THE UNIVERSITY of EDINBURGH

SpeechWave

KING'S College LONDON

Contrastive Representation Learning

Erfan Loweimi

Centre for Speech Technology Research (CSTR)
University of Edinburgh

Outline

- Contrastive Learning
- Unsupervised Contrastive Learning
  - CPC
  - SimCLR
- Supervised Contrastive Learning
- Conclusion

Self-Supervised Learning (SSL)

- **Goal**: Learning universal transferable representation
- **Paradigms**:
  - *Generative*
    - Focus on sample-level reconstruction + Independent assumption
    - Lower ability in modelling correlation & structure
  - *Contrastive*
    - Learn by contrasting *positive* & *negatives* in a *latent space*
  - ...

Contrastive Learning

- Learn an encoder, *f(x)*, such that ...

$$\text{score}(f(x), f(x^+)) \gg \text{score}(f(x), f(x^-))$$

  - *x*: **A**nchor (reference or baseline)
  - x^+: **P**ositive (similar) → data augmentation (view)
  - x^-: **N**egative (dissimilar) → sampling (???)
  - *Score*: a similarity/agreement measure

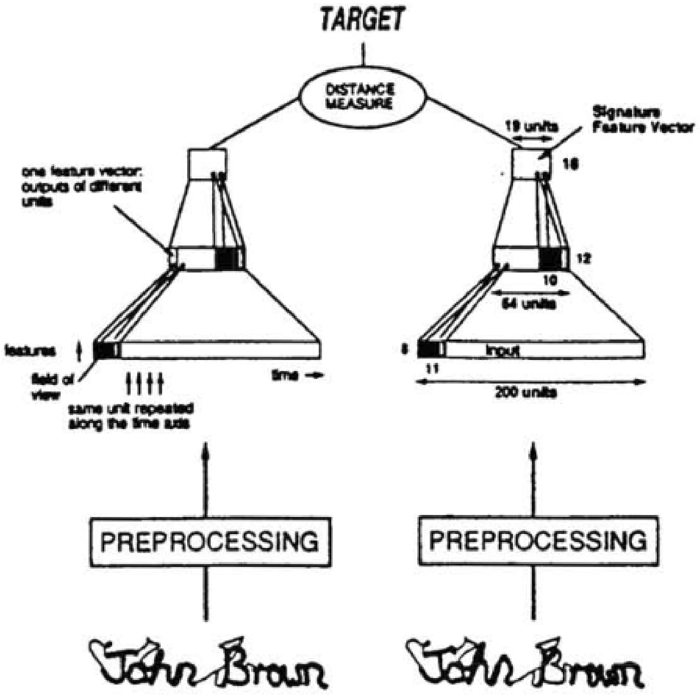
- Contrastive Loss ... *distance-based* (*metric learning*) ... NOT *error-prediction*

Siamese Neural Network (SNN)

Advances in Neural Information Processing, 1994

Signature Verification using a "Siamese" Time Delay Neural Network

Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger and Roopak Shah
AT&T Bell Laboratories
Holmdel, NJ 07733
jbromley@big.att.com

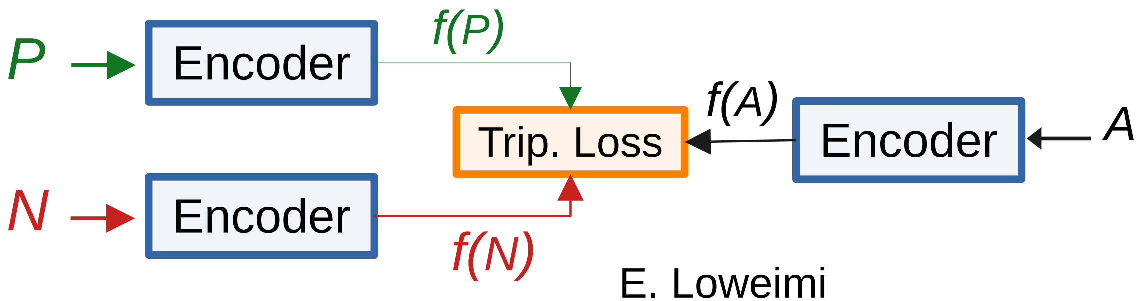


Siamese Neural Network (SNN)

- Returns embedding (similar ↔ close), NOT $p(y|x)$
- Contains two identical subnets (encoder)
- Loss: $L = L^+ + L^-$ or *Triplet* loss: $L(A,P,N)$
- Hard negative mining required ($f(N)$ nearby $f(A)$)

α: margin

$$\mathcal{L}_{\text{Triplet}}(A,P,N) = \max(\| f(A) - f(P) \|^2 - \| f(A) - f(N) \|^2 + \alpha, 0)$$



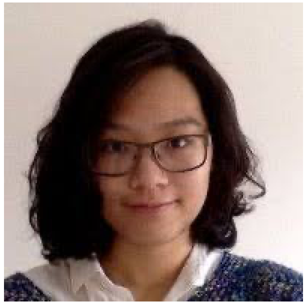
NIPS 2018

Representation Learning with Contrastive Predictive Coding

Aaron van den Oord
DeepMind
avdnoord@google.com

Yazhe Li
DeepMind
yazhe@google.com

Oriol Vinyals
DeepMind
vinyals@google.com



Contrastive Predictive Coding (CPC)

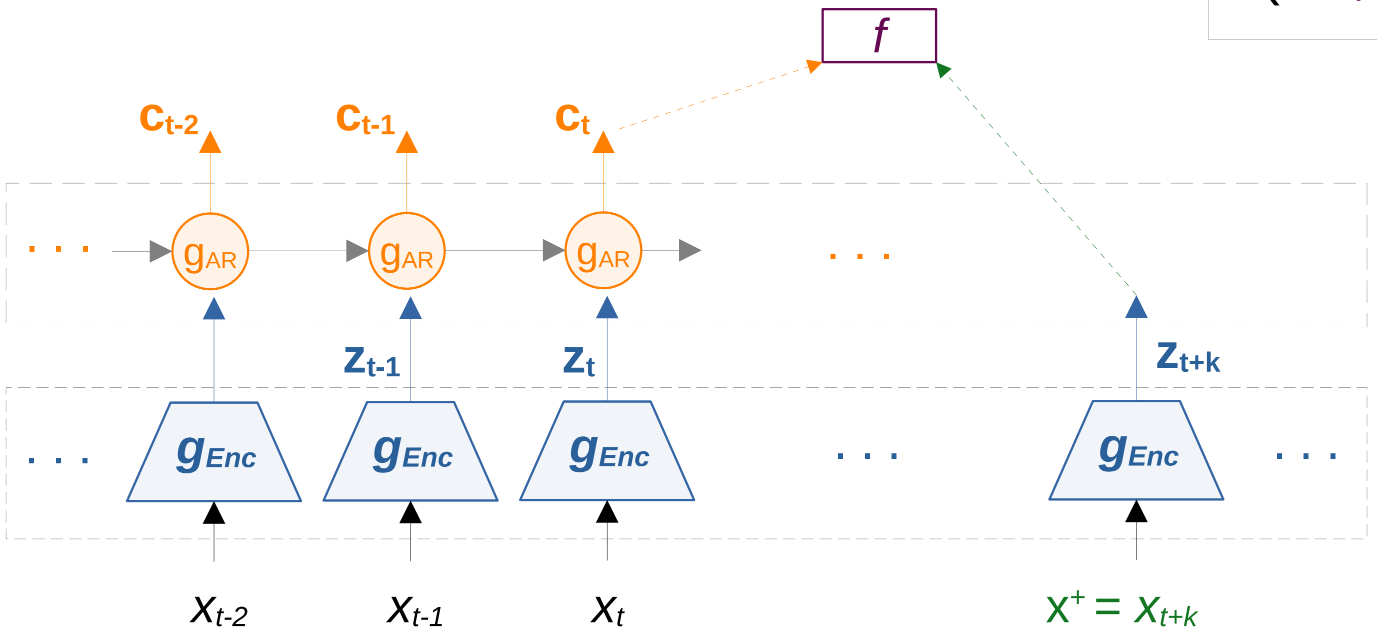
- Coding: Representation Learning
- Predictive:
  - Models correlation between "*history+current*" & "*future*"
  - Requires learning global/local structure & shared info between parts … beyond local smoothness
- Contrastive: Learning paradigm ... Loss

