The SimCLR Model

Nguyen Thai Hoc[1,2]

[1]University of Information and Communication Technology, Thai Nguyen
[2]Institute of Applied Science and Technology,
University of Information and Communication Technology, Thai Nguyen

Outlines

Introduction to the SimCLR model

* SimCLR was first (SimCLR v1) introduced in February 2020 and was presented in the research paper titled ”A Simple Framework for Contrastive Learning of Visual Representations”.
* An improved version (SimCLR v2) was released in June 2020.

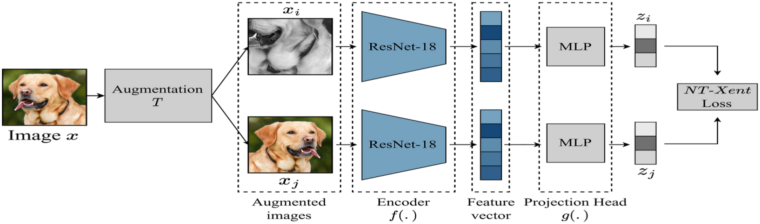
Problems

- Eliminates dependence on labeled data
- Leverages data augmentation effectively
- Flexible and adaptable

Motivation of SimCLR

1. Agreement under transformations (Becker & Hinton 1992)
2. Handcrafted pretext tasks
   - Relative path prediction (Doersch, 2015)
   - Jigsaw puzzles (Noroozi and Favaro, 2016)
   - Colorization (Zhang, 2016)
   - Rotation prediction (Gidaris, 2018)
3. Contracstive visual representation learning
   - A key concepts in SimCLR, dates back to (Hadsell 2006). Learn representations by constrasting "positive pairs"against "negative pairs"
   - Introduced a "memory bank"to store feature vectors for contrasting (Wu 2018)

Architecture



- Data augmentation: x -> (x_i, x_j)
- Base encoder: f(.) -> extracts representation (average pooling layer)
- Projection head. g(.) maps representation to the space latent, where contrastive loss is applied.
- Contrastive loss function (NT Xent loss): x_k including a positive pair (x_i, x_j). Aims to identify $x_j \in x_k$ with $k \neq i$

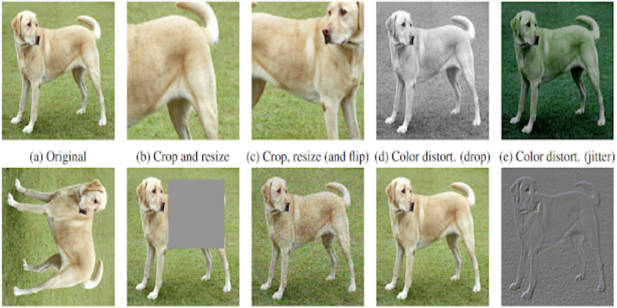
Training PipeLine - SimCLR Pseudocode

- Data augmentaion
- Feature extraction and mapped to the latent space using projection head
- Computer the costracstive loss
- Update the networks f(.) and g(.) to minimize $\mathcal{L}$. Retain f(.) and discard g(.).

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of f, g, $\mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^N$ **do**
  for all $k \in \{1, \dots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}$, $t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \dots, 2N\}$ and $j \in \{1, \dots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

Traning PipeLine - Data Augmentation

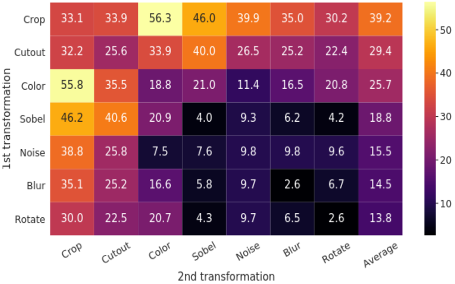


(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

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

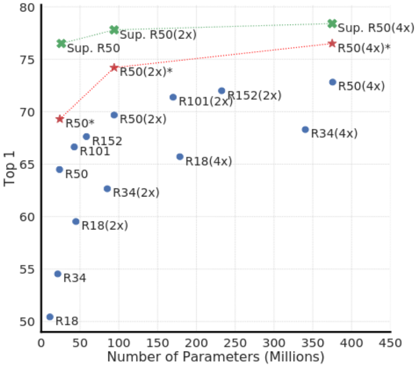
Training PipeLine - Data Augmentation

- Data augmentation defines predictive task
- Composition of data augmentation operations is crucial for learning good representations
- Contrastive learning needs stronger data augmentation than supervised learning



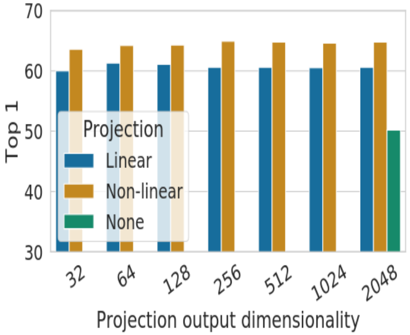
Training PipeLine - Base Encoder

- Unsupervised learning benefits more from bigger models than its supervised counterpart.



