log \Big(\sum_{(i,k)\in\mathcal{N}} \exp(\epsilon - D_{ik}) + \sum_{(j,l)\in\mathcal{N}} \exp(\epsilon - D_{jl}) \Big)$$

where $D_{ij} = \|f(\mathbf{x}_i) - f(\mathbf{x}_j)\|_2$

$(i, j) \in \mathcal{P}$

$\mathcal{P}$ set of positive pairs

$\mathcal{N}$ set of negative pairs



(Song et al. 2015)

Contrastive Learning: Inter-Sample Classification

Noise Contrastive Estimation (NCE) (Gutmann & Hyvarinen 2010) runs logistic regression to tell apart the target data from noise.

Given target sample distribution p and noise distribution q,

$$\mathcal{L}_{\text{NCE}} = -\frac{1}{N}\sum_{i=1}^{N}\Big[\log \sigma(\ell_\theta(\mathbf{x}_i)) + \log(1 - \sigma(\ell_\theta(\tilde{\mathbf{x}}_i)))\Big] \longleftarrow \text{just cross entropy}$$

where logit $\ell_\theta(\mathbf{u}) = \log \frac{p_\theta(\mathbf{u})}{q(\mathbf{u})} = \log p_\theta(\mathbf{u}) - \log q(\mathbf{u})$

sigmoid $\sigma(\ell) = \frac{1}{1 + \exp(-\ell)} = \frac{p_\theta}{p_\theta + q}$

Contrastive Learning: Inter-Sample Classification

InfoNCE (van den Oord, et al. 2018): uses categorical cross-entropy loss to identify the positive sample amongst a set of unrelated noise samples.

Given a context vector c, the positive sample should be drawn from the conditional distribution p(x|c), while N−1 negative samples are drawn from the proposal distribution p(x), independent from the context c.

The probability of detecting the positive sample correctly is:

$$p(C = \mathbf{pos} | X, \mathbf{c}) = \frac{f(\mathbf{x}_{\text{pos}}, \mathbf{c})}{\sum_{j=1}^{N} f(\mathbf{x}_j, \mathbf{c})}$$ where the density function is $$f(\mathbf{x}, \mathbf{c}) \propto \frac{p(\mathbf{x}|\mathbf{c})}{p(\mathbf{x})}$$

Contrastive Learning: Inter-Sample Classification

Soft-Nearest Neighbors Loss (Frosst et al. 2019) extends the loss function to include multiple positive samples given known labels.

Given a batch of samples $\{\mathbf{x}_i, y_i\}_{i=1}^{B}$,

$$\mathcal{L}_{\text{snn}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\sum_{i\neq j, y_i = y_j, j=1,\dots,B} \exp(-f(\mathbf{x}_i, \mathbf{x}_j)/\overset{\text{temperature term}}{\tau})}{\sum_{i\neq k, k=1,\dots,B} \exp(-f(\mathbf{x}_i, \mathbf{x}_k)/\tau)}$$

Hard Negative Mining

Hard negative samples are different to learn. They should have different labels from the anchor sample, but the embedding features may be very close.

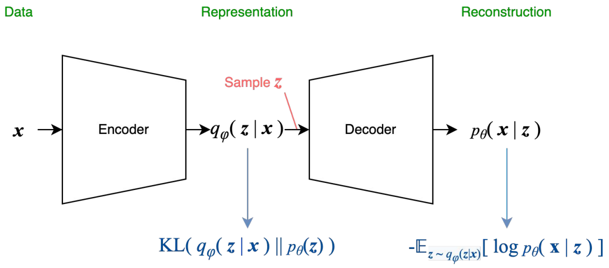
Hard negative mining is important for contrastive learning.

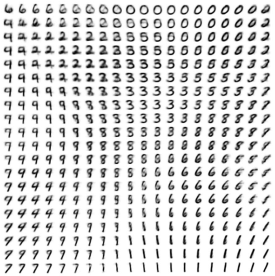
Challenging negative samples encourages the model to learn better representations that can distinguish hard negatives from true positives.