CPC Components

- Architecture: Encoder + AutoRegressive
- Model: Log-Bilinear (similarity measure)

$$z_t = g_{Enc}(x_t) \qquad c_t = g_{AR}(z_t)$$

$$f_k(x_{t+k}, c_t) = \exp(z_{t+k}^T W_k c_t)$$



g_{AR}: RNN-ish
g_{enc}: CNN-ish

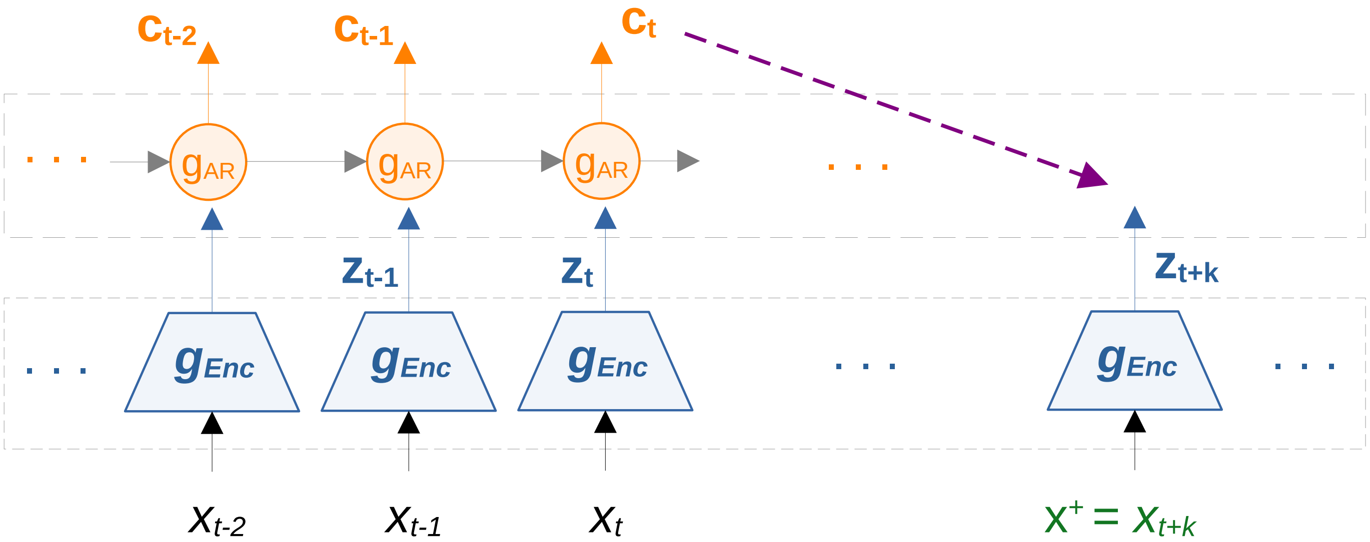
CPC Goal

* Goal: using c_t, *predict* z_{t+k}

* **Question**: Predict "$t+k$" or "$t+1 \leq t \leq t+k$" ???

c_t: "*current* + *history*"

x_{t+k}: "*future*"



g_{AR}: RNN-ish
g_{enc}: CNN-ish

CPC Goal

* Goal: using c_t, *predict* z_{t+k}

* How: Contrastive Loss

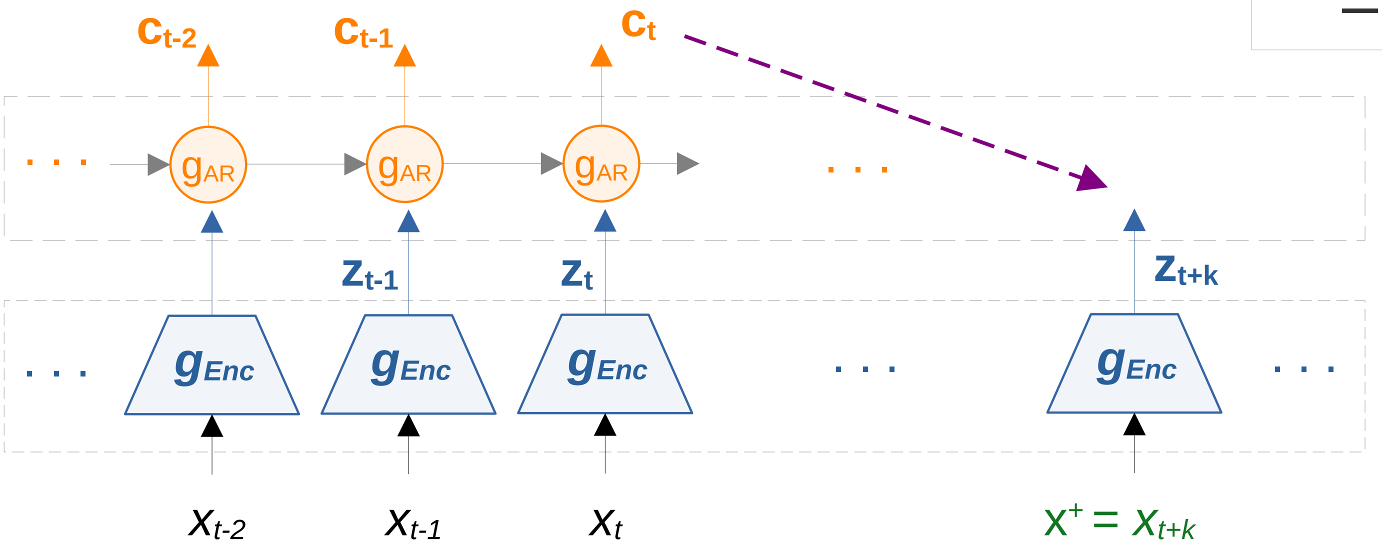
Triplet ...

– Anchor: c_t

– $x^+ \sim P(x|c_t) \leftrightarrow x^+ = x_{t+k}$

– $x^- \sim P(x_{t+k})$

Proposal distribution



g_{AR}: RNN-ish
g_{enc}: CNN-ish

CPC Loss: InfoNCE

- Given X = {x_1, x_2, …, x_N}, $x^+ = x^{t+k}$; $\#x^- = N-1$
- InfoNCE* is different from NCE** loss used in Word2Vec

$$\mathcal{L}_X(\text{anchor}) = -\log \frac{\text{sim}(x^+, \text{anchor})}{\sum_{x_j \in X} \text{sim}(x_j, \text{anchor})}$$

$$= -\log \frac{\text{sim}(x^+, \text{anchor})}{\text{sim}(x^+, \text{anchor}) + \sum_{x_j \in X_{neg}} \text{sim}(x_j, \text{anchor})}$$

General form

$$\mathcal{L}^{\text{InfoNCE}} = \mathbb{E}_X \left[\mathcal{L}_X \right]$$

* NCE: Noise Contrastive Estimation
** Actually, negative sampling (simpler version) is used.

CPC Loss: InfoNCE

- Given $X = \{x_1, x_2, \ldots, x_N\}$, $x^+ = x^{t+k}$; $\#x^- = N\text{-}1$
- InfoNCE* is different from NCE** loss used in Word2Vec

$$\mathcal{L}_X = -\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}$$

$$= -\log \frac{f_k(x_{t+k}, c_t)}{f_k(x_{t+k}, c_t) + \sum_{x_j \in X_{neg}} f_k(x_j, c_t)}$$

In CPC context

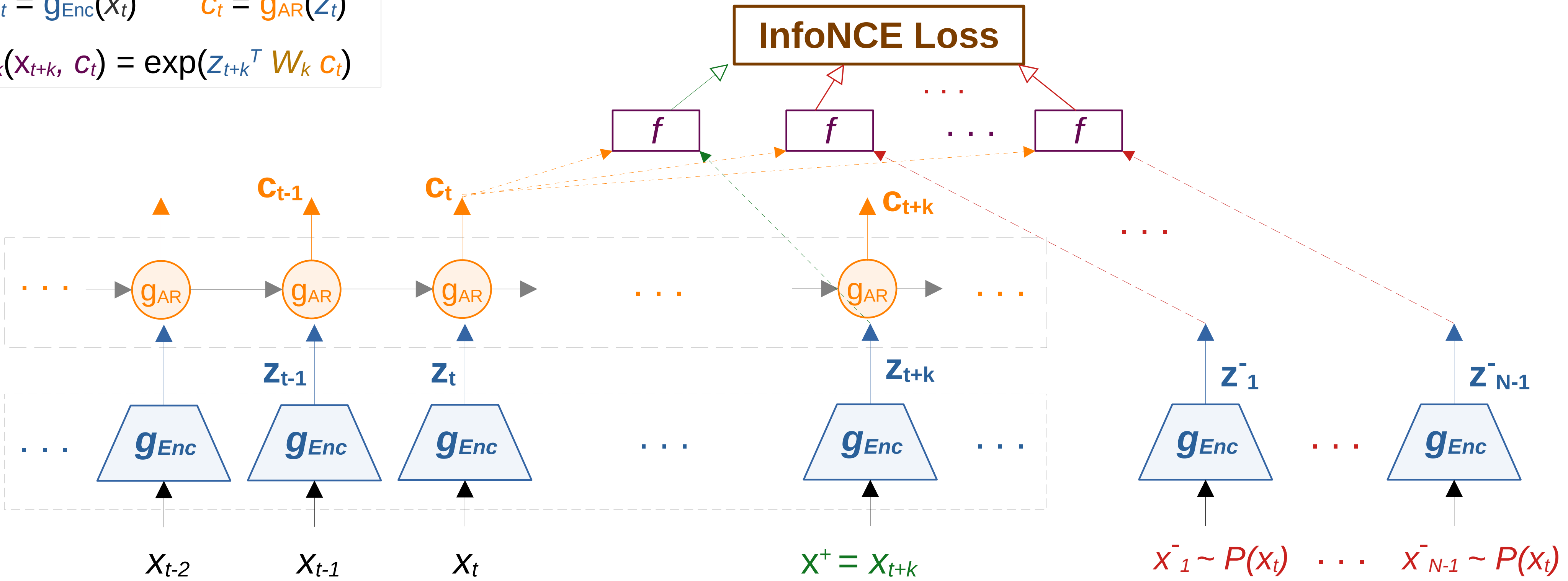
$$\mathcal{L}^{\text{InfoNCE}} = \mathbb{E}_X \left[\mathcal{L}_X \right]$$

* NCE: Noise Contrastive Estimation
** Actually, negative sampling (simpler version) is used.