Training PipeLine - Projection Head

- Non-linear projection head improves the representations quality of the layer before it: None vs linear vs non-linear
- Choose the projection head output dimensionnality



Training PipeLine - NT Xent Loss

(+) Logistic loss:

$$L(y, \hat{y}) = -y \log(\hat{y}) - (1 - y) \log(1 - \hat{y})$$

(+) Margin loss:

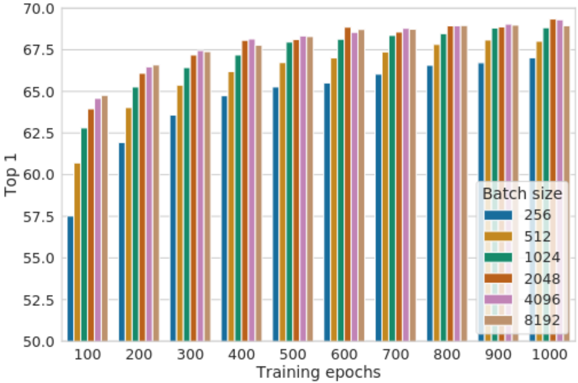
$$\mathcal{L}(y, D) = \frac{1}{2}(y \cdot D^2 + (1 - y) \cdot \max(0, m - D^2))$$

(+) NT Xent loss:

$$L_{i,j} = -\log \frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} 1_{k \neq i} \exp(\mathrm{sim}(z_i, z_k)/\tau)}$$

Training PipeLine - Batch Sizes

- Benefits from larger batch sizes and more training steps compared to supervised leanring.



Training PipeLine - Linear Protocol Evaluation

Linear protocol evaluation is a common method to evaluate the representations of a model

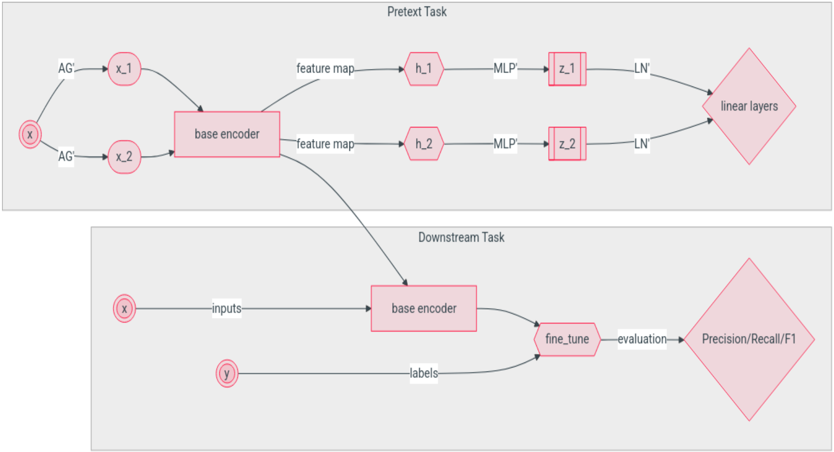
(+) Workflow:

- Add a classifier layer basic such as fully connected layer (FC) or linear classifier
- Fine-tuning with labeled data

(+) Objectives:

- Evaluating the quality of the model's representations ensures that a good model will perform well even when only a simple classifier layer is used

Experimental Setup - Training Model Process



Experimental Setup - Implementation Plan

(+) Task 01: Pretext Task

- S1: Data augmentation: Random crop resize, color distortion, gaussian blur.
- S2: InceptionV3, InceptionResNetV2, Xception, ResNet50.
- S3: Output the projection head dimensionality 32/64/128.
- S4: Batch size 32/64/128, Epochs ?
- S5: Evaluation Linear Protocol (metrics)

(+) Task 02: Downstream Task

- S1: Choose the models with the best performance after being evaluated in task 1, then fine-tuning these on labeled dataset for classification task (5 class).
- S2: Evaluate the model's performance using classification metrics such as Precicion, Recall and F1-score.

Results & Comparisons

(+) Datasets

- Pre-trained: 15703
- Evaluation Linear Protocol / Fine-tune: 3365
- Test after fine-tuned: 3366

Name / Backbone	Unlabeled Data	Evaluation Linear Protocol	Fine-tune	Projection Head / Batch Size	F1 score
InceptionV3	15.000	3000	3000	128	**72.57** / **67.45**
Inception ResNet V2	15.000	3000	3000	128	**74.04** / **66.20**
Xception	15.000	3000	3000	128	**67.18** / **71.21**
ResNet50	15.000	3000	3000	128	**65.14**

References

- (1) A Simple Framework for Contrastive Learning for Visual Representations
- (2) Big Self-Supervised Models are Strong Semi-Supervised Learners

Thank you for your attention. I look forward to your thoughts and feedback !