Pretext tasks

Image Pretext Tasks: Variational Autoencoders

Auto-Encoding Variational Bayes (Kingma et al. 2014)

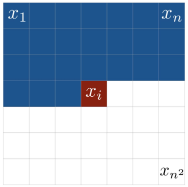

Data
Representation
Reconstruction
$\boldsymbol{x}$
Encoder
$q_\varphi(\boldsymbol{z} \mid \boldsymbol{x})$
Sample $\boldsymbol{z}$
Decoder
$p_\theta(\boldsymbol{x} \mid \boldsymbol{z})$
$\mathrm{KL}(q_\varphi(\boldsymbol{z} \mid \boldsymbol{x}) \| p_\theta(\boldsymbol{z}))$
$-\mathbb{E}_{z \sim q_\varphi(z|x)}[\log p_\theta(\mathbf{x} \mid \boldsymbol{z})]$



(Kingma et al. 2014)

Vision Pretext Tasks: Autoregressive Image Generation

- Neural autoregressive density estimation (NADE; Larochelle et al. 2011)
- PixelRNN, PixelCNN (Oord et al. 2016)
- Image GPT (Chen et al. 2020)



Raster scan order

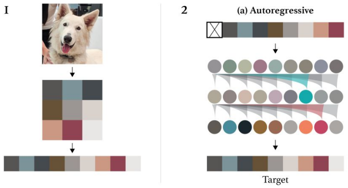
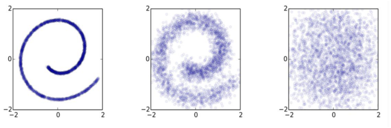


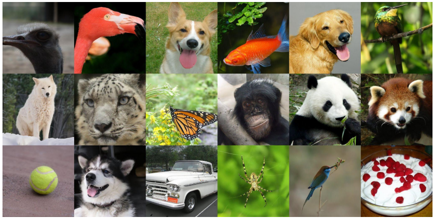
Image GPT (Chen et al. 2020)

Vision Pretext Tasks: ~~Autoregressive~~ Image Generation

Diffusion modeling: Follows a Markov chain of diffusion steps to slowly add random noise to data and then learn to reverse the diffusion process to construct desired data samples from the noise. (Sohl-Dickstein et al 2015; Yang & Ermon 2019; Ho et al. 2020; Dhariwal & Nichol 2021)



Diffusion modeling

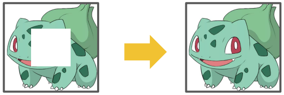


(Dhariwal & Nichol 2021)

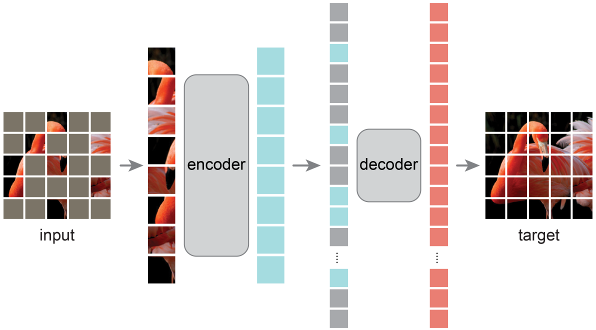
Vision Pretext Tasks: Masked Prediction

- **Denoising autoencoder** (Vincent et al. 2008)
  - Add noise = Randomly mask some pixels
  - Only reconstruction loss

- **Context autoencoder** (Pathak et al. 2016)
  - Mask a random region in the image
  - Reconstruction loss + adversarial loss

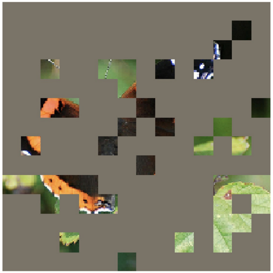


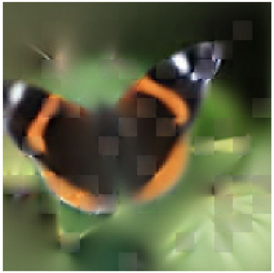
How MAE Works?