CPC Framework

$z_t = g_{Enc}(x_t)$ $c_t = g_{AR}(z_t)$

$f_k(x_{t+k}, c_t) = \exp(z_{t+k}^T W_k c_t)$



g_{AR}: RNN-ish
g_{enc}: CNN-ish

InfoNCE Interpretation (1)

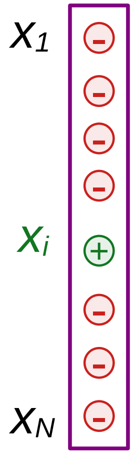
- Given $\boldsymbol{X} = \{x_1, x_2, \ldots, x_N\}$, $x^+ = x^{t+k}$; $\#x^- = N-1$
- InfoNCE Loss ≡ Categorical CE Loss

$$\mathcal{L}_X = -\log \frac{f_k(x_{t+k}, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)}$$

$P(x_i = x^+ \mid X, c_t)$

$$= -\log \frac{f_k(x_{t+k}, c_t)}{f_k(x_{t+k}, c_t) + \sum_{x_j \in X_{neg}} f_k(x_j, c_t)}$$

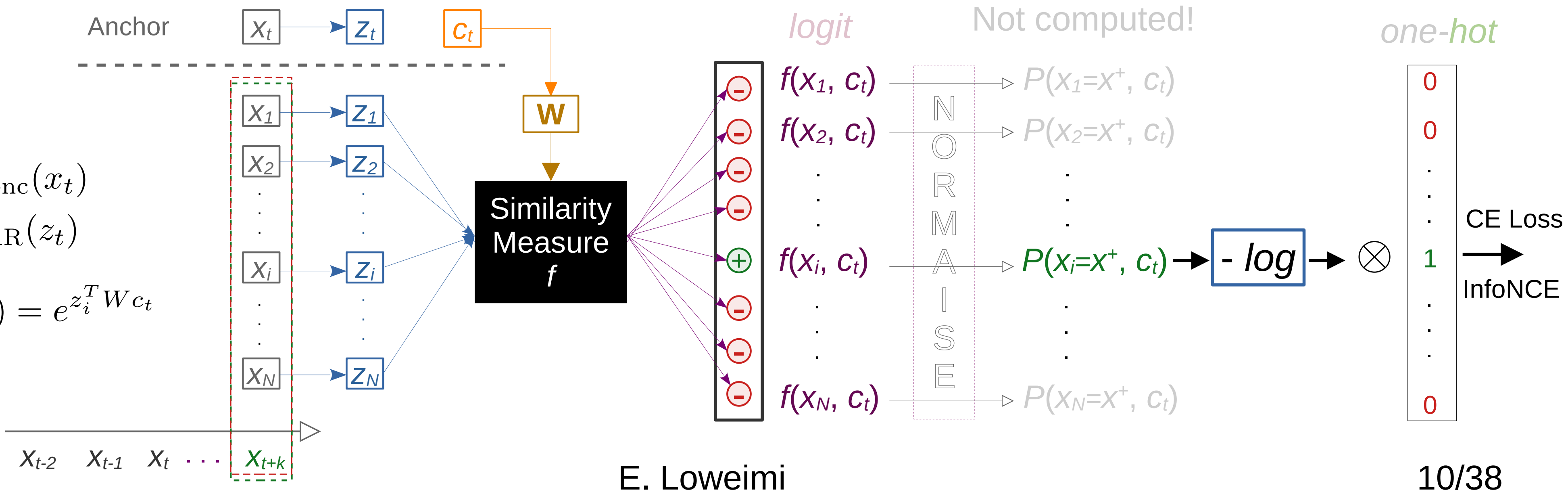
$$\mathcal{L}^{\text{InfoNCE}} = \mathbb{E}_X \left[\mathcal{L}_X \right]$$



InfoNCE Interpretation (1)

- Given $X = \{x_1, x_2, \ldots, x_N\}$, $x^+ = x^{t+k}$; $\#x^- = N\text{-}1$
- InfoNCE ≡ Categorical CE Loss ≡ Models $P(x_i = x^+ \mid X, c_t)$

$$z_t = g_{\text{Enc}}(x_t)$$
$$c_t = g_{\text{AR}}(z_t)$$
$$f(x_i, c_t) = e^{z_i^T W c_t}$$



InfoNCE Interpretation (1)

- Given X = {x_1, x_2, …, x_N}, $x^+ = x^{t+k}$; #x^- = N-1
- Minimise InfoNCE Loss ≡ Maximise $P(x_i = x^+ \mid X, c_t)$

$$P(x_i = x^+ \mid X, c_t) = \frac{f_k(x_i = x^+, c_t)}{\sum_{x_j \in X} f_k(x_j, c_t)} = \frac{f_k(x_i = x^+, c_t)}{f_k(x = x^+, c_t) + \sum_{x_j \in X_{neg}} f_k(x_j, c_t)}$$

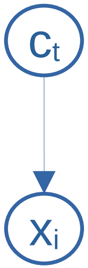
$$\mathcal{L}_{\text{InfoNCE}} = -\mathbb{E}_X \left[\log P(x_i = x^+ \mid X, c_t) \right]$$

InfoNCE Loss Interpretation (2)

- Maximising $P(x_i = x^+ \mid X, c_t)$ ≡ Minimising $L_{InfoNCE}$
- Maximising $P(x_i = x^+ \mid X, c_t)$ *RELATED* to maximising $I(x;c)$

Proof in Appendix A

$$P(x_i = x^+ \mid X, c_t) = \cdots = \frac{\frac{P(x_i|c_t)}{P(x_i)}}{\sum_j \frac{P(x_j|c_t)}{P(x_j)}} \propto \frac{P(x_i \mid c_t)}{P(x_i)}$$



$$\mathbb{I}(x;c) = \sum_{(x,c)} P(x,c) \log \frac{P(x,c)}{P(x)P(c)} = \sum_{(x,c)} P(x,c) \log \frac{P(x|c)}{P(x)} \propto \frac{P(x|c)}{P(x)}$$

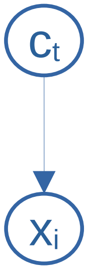
Mutual Information (MI)

InfoNCE Loss Interpretation (2)

- Minimising $L_{InfoNCE}$ ≡ Maximising $P(x_i = x^+ \mid X, c_t)$ …
- … is *RELATED, but NOT EQUIVALENT*, to maximising $I(x;c)$ …

$$P(x_i = x^+ \mid X, c_t) = \cdots = \frac{\frac{P(x_i|c_t)}{P(x_i)}}{\sum_j \frac{P(x_j|c_t)}{P(x_j)}} \propto \frac{P(x_i \mid c_t)}{P(x_i)}$$

$$\mathbb{I}(x;c) = \sum_{(x,c)} P(x,c) \log \frac{P(x,c)}{P(x)P(c)} = \sum_{(x,c)} P(x,c) \log \frac{P(x|c)}{P(x)} \propto \frac{P(x|c)}{P(x)}$$



$$\boxed{\mathbb{I}(x;c) \geq \log N \; - \; \mathcal{L}_{\text{InfoNCE}}}$$

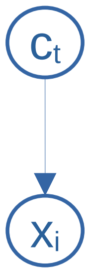
InfoNCE Loss Interpretation (3)

- Minimising $L_{InfoNCE}$ ≡ maximising the lower bound of $I(x;c)$
- Effect of Larger N
  - ✔ Tighter lower bound
  - ✔ Implicit hard mining

$$\mathbb{I}(x;c) = \sum_{x,c} P(x,c) \log \frac{P(x,c)}{P(x)} = \cdots = \mathbb{E}_X \left[\log \frac{P(x_{t+k}, c_t)}{P(x_{t+k})} \right]$$

$$\mathbb{I}(x;c) \geq \log N \; - \; \mathcal{L}_{\text{InfoNCE}}$$

X = {positive, negatives}



Quiz Time

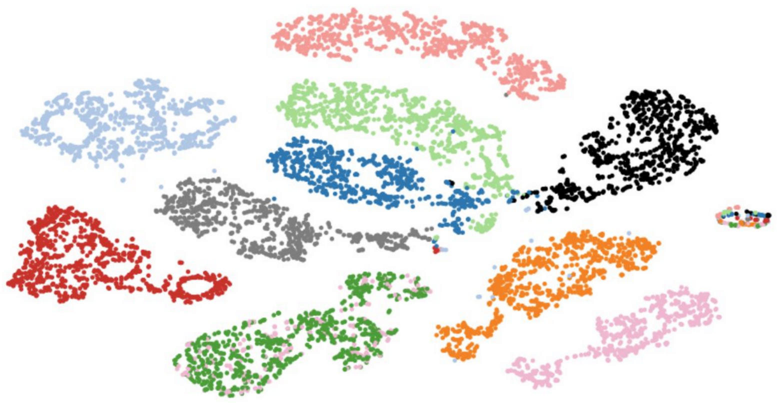
$$z_t = g_{\text{Enc}}(x_t)$$
$$c_t = g_{\text{AR}}(z_t)$$

