Do Lessons from Metric Learning Generalize to Image-Caption Retrieval?

Maurits Bleeker & Maarten de Rijke
{m.j.r.bleeker, m.derijke}@uva.nl

April 13, 2022

IRLab
University of Amsterdam

Task: Image-Caption Retrieval

Image-Caption Retrieval i

Task: Image-Caption Retrieval (ICR) is the task of retrieving images or captions based on a **query** in the different *modality*.

Data: A set of N image-caption tuples/pairs, for each image $x^i_{\mathcal{I}}$, we have k captions $x^i_{\mathcal{C}j}$, $1 \leq j \leq k$.

$\mathcal{D} = \{(x^i_{\mathcal{I}}, x^i_{\mathcal{C}1}, \ldots, x^i_{\mathcal{C}k}), \ldots\}_{i=1}^{N}$

- Flickr30k and MS-COCO are two common train and evaluation benchmarks.

Evaluation: Given a query image or caption, find the corresponding image or caption in a set of 5000/1000 captions or images.

Evaluation metric: *Recall*@$\{1, 5, 10\}$ and mAP.

Image-Caption Retrieval ii

To narrow down the scope of this project:

- We use simple ICR methods that:
  - do not require a big compute infrastructure,
  - or are optimized with a vast amount of training data.
- We use global matching methods:
  - I.e. one global representation for the image and the caption.
- We use relatively small datasets [6, 9], compared to SOTA (Jia et al. [5], Yuan et al. [10]) (pre-trained) image-caption retrieval.

Image-Caption Retrieval iii

Figure 1: Contrastive Image-Caption Retrieval framework. Source: `https://openai.com/blog/clip/`.

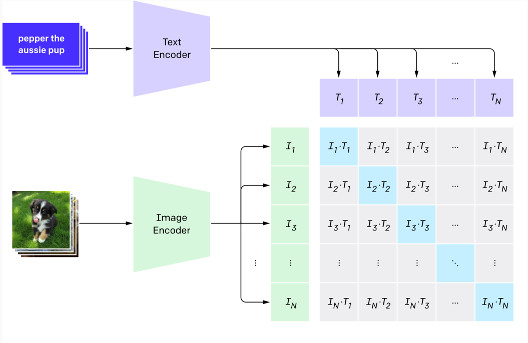

Image-Caption Retrieval - Notation i

1. Latent image representation I_i, computed by the Image Encoder.
2. Latent caption representation T_i, computed by the Caption Encoder.
3. $s = sim(I_i, T_i)$, similarity score metric:
    - $sim = \frac{I_i, T_i}{||I_i||||C_i||}$

Image-Caption Retrieval - Notation ii

Two main developments are accelerating progress in the ICR field:

- New methods for the encoder models.
- Transformer-based methods with more data (pre-training).

The standard training loss for ICR models, that are trained from scratch, is the Triplet loss with semi hard-negatives (in batch negatives).

Image-Caption Retrieval - Notation iii

- Given query $\mathbf{q}$, the task is to rank all candidates in a candidate set $\Omega = \{\mathbf{v}_i \mid i = 0, \ldots, n\}$.
- A matching candidate is denote as $\mathbf{v}^+$.
- Negative candidate(s) as $\mathbf{v}^-$.
- $\mathbf{v}^+ \in \mathcal{P}_\mathbf{q}$ (*positive* candidate set)
- $\mathbf{v}^- \in \mathcal{N}_\mathbf{q}$ (*negative* candidate set)
- $\mathcal{S}_\Omega^\mathbf{q} = \{s_i = \langle \frac{\mathbf{q}}{\|\mathbf{q}\|} \frac{\mathbf{v}_i}{\|\mathbf{v}_i\|} \rangle, i = 0, \ldots, n\}$.

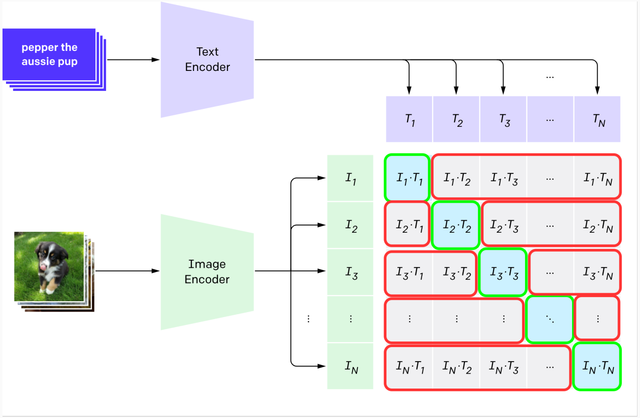

Figure 2: The query $\mathbf{q}$ is an image I_*, $\mathbf{v}^+ \in \mathcal{P}_\mathbf{q}$ in given in green, $\mathbf{v}^- \in \mathcal{N}_\mathbf{q}$ in red