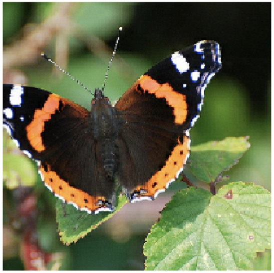


Reconstruct

MAE Reconstruction Example

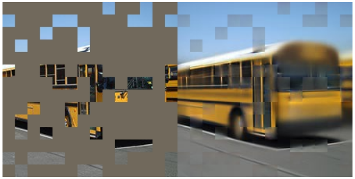

Masked input: 80%


MAE's guess

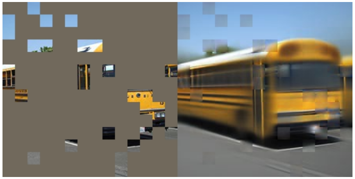

Ground truth

MAE Can Generalize

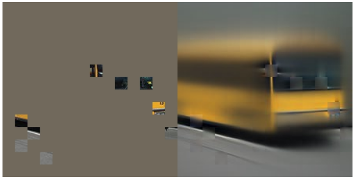
original



75% mask



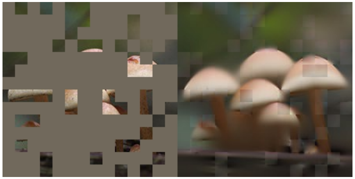
85% mask

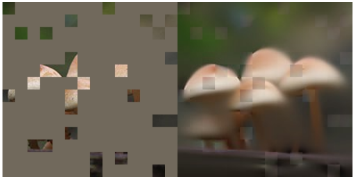


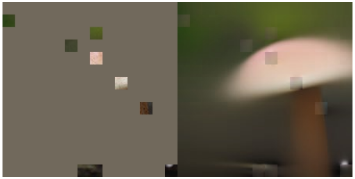
95% mask

MAE Can Generalize

original

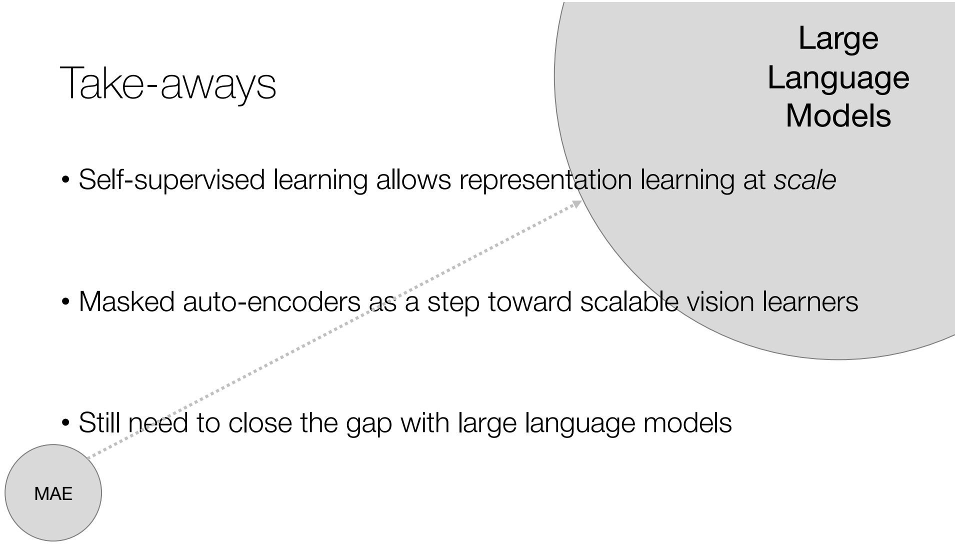

75% mask


85% mask


95% mask

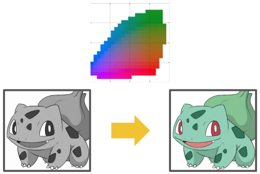
Take-aways

- Self-supervised learning allows representation learning at *scale*
- Masked auto-encoders as a step toward scalable vision learners
- Still need to close the gap with large language models



Vision Pretext Tasks: Colorization and More

- **Colorization** (Zhang et al. 2016)
  - Predict the binned CIE Lab color space given a grayscale image.