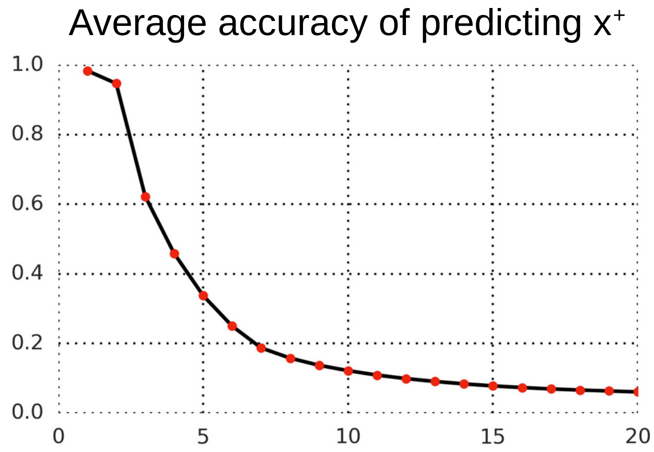
- Recall $f(x_{t+k}, c_t) = \exp(z_{t+k}^T (W c_t))$; Compare following h_is with f ...
  - Q1: $h_1(x_{t+k}, c_t) = \exp(z_{t+k}^T (W z_t))$
  - Q2: $h_2(x_{t+k}, c_t) = \exp(c_{t+k}^T (W c_t))$
  - Q3: $h_3(x_{t+k}, c_t) = \exp((z_{t+k}^T W) c_t)$
  - **Q4**: $h_4(x_{t+k}, c_t) = \exp(x_{t+k}^T (W c_t))$
- Q5: Larger N is better, here N=8; What does upperbound N?
- Q6: Compare CPC with LPC
- Q7: Compare CPC with AutoEncoder (Prediction vs Reconstruction)

Experimental Setting

- Data: 100h LibriSpeech, 16 kHz
- Task: phone (41 classes) & speaker (251 classes) classification
- g_{Enc}: ResNet; 5-layer CNN + FC (512 nodes) $\rightarrow z_t$
  - Input: raw wave; Strides: [5,4,2,2,2] → frame shift: 160 samples = 10 ms
- g_{AR}: GRU with 256 nodes $\rightarrow c_t$
- Optimiser: Adam
- Size of mini-batch: N=8; Prediction target: k=12
- Classifier: Multi-class linear logistic regression

Experimental Results (1)





* t-SNE visualisation
  – k=12, N=8
  – Each colour ↔ a speaker

* Larger k → lower accuracy in predicting x^+ [N=8]
  – Recall InfoNCE ≡ Categorical CE

Experimental Results (2)

Train a linear classier on top of ...

Method	ACC	
Phone classification		
Random initialization	27.6	
MFCC features	39.7	
CPC	64.6	**72.5**
Supervised	74.6	
Speaker classification		
Random initialization	1.87	
MFCC features	17.6	
CPC	97.4	
Supervised	98.5	

* Random Init.: g_{Enc} and g_{AR} untrained

* Supervised: train E2E with the same architecture

* Using non-linear classifier (single hidden layer): 64.6 → **72.5**

==>> CPC representation is NOT perfectly linearly separable

Experimental Results (3)

* Optimal *k* for phone classification is 12.

* Optimal *k* for speaker classification is ???

* Best negative samples ... from Same *spk*, why?
Supplies harder (≡ better) negatives than Mixed spk.

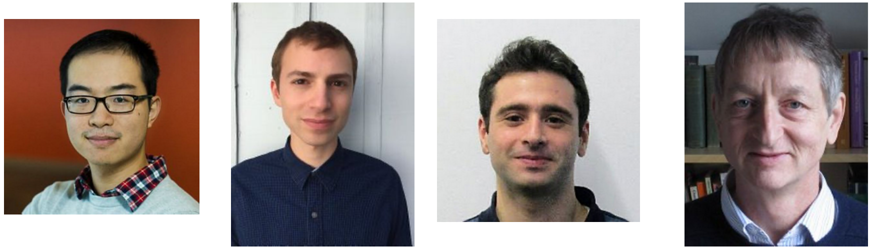
* Hard negative: a negative that is closer or as close as a positive sample to the anchor.

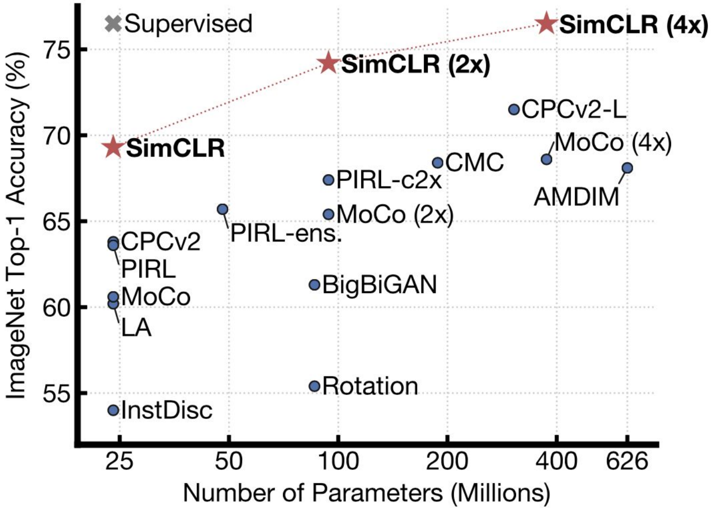
Method	ACC
#steps predicted	
2 steps	28.5
4 steps	57.6
8 steps	63.6
12 steps	64.6
16 steps	63.8
Negative samples from	
Mixed speaker	64.6
Same speaker	65.5
Mixed speaker (excl.)	57.3
Same speaker (excl.)	64.6
Current sequence only	65.2

PMLR 2020

A Simple Framework for Contrastive Learning of Visual Representation

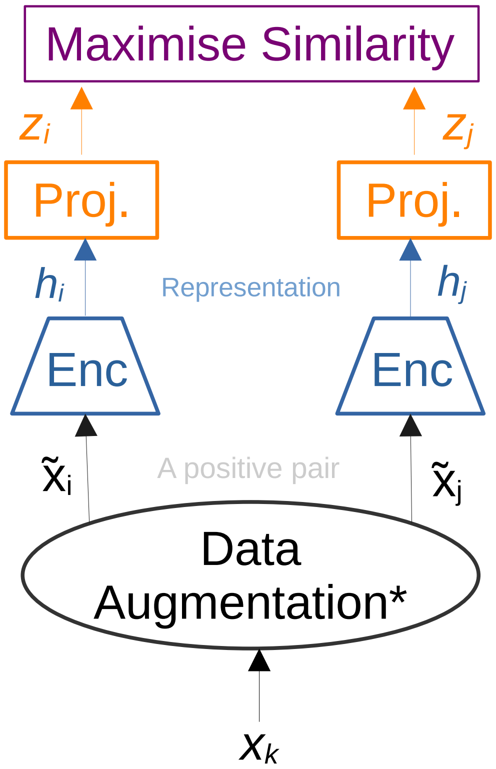
Ting Chen [1] **Simon Kornblith** [1] **Mohammad Norouzi** [1] **Geoffrey Hinton** [1]





Framework's Modules

- Data Augmentation*, $\tilde{x} = \text{Aug}(x)$
  - Two correlated views per anchor
- Encoder: ResNet (no RNN)
  - $h = \text{Enc}(x)$ → downstream tasks
- Projection Head
  - $z = g(h) = W_2\ \text{ReLU}(W_1\ h)$ rather than $W_1\ h$



Contrastive Loss: NT-Xent

- Given *batch*: $\{x_k, y_k\}$, $k=1, \ldots, N$
- Data Aug returns: $\{\tilde{x}_l, \tilde{y}_l\}$, $l = \{1, \ldots, 2N\}$ ← *multi-viewed batch*
  - Consists of N positive pairs: $(\tilde{x}_{2k-1}, \tilde{x}_{2k})$; $\tilde{x}_{2k-1} \sim \boldsymbol{T}$ & $\tilde{x}_{2k} \sim \boldsymbol{T}$
- *Sim* is cosine similarity (L_2-**N**ormed)

Normalised
Temperature-scale
Cross **Ent**ropy

$$\mathcal{L} = \sum_{i \in I} \mathcal{L}_i = -\sum_{i=1}^{2N} \log \frac{\exp(\mathrm{sim}(z_i, z_{j(i)})/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\mathrm{sim}(z_i, z_k)/\tau)}$$