Losses: Metric Learning Functions

Metric Learning functions i

- *Metric learning* focuses on loss functions that result in more accurate item representations (in terms of a given evaluation metric).
    - That can distinguish between similar and dissimilar items in a low-dimensional latent space (Musgrave et al. [7]).
- There has been important progress in metric learning, that result in better evaluation scores on a specific (evaluation) task.
- There has been barely any work that either tries different loss functions or designs new loss functions for the ICR task.
- New loss functions might result in higher evaluation performances, without requiring more data or larger models.

Research Question: *Can newly introduced metric learning approaches, that is, alternative loss functions, be used to increase the performance of ICR methods*?

Why? More data, or more complex network architectures, should not be the only remedy to improve the evaluation scores.

Metric Learning functions iii

We compare three loss functions for the ICR task:

1. The Triplet loss (hinge loss), including semi-hard negative mining,
2. NT-Xent loss and
3. SmoothAP.

The goal is to test a small, but diverse set of loss functions.

Triplet loss SH i

Loss: The Triplet loss with semi-hard negatives (in batch negatives) (Faghri et al. [4]).

$$\mathcal{L}^{\mathbf{q}}_{TripletSH} = \max(\alpha - s^{+} + s^{-}, 0),$$
$$\mathcal{L}_{TripletSH} = \sum_{\mathbf{q} \in \mathcal{B}} \mathcal{L}^{\mathbf{q}}_{Triplet}.$$

- **Intuition:** Make the distance between s^{-} and s^{+} bigger than α.
- Where α is a margin parameter,
- $s^{-} = \max(S^{\mathbf{q}}_{\mathcal{N}})$,
- $s^{+} = s_0 \in S^{\mathbf{q}}_{\mathcal{P}}$.
- Standard choice of optimization function for many ICR methods.

Loss: The Triplet loss (N-Triplets).

$$
\begin{aligned}
\mathcal{L}^{\mathbf{q}}_{Triplet} &= \sum_{s^- \in S^q_{\mathcal{N}}} \max(\alpha - s^+ + s^-, 0), \\
\mathcal{L}_{Triplet} &= \sum_{\mathbf{q} \in \mathcal{B}} \mathcal{L}^{\mathbf{q}}_{Triplet}.
\end{aligned}
$$

Loss: NT-Xent/InfoNCE (Chen et al. [2], Oord et al. [8]).

Intuition: The final loss/optimization is computed across **all** pairs in the batch, using softmax normalization.

$$\mathcal{L}_{NT\text{-}Xent} = -\frac{1}{|\mathcal{B}|} \sum_{\mathbf{q} \in \mathcal{B}} \log \frac{\exp(s^+/\tau)}{\sum_{s_i \in \mathcal{S}_\Omega^{\mathbf{q}}} \exp(s_i/\tau)},$$

SmoothAP i

- SmoothAP (Brown et al. [1]) is a smooth approximation of the Average Precision Metric.

The Average Precision metric is defined as follows:

$$AP_{\mathbf{q}} = \frac{1}{|\mathcal{S}_{\mathcal{P}}^{\mathbf{q}}|} \sum_{i \in \mathcal{S}_{\mathcal{P}}^{\mathbf{q}}} \frac{\mathcal{R}(i, \mathcal{S}_{\mathcal{P}}^{\mathbf{q}})}{\mathcal{R}(i, \mathcal{S}_{\Omega}^{\mathbf{q}})},$$

Where $\mathcal{R}(i, \mathcal{S})$ is a (non-differentiable) function that returns the ranking of candidate $i \in \mathcal{S}$ in the candidate set:

- With some tricks (i.e. using a sigmoid function), $\mathcal{R}(i, \mathcal{S})$ can be reformulated into a differentiable function.
- **Intuition:** Instead of solely optimizing the similarity between the positive and negative candidates, this loss function tries to optimize a ranking directly.