- **Split-brain autoencoder** (Zhang et al. 2017)
  - Predict a subset of color channels from the rest of channels.
  - Channels: luminosity, color, depth, etc.

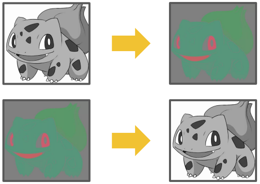
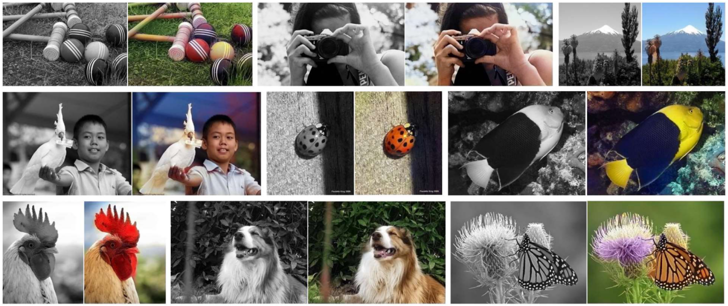


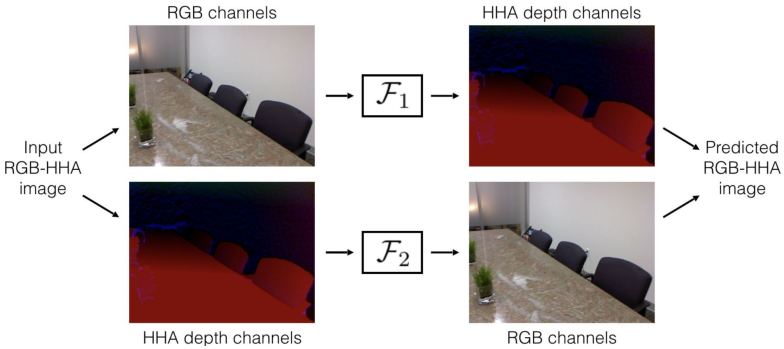
Image example II: colourization

Train network to predict pixel colour from a monochrome input



Colorful Image Colorization, Zhang *et al*., ECCV 2016

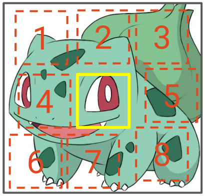
Learning features from colorization: Split-brain Autoencoder



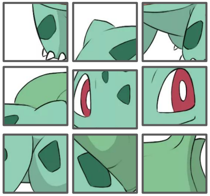
Source: Richard Zhang / Phillip Isola

Vision Pretext Tasks: Innate Relationship Prediction

- Learn the relationship among image patches:
    - Predict **relative positions** between patches (Doersch et al 2015)
    - **Jigsaw puzzle** using patches (Noroozi & Favaro 2016)



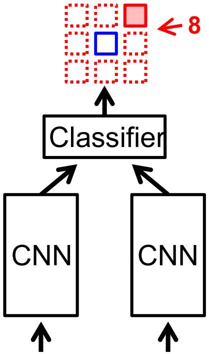
Given a patch, predict which one of 8 neighboring locations another patch is in

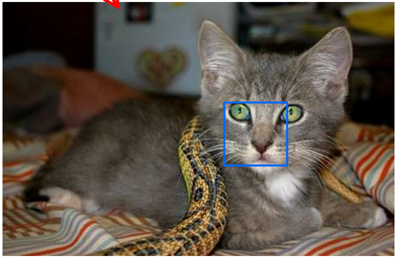


Output a probability vector per patch index out of a predefined set of permutations