NT-Xent vs Multi-class N-pair Loss

- Multi-class N-pair loss (N-pair-mc) → Extension of Triplet
  - instead of one negative sample, involves *N-1* negative pairs
- NT-Xent & N-pair-mc r identical equation-wise, except temperature scaling
- Both include *N* pairs BUT generated differently …
  - NT-Xent ↔ Data Aug*; N-pair-mc ↔ class labels

$$\mathcal{L}_{\text{N-pair-mc}} = \log\left(1 + \sum_{k=1}^{2N} 1_{[k \neq i,j]} \exp(\mathbf{z}_i^T \mathbf{z}_k - \mathbf{z}_i^T \mathbf{z}_j)\right)$$

Positive pair: (x_i, x_j)

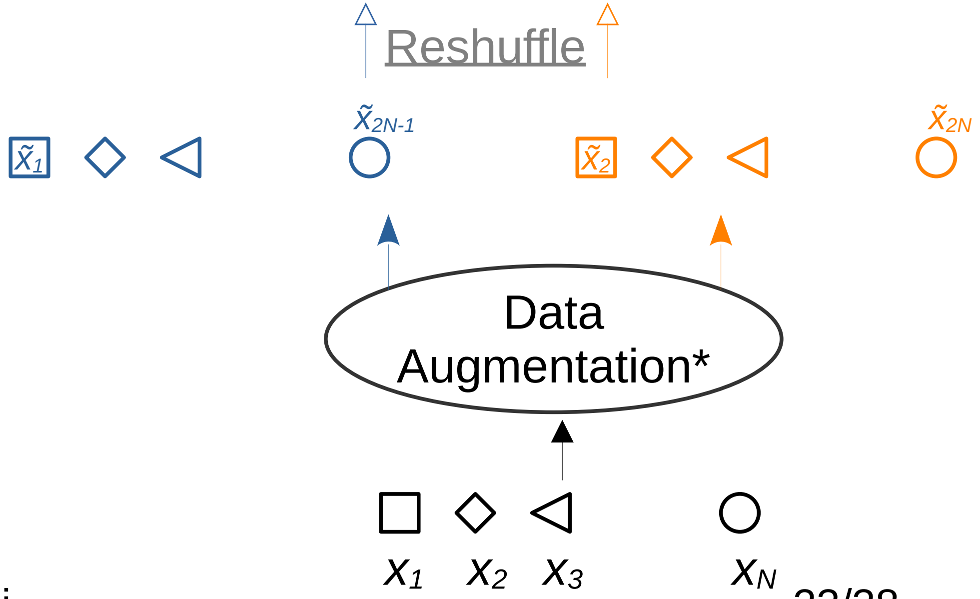
$$= -\log \frac{\exp(\mathbf{z}_i^T \mathbf{z}_j)}{\exp(\mathbf{z}_i^T \mathbf{z}_j) + \sum_{k=1}^{2N} 1_{[k \neq i,j]} \exp(\mathbf{z}_i^T \mathbf{z}_k)}$$

$\log(1 + \exp(x)) \approx \max(0, x)$

Understanding Data Augmentation*

$$\mathcal{L} = \sum_{i \in I} \mathcal{L}_i = -\sum_{i=1}^{2N} \log \frac{\exp(\text{sim}(z_i, z_{j(i)})/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

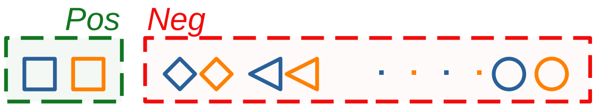
Note: j(i=1) = 2, j(i=2) = 1, …

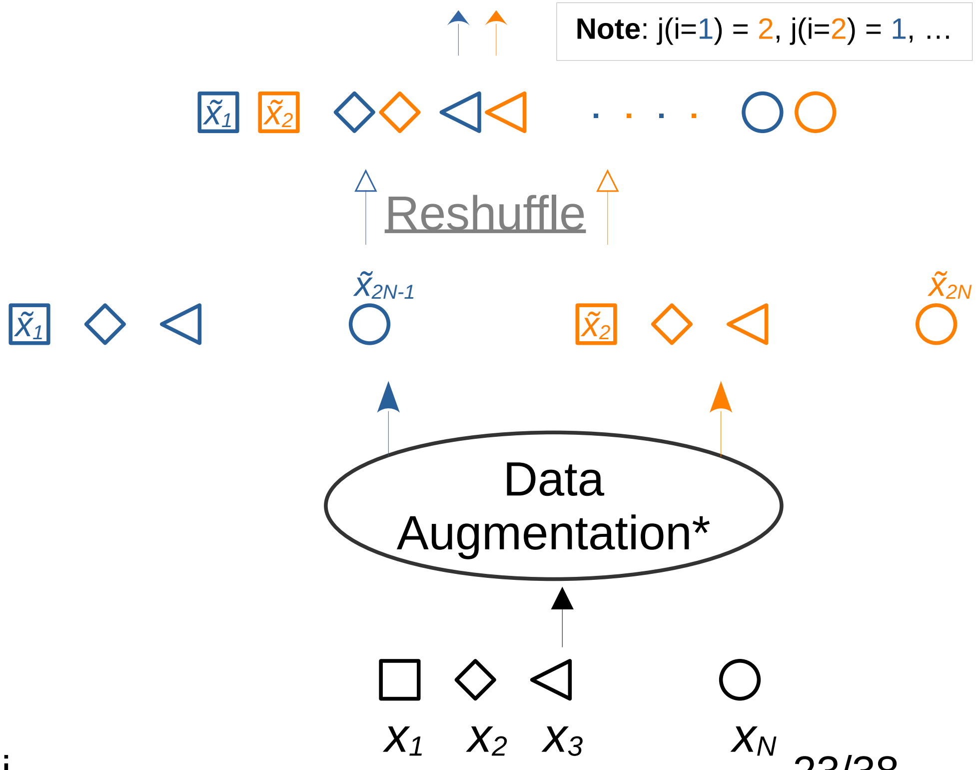


Understanding Data Augmentation*

$$\mathcal{L} = \sum_{i \in I} \mathcal{L}_i = -\sum_{i=1}^{2N} \log \frac{\exp(\text{sim}(z_i, z_{j(i)})/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

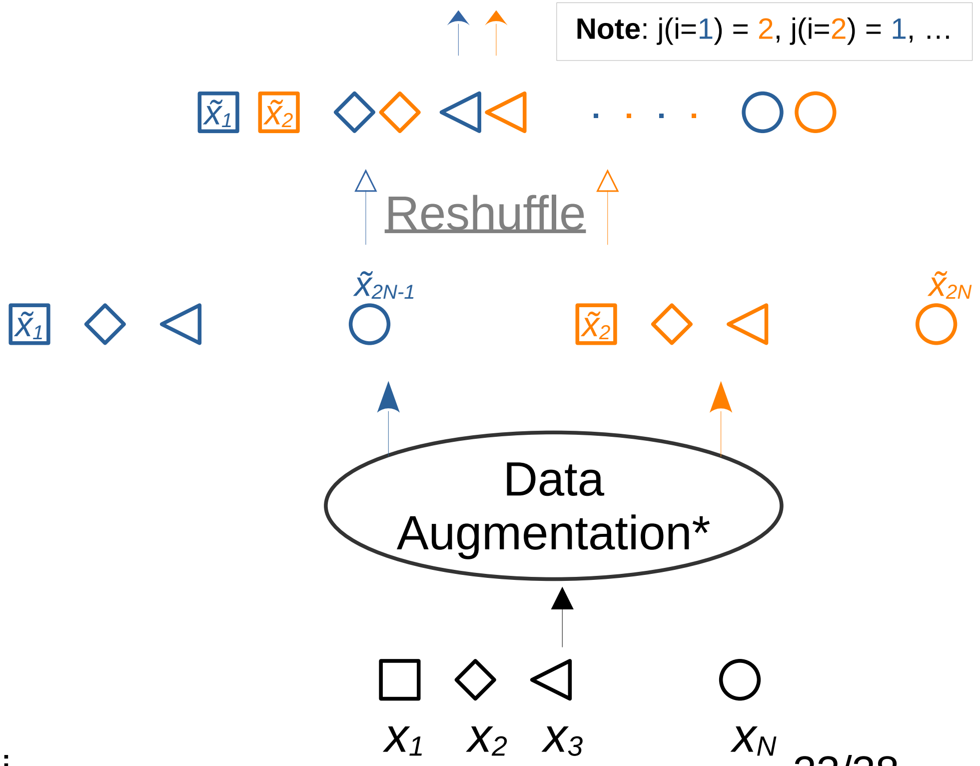
<u>*i* = 1,</u> anchor: $\tilde{x}_1$, x^+: $\tilde{x}_2$



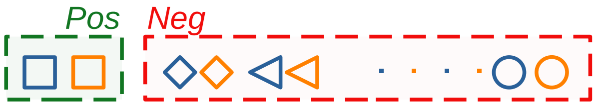


Understanding Data Augmentation*

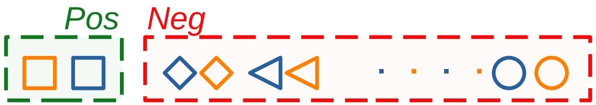
$$\mathcal{L} = \sum_{i \in I} \mathcal{L}_i = -\sum_{i=1}^{2N} \log \frac{\exp(\text{sim}(z_i, z_{j(i)})/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$