Example: relative positioning

Train network to predict relative position of two regions in the same image

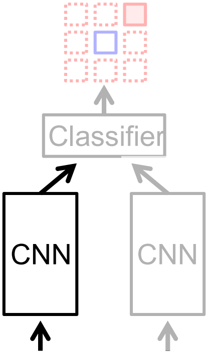


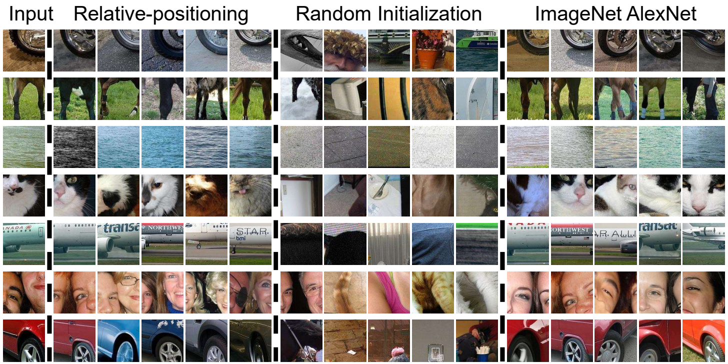


Randomly Sample Patch
Sample Second Patch

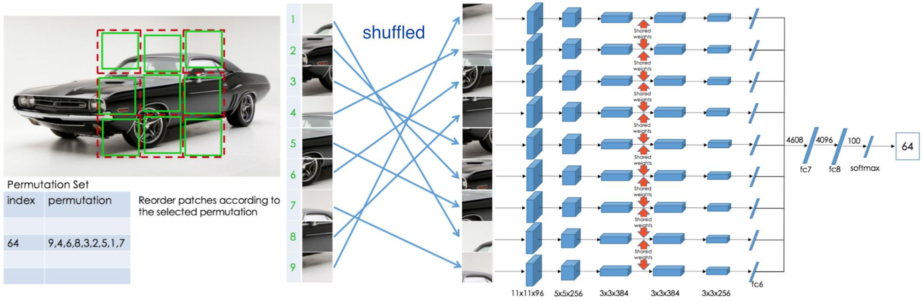
Unsupervised visual representation learning by context prediction,
Carl Doersch, Abhinav Gupta, Alexei A. Efros, ICCV 2015

What is learned?





Pretext task: solving “jigsaw puzzles”

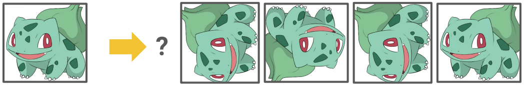


index	permutation
64	9,4,6,8,3,2,5,1,7

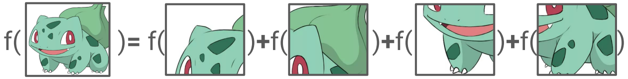
(Image source: Noroozi & Favaro, 2016)

Vision Pretext Tasks: Innate Relationship Prediction

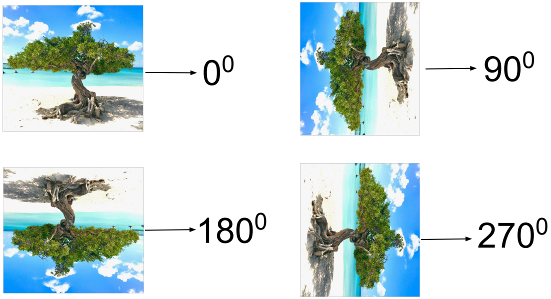
- RotNet: predict **which rotation** is applied (Gidaris et al. 2018)
    - Rotation does not alter the semantic content of an image.



- Representation Learning by Learning to Count (Noroozi et al. 2017)
    - Counting features across patches without labels, using equivariance of counts



Type of hidden data/property



Input: image rotated by [0, 90, 180, 270]

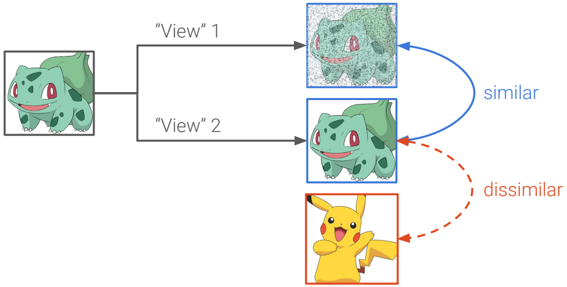
Output: 4-way classification

Gidaris et al., 2018, Predicting Image Rotations

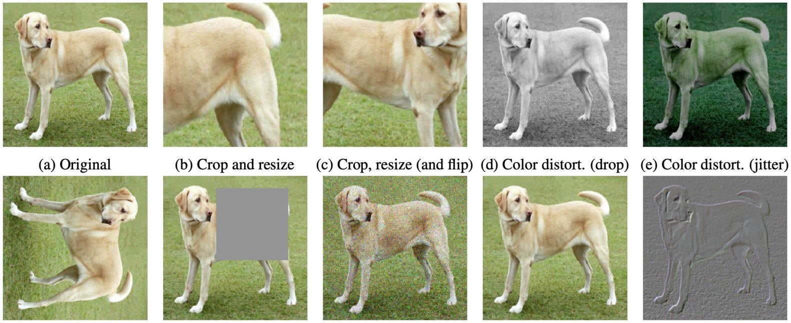
Abordagens contrastivas

Vision Pretext Tasks: Contrastive Learning

The common approach is to make multiple views (e.g. data augmentation) to one image and consider the image and its distorted version as similar pairs, while different images are treated dissimilar.



SimCLR: generating positive samples from data augmentation

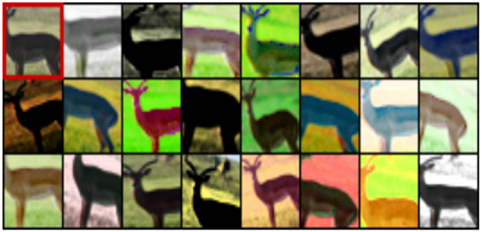

(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Source: Chen et al., 2020

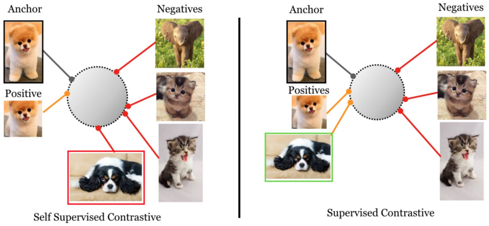
Image example III: exemplar networks

- Exemplar Networks (Dosovitskiy *et al*., 2014)
- Perturb/distort image patches, e.g. by cropping and affine transformations
- Train to classify these exemplars as same class



Vision Pretext Tasks: Combining with Supervised Loss

- Combine supervised loss + self-supervised learning
  - Self-supervised semi-supervised learning (**S4L**; Zhai et al 2019)
  - Unsupervised data augmentation (**UDA**; Xie et al 2019)
- Use known labels for contrastive learning
  - **Supervised Contrastive Loss** (SupCon; Khosla et al. 2021)



(Khosla et al. 2021)

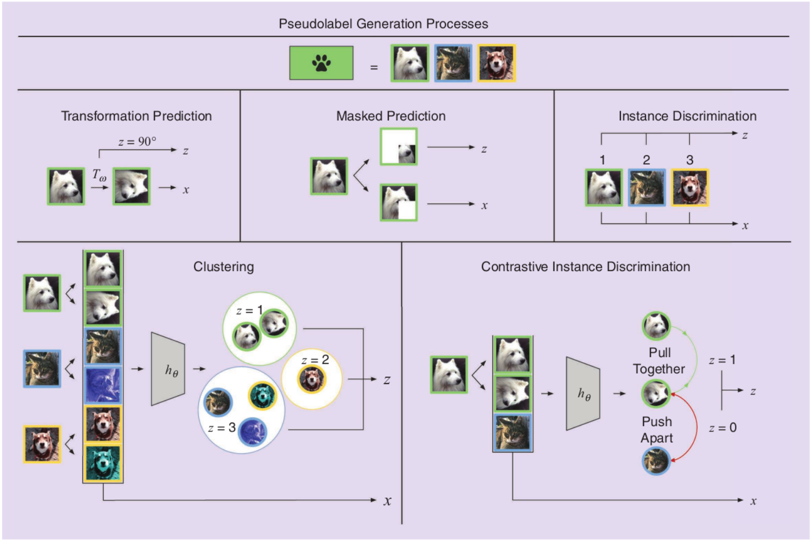


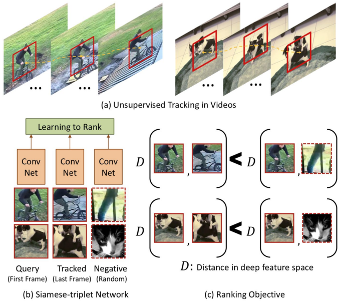
FIGURE 3. Illustrative examples of the way pseudolabels are generated in the four families of pretext tasks of our taxonomy: TP, masked prediction, instance discrimination, and clustering. An additional depiction is included of the popular version of instance discrimination using contrastive losses. The squares represent inputs x, while circles portray the feature vectors of those inputs, $h_\theta(x)$.