_i = 1,_ anchor: $\tilde{x}_1$, x^+: $\tilde{x}_2$



_i = 2,_ anchor: $\tilde{x}_2$, x^+: $\tilde{x}_1$

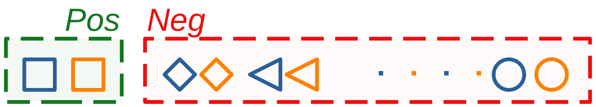


Obviously $\mathcal{L}_1 \neq \mathcal{L}_2$

Understanding Data Augmentation*

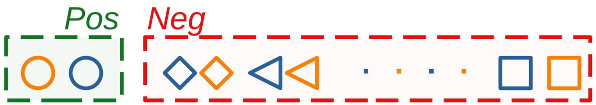
$$\mathcal{L} = \sum_{i \in I} \mathcal{L}_i = -\sum_{i=1}^{2N} \log \frac{\exp(\mathrm{sim}(z_i, z_{j(i)})/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\mathrm{sim}(z_i, z_k)/\tau)}$$

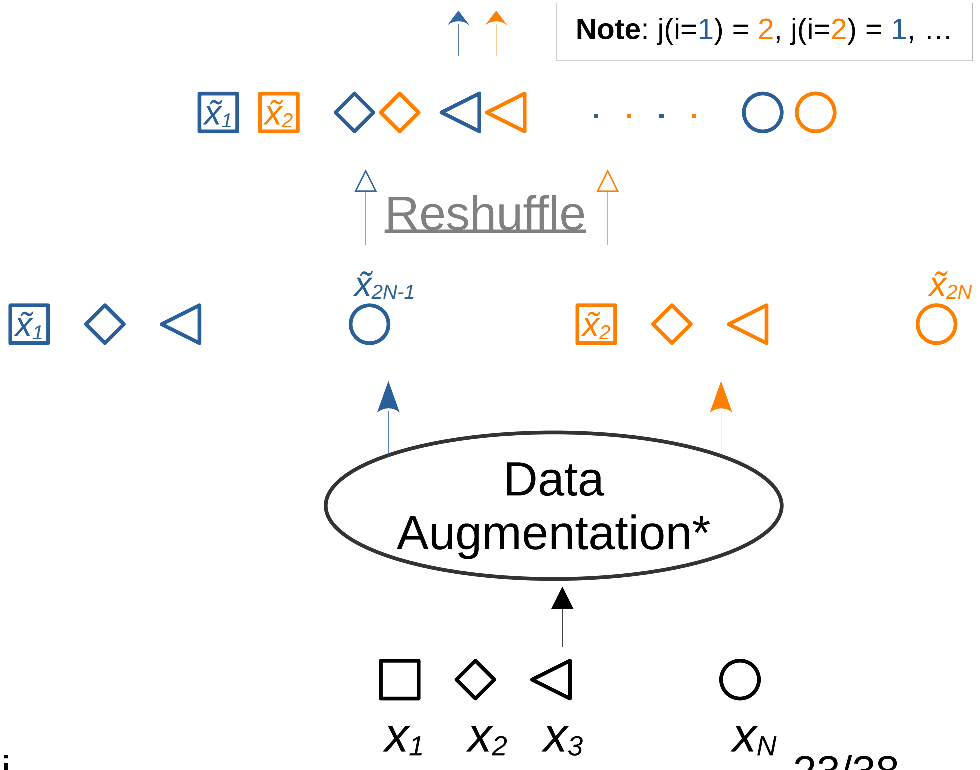
<u>*i* = 1,</u> anchor: $\tilde{x}_1$, x^+: $\tilde{x}_2$



.
.
.

<u>*i* = 2*N*</u>, anchor: $\tilde{x}_{2N}$, x^+: $\tilde{x}_{2N}$

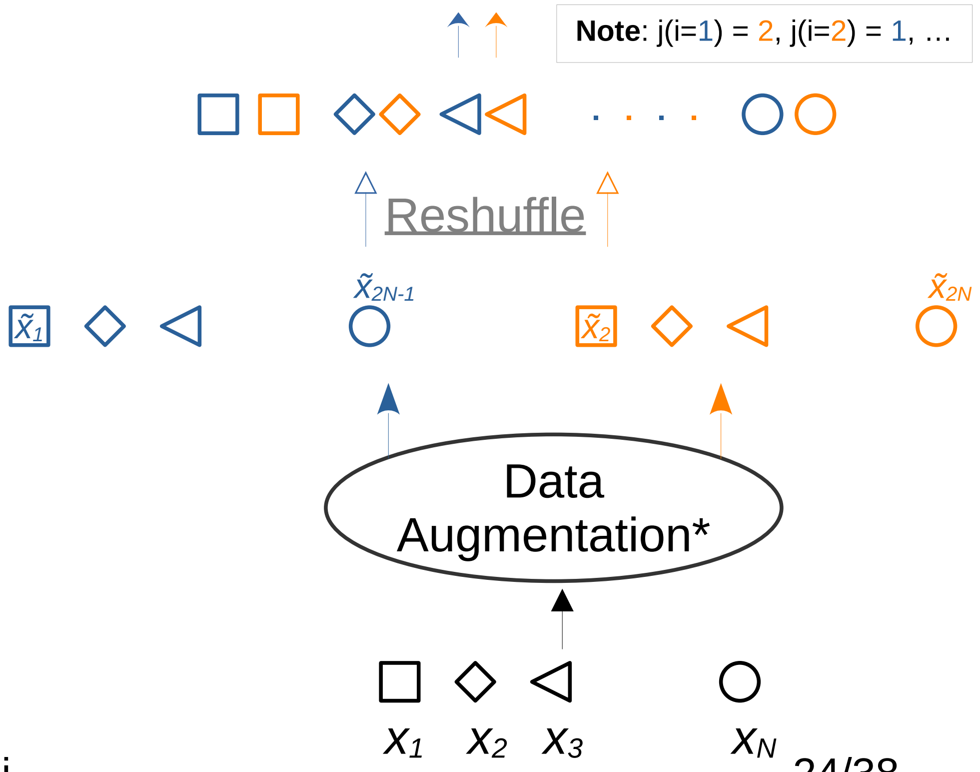




Data Augmentation* … Advantage

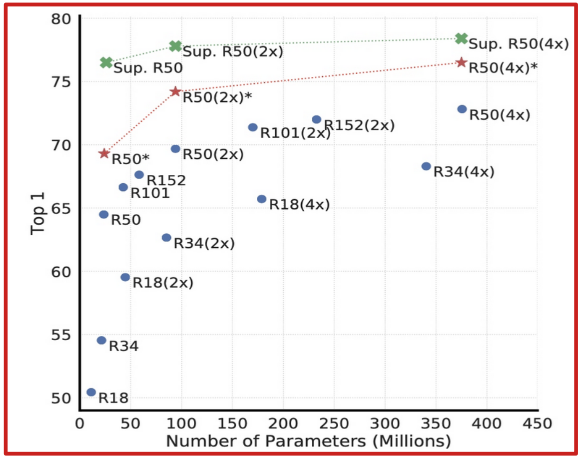
$$\mathcal{L} = \sum_{i \in I} \mathcal{L}_i = -\sum_{i=1}^{2N} \log \frac{\exp(\text{sim}(z_i, z_{j(i)})/\tau)}{\sum_{k=1}^{2N} 1_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}$$

* Anchor is not directly used in loss
* More general framework than CPC
* CPC was applicable to sequential data



Experimental Results (1)

- Train a Linear classifier on top of learned representation →



- Semi-supervised
  - Sample n% (class balanced)
  - fine-tune

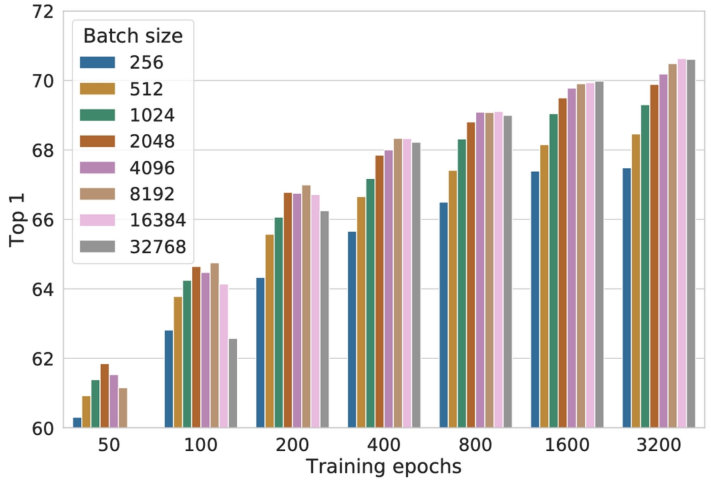
Architecture	Label fraction 1%		10%		100%	
	Top 1	Top 5	Top 1	Top 5	Top 1	Top 5
ResNet-50	49.4	76.6	66.1	88.1	76.0	93.1
ResNet-50 (2×)	59.4	83.7	71.8	91.2	79.1	94.8
ResNet-50 (4×)	64.1	86.6	74.8	92.8	80.4	95.4