Vídeos

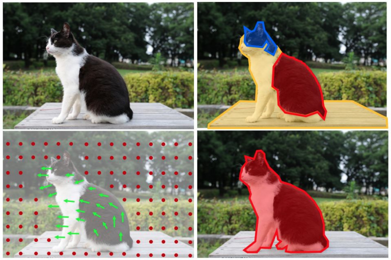
Video Pretext Tasks: Optical Flow

Tracking object movement tracking in time

- Tracking movement of image patches (Wang & Gupta 2016)

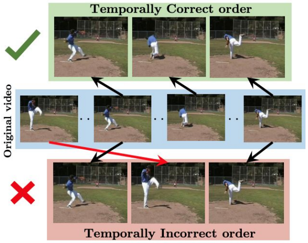


- Segmenting based on motion (Pathak et al. 2017)

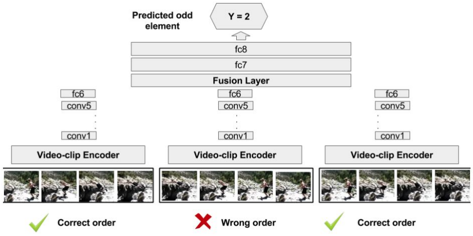


Video Pretext Tasks: Sequence Ordering

- Temporal order verification (Misra et al. 2016, Fernando et al. 2017)



(Misra et al. 2016)

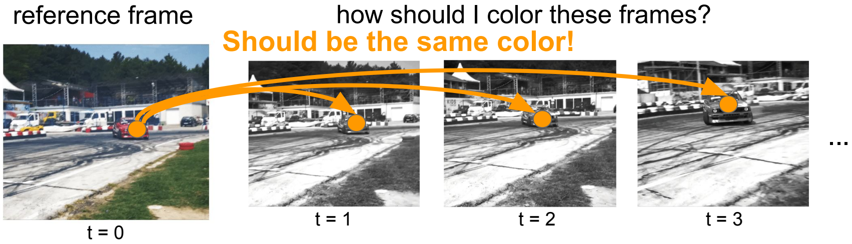


(Fernando et al. 2017)

- Predict *the arrow of time*, forward or backward (Wei et al. 2018)

Pretext task: video coloring

Idea: model the *temporal coherence* of colors in videos



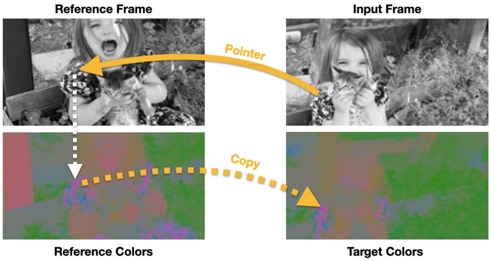
Hypothesis: learning to color video frames should allow model to learn to track regions or objects without labels!

Source: Vondrick et al., 2018

Video Pretext Tasks: Colorization

Tracking emerges by colorizing videos (Vondrick et al. 2018)

- Copy colors from a reference frame to another target frame in grayscale by leveraging the natural temporal coherence of colors across video frames.

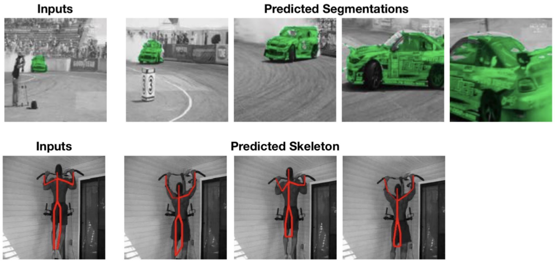


(Vondrick et al. 2018)

Video Pretext Tasks: Colorization

Tracking emerges by colorizing videos (Vondrick et al. 2018)

- Used for video segmentation or human pose estimation without fine-tuning!



(Vondrick et al. 2018)

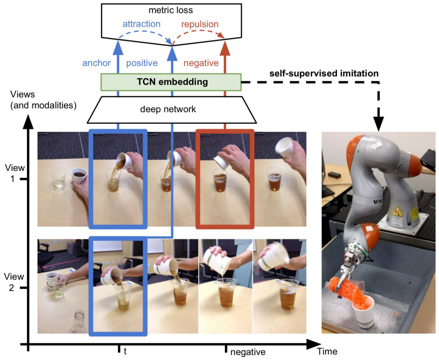
Video Pretext Tasks: Contrastive Multi-View Learning

TCN (Sermanet et al. 2017)

- Use triplet loss
- Different viewpoints at the same timestep of the same scene should share the same embedding, while embedding should vary in time, even of the same camera viewpoint.

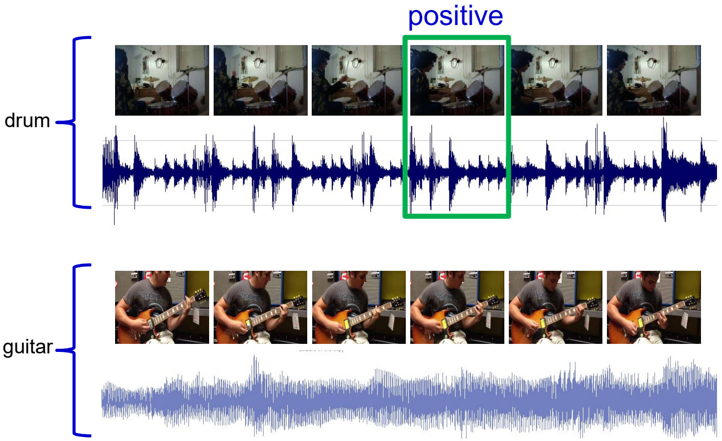
Multi-frame TCN (Dwibedi et al. 2019)

- Use n-pairs loss
- Multiple frames are aggregated into one embedding.

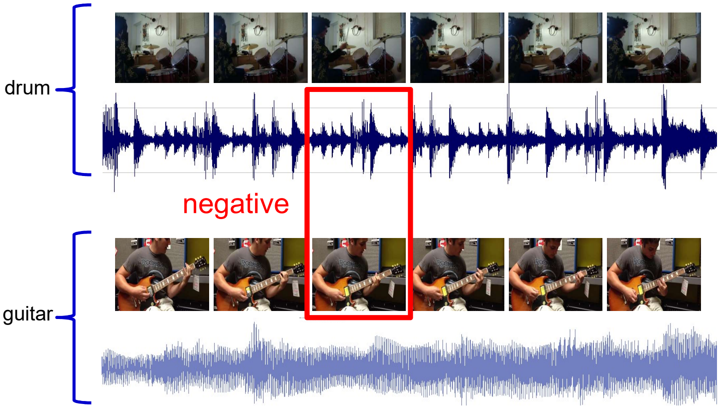


(Sermanet et al. 2017)

Audio-Visual Correspondence

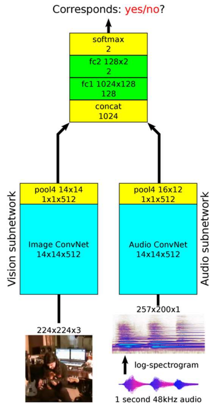


Audio-Visual Correspondence

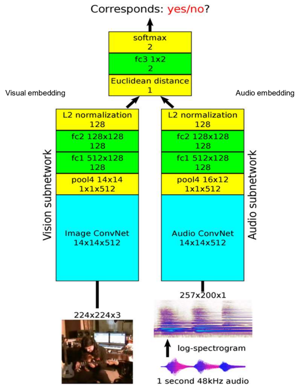


To embed or not to embed?

Concatenation



Embedding



Features available

Cross-modal alignment in embedding