2x: 2 times wider ResNet-50

Experimental Results – #Epochs & Batch size

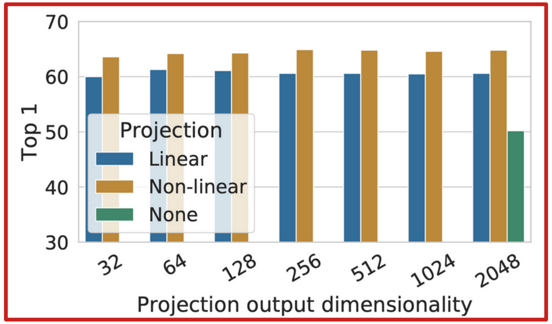
- Compared with supervised paradigm we require …
  - More epochs (> x100)
  - Larger batch (> x1000)

* ALMOST one order of magnitude larger

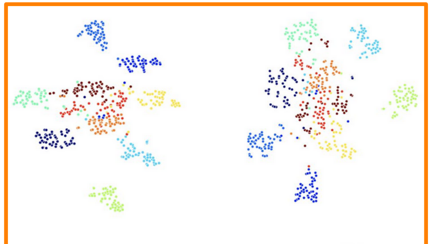


Experimental Results – Projection Head

- Role: map to a loss space
- Effect on overall accuracy:
  - Non-linear > Linear > None
- Its dimension is not critical!
- Useful for computing loss, NOT as a representation!
  - Accuracy & cluster separation



What to predict?	Random guess	Representation h	$g(h)$
Color vs grayscale	80	99.3	97.4
Rotation	25	67.6	25.6
Orig. vs corrupted	50	99.5	59.6
Orig. vs Sobel filtered	50	96.6	56.3



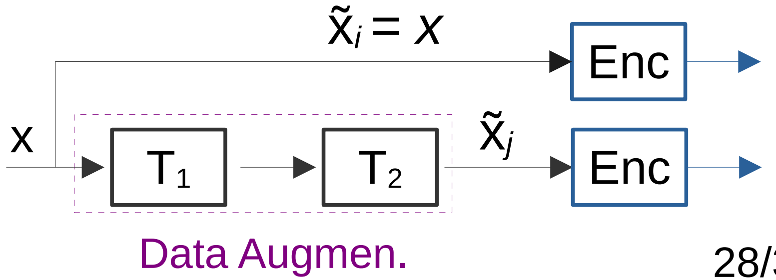
(a) h (b) $z = g(h)$

Experimental Results – Data Augmen.

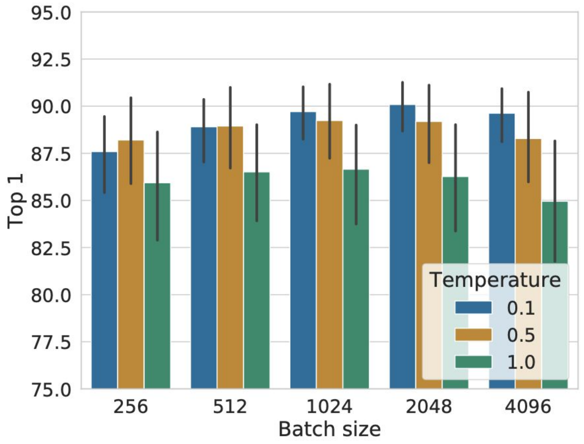
- Composition of Data Augmentation is important!
- Best (here):
  - Random crop + colour distortion
- Note: low performance because of asymmetric augmentation
  - Recall $(\tilde{x}_i, \tilde{x}_j) \sim \boldsymbol{T}$

ImageNet Top-1 Accuracy

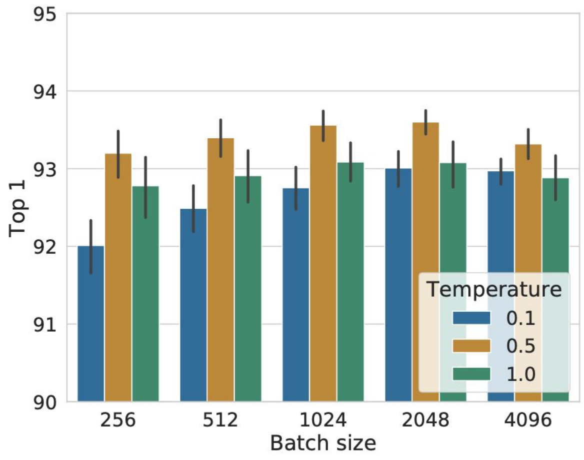
1st transformation \ 2nd transformation	Crop	Cutout	Color	Sobel	Noise	Blur	Rotate	Average
Crop	33.1	33.9	56.3	46.0	39.9	35.0	30.2	39.2
Cutout	32.2	25.6	33.9	40.0	26.5	25.2	22.4	29.4
Color	55.8	35.5	18.8	21.0	11.4	16.5	20.8	25.7
Sobel	46.2	40.6	20.9	4.0	9.3	6.2	4.2	18.8
Noise	38.8	25.8	7.5	7.6	9.8	9.8	9.6	15.5
Blur	35.1	25.2	16.6	5.8	9.7	2.6	6.7	14.5
Rotate	30.0	22.5	20.7	4.3	9.7	6.5	2.6	13.8



Experimental Results – Temperature



(a) Training epochs ≤ 300



(b) Training epochs > 300

* Temperature adjustment is helpful for any batch size or #epochs.
  – [HERE] $\tau_{optimal}$ usually < 1

* More discussion on τ role in the next paper ...

NIPS 2020

Supervised Contrastive Learning

Prannay Khosla *
Google Research

Piotr Teterwak *
Boston University

Chen Wang †
Snap Inc.

Aaron Sarna ‡
Google Research

Yonglong Tian
MIT

Phillip Isola
MIT

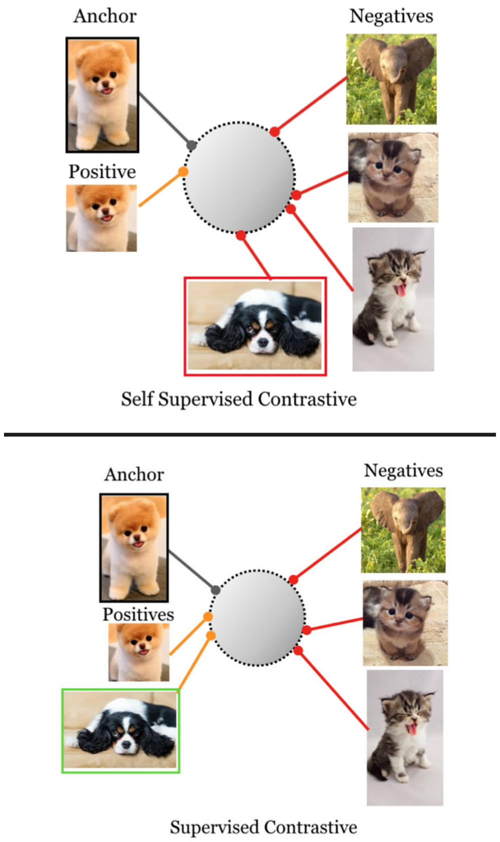
Aaron Maschinot
Google Research

Ce Liu
Google Research

Dilip Krishnan
Google Research

Motivation

- Unsupervised:
  - Triplet: (*anchor*, x^+, x^-)
- **Challenge**:
  - x^- & anchor … same class
  - Multiple x^+s in X
- **Solution**: use class labels
- **How**: *supervised* contrastive



Recall NT-Xent ...

- Given *batch*: $\{x_k, y_k\}$, $k=1, \ldots, N$
  - Data Aug returns: $\{\tilde{x}_l, \tilde{y}_l\}$, $l = \{1, \ldots, 2N\}$ ← *multiviewed batch*
    - $\tilde{x}_{2k-1}$ & $\tilde{x}_{2k}$ ← two views of x_k
  - Triplet → $\tilde{x}_i$: anchor; $\tilde{x}_{j(i)}$: x^+; $\tilde{x}_m$: x^- where $m = l - \{i, j(i)\}$
  - z_i = Proj (Enc($\tilde{x}_i$))

$$\mathcal{L}^{self} = \sum_{i \in I} \mathcal{L}_i^{self} = -\sum_{i \in I} \log \frac{\exp(z_i^T z_{j(i)}/\tau)}{\sum_{a \in A(i)} \exp(z_i^T z_a/\tau)}$$

$A(i) = I - \{i\}$

Supervised Contrastive Loss (SupCon)

$$\mathcal{L}^{self} = \sum_{i \in I} \mathcal{L}_i^{self} = -\sum_{i \in I} \log \frac{\exp(z_i^T z_{j(i)}/\tau)}{\sum_{a \in A(i)} \exp(z_i^T z_a/\tau)}$$

Simply ... Average over batch positives!

Avg **outside** log

$$\mathcal{L}_{out}^{sup} = \sum_{i \in I} \mathcal{L}_{out,i}^{sup} = -\sum_{i \in I} \frac{1}{|P(i)|} \sum_{p \in P(i)} \log \frac{\exp(z_i^T z_p/\tau)}{\sum_{a \in A(i)} \exp(z_i^T z_a/\tau)}$$

$$\mathcal{L}_{in}^{sup} = \sum_{i \in I} \mathcal{L}_{in,i}^{sup} = -\sum_{i \in I} \log \left\{ \frac{1}{|P(i)|} \sum_{p \in P(i)} \frac{\exp(z_i^T z_p/\tau)}{\sum_{a \in A(i)} \exp(z_i^T z_a/\tau)} \right\}$$

Avg **inside** log

* P(i): set of x^+ in A(i) for anchor x_i
* |P(i)|: cardinality

Learning a Nonlinear Embedding by Preserving Class Neighbourhood Structure

Ruslan Salakhutdinov and Geoffrey Hinton
Department of Computer Science
University of Toronto

SupCon vs NCA

The NCA objective (as in [9]) is to maximize the expected number of correctly classified points on the training data:

$$O_{NCA} = \sum_{a=1}^{N} \sum_{b:c^a=c^b} p_{ab} \tag{5}$$

One could alternatively maximize the sum of the log probabilities of correct classification:

$$O_{ML} = \sum_{a=1}^{N} \log \Big(\sum_{b:c^a=c^b} p_{ab} \Big) \tag{6}$$

$$\mathcal{L}_{out}^{sup} = \sum_{i\in I} \mathcal{L}_{out,i}^{sup} = -\sum_{i\in I} \frac{1}{|P(i)|} \sum_{p\in P(i)} \log \frac{\exp(z_i^T z_p/\tau)}{\sum_{a\in A(i)} \exp(z_i^T z_a/\tau)}$$

SupCon includes **log**, **NCA** does not.

$$\mathcal{L}_{in}^{sup} = \sum_{i\in I} \mathcal{L}_{in,i}^{sup} = -\sum_{i\in I} \log \left\{ \frac{1}{|P(i)|} \sum_{p\in P(i)} \frac{\exp(z_i^T z_p/\tau)}{\sum_{a\in A(i)} \exp(z_i^T z_a/\tau)} \right\}$$

SupCon: L_{in} vs L_{out}

- Jensen's Inequality
  - Loss (- *log*) is convex
  - A typo ...

$$\mathcal{L}_{in}^{sup} \leq \mathcal{L}_{out}^{sup}$$

$$\mathcal{L}_{out}^{sup} \leq \mathcal{L}_{in}^{sup}$$

cause log is a concave function, Jensen's Inequality [23] implies that $\mathcal{L}_{out}^{sup} \leq \mathcal{L}_{in}^{sup}$. One might thus be tempted to conclude that $\mathcal{L}_{in}^{sup}$ is the superior supervised loss function (since it bounds $\mathcal{L}_{out}^{sup}$). However, this conclusion is *not* supported

Got it right here

SupCon: L_{in} vs L_{out}

- Although $L_{in} \leq L_{out}$, L_{out} is better; Y?
  - Grad L_{in} does not see 1 / |P(i)|
    - Susceptible to bias in |P(i)|
  - I think … *log Σ* vs. *Σ log*
    - *Σ log* ... closer to *Gaussian*
      - Mean ↔ better representative

Loss	Top-1
$\mathcal{L}^{sup}_{out}$	78.7%
$\mathcal{L}^{sup}_{in}$	67.4%

ImageNet Top-1

Experimental Results (1)

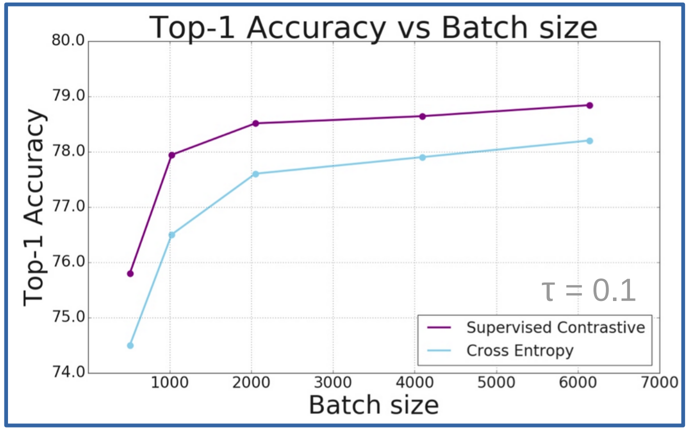
Dataset	SimCLR[3]	Cross-Entropy	Max-Margin [32]	SupCon
CIFAR10	93.6	95.0	92.4	**96.0**
CIFAR100	70.7	75.3	70.5	**76.5**
ImageNet	70.2	78.2	78.0	**78.7**

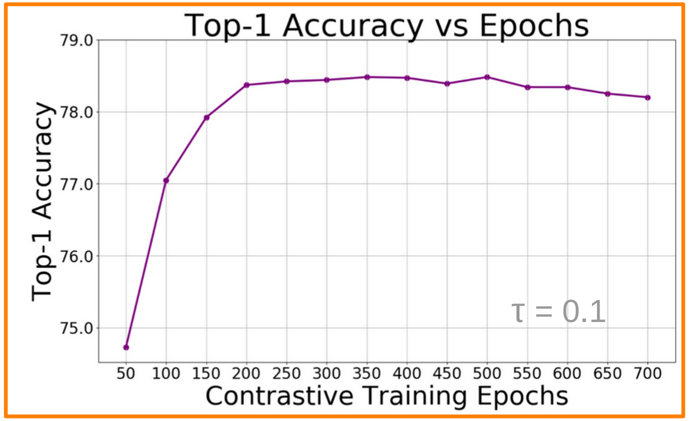
* *SupCon* outperforms *SimCLR;* is it fair?

1 [3]	3	5	7	9	No cap (13)
69.3	76.6	78.0	78.4	78.3	78.5

* Effect of #x^+ in mini-batch (*N*): Diminishing return after 7.

Experimental Results (2)





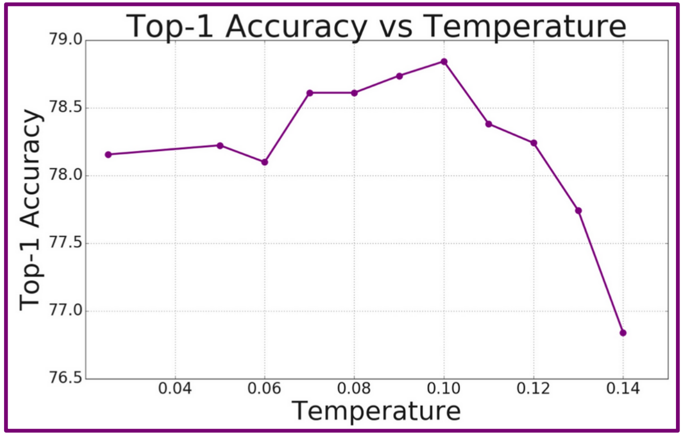
* Batch size is important: The larger the N, the better

* Contrastive learning requires more epochs than supervised

Experimental Results (3)

* **Temperature** (τ) adjustment is helpful ...

* OPTIMAL [here]: 0.1



* Temperature Effects ...

1) Higher τ → Smaller Grad-Loss → $\| \nabla \mathcal{L} \| \propto \frac{1}{\tau}$
2) Higher τ → Smoother (softer/flatter) distribution
3) Lower τ → Makes the hard negatives harder (Y?) $e^{\mathrm{sim}(\mathbf{e}_a, \mathbf{e}_n)/\tau} \overset{\tau < 1}{\gg} e^{\mathrm{sim}(\mathbf{e}_a, \mathbf{e}_n)}$
4) Very Low τ → Numerical instability

Conclusion – Contrastive Learning

- Goal: universal transferable representation learning
- Paradigms: unsupervised (CPC, SimCLR) & supervised
- Loss function: InfoNCE (CPC), NT-Xent (SimCLR), SupCon
- Modules: Data Aug, Encoder+RNN, Projection
- Influential factors:
  - (Composite) Data Aug, Encoder, non-linear projection, #epochs, batch size, temperature, etc.

That’s It!

- Thanks for Your Attention!
- Q/A

- Appendices:
  - A: Density Ratio Proof

Appendix A: Density Ratio Proof

$$P(x_i = x^+ \mid X, c_t) = \frac{P(x_i = x^+, X \mid c_t)}{\sum_{j=1}^{N} P(x_j = x^+, X \mid c_t)} = \cdots = \frac{\frac{P(x_i|c_t)}{P(x_i)}}{\sum_j \frac{P(x_j|c_t)}{P(x_j)}}$$

$$= \frac{1}{Z(c_t)} \frac{P(x_i \mid c_t)}{P(x_i)} \propto \frac{P(x_i \mid c_t)}{P(x_i)}$$

$$P(x_i = x^+, X \mid c_t) = \Pi_{k=1}^{N} P(x_k, x_i = x^+ \mid c_t)$$

$$= \cdots = P(x_i \mid c_t)\ \Pi_{k \neq i} P(x_k) = P(x_i \mid c_t)\ \frac{\Pi_{k=1}^{N} P(x_k)}{P(x_i)}$$

$$P(x_j|c_t) = \begin{cases} P(x_j|c_t), & x_j = x^+ \\ P(x_j), & x_j = x^- \end{cases}$$