$$AP_{\mathbf{q}} \approx \frac{1}{|\mathcal{S}_{\mathcal{P}}^{q}|} \sum_{i \in \mathcal{S}_{\mathcal{P}}^{\mathbf{q}}} \frac{1 + \sum_{j \in \mathcal{S}_{\mathcal{P}}^{\mathbf{q}}, j \neq i} \mathcal{G}(D_{ij}; \tau)}{1 + \sum_{j \in \mathcal{S}_{\mathcal{P}}^{\mathbf{q}}, j \neq i} \mathcal{G}(D_{ij}; \tau) + \sum_{j \in \mathcal{S}_{\mathcal{N}}^{\mathbf{q}}} \mathcal{G}(D_{ij}; \tau)}.$$

For the ICR task, we evaluate a ranking in the end. Why not optimize with a ranking metric directly?

Do Findings from Metric Learning Extend to ICR?

Experimental setup i

- We take the VSE++ and VSRN as two ICR methods.
  - VSE++: ConvNet (Image Encoder), single layer GRU (Caption Encoder).
  - VSRN: Pre-computed feature map, Graph CNN and a GRU (Image Encoder), single layer GRU (Caption Encoder).
- We only change the loss function, the rest remains the same.
- We evaluate which loss function results in the highest evaluation performance.
  - The goal is to evaluate if promising loss functions from other metric learning tasks improve the ICR evaluation scores
- We evaluate on the MS-COCO and Flickr30k benchmark datasets.

Experimental setup ii

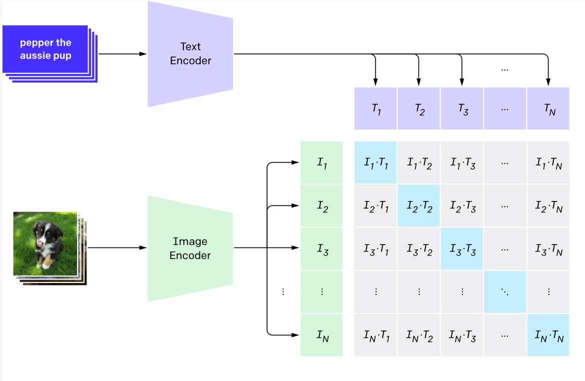

Figure 3: Contrastive Image-Caption Retrieval framework. Source: `https://openai.com/blog/clip/`.

Table 1: Evaluation scores for the Flickr30k, for the VSE++ and VSRN methods.

Loss function	#	hyper param	i2t R@1	i2t R@5	i2t R@10	i2t average recall	i2t mAP@5	t2i R@1	t2i R@5	t2i R@10	t2i average recall	rsum
			Flickr30k									
			VSE++									
Triplet loss	1.1	$\alpha = 0.2$	30.8±.7	62.6±.3	74.1±.8	55.9±.3	0.41±.00	23.4±.3	52.8±.1	65.7±.3	47.3±.1	309.4±0.9
Triplet loss SH	1.2	$\alpha = 0.2$	**42.4**±.5	**71.2**±.7	**80.7**±.7	64.8±.6	0.50±.01	**30.0**±.3	**59.0**±.2	**70.4**±.4	53.1±.2	**353.8**±1.6
NT-Xent	1.3	$\tau = 0.1$	37.5±.6	68.4±.6	77.8±.5	61.2±.3	0.47±.00	27.0±.3	57.3±.3	69.1±.2	51.1±.2	337.1±1.3
SmoothAP	1.4	$\tau = 0.01$	**42.1**±.8	**70.8**±.6	**80.6**±.8	64.5±.4	0.50±.00	29.1±.3	58.1±.1	69.7±.2	52.3±.2	350.4±1.7
			VSRN									
Triplet loss	1.5	$\alpha = 0.2$	56.4±.7	83.6±.6	90.1±.2	76.7±.5	0.63±.01	43.1±.3	74.4±.3	83.1±.4	66.9±.3	430.7±1.8
Triplet loss SH	1.6	$\alpha = 0.2$	**68.3**±1.3	**89.6**±.7	**94.0**±.5	84.0±.5	0.73±.01	**51.2**±.9	**78.0**±.6	**85.6**±.5	71.6±.6	**466.6**±3.3
NT-Xent	1.7	$\tau = 0.1$	50.9 ±.5	78.9±.7	86.6±.4	72.2±.4	0.59±.00	40.6±.6	71.9±.2	81.7±.3	64.7±.2	410.6±1.5
SmoothAP	1.8	$\tau = 0.01$	63.1±1.0	86.6±.8	92.4±.5	80.7±.7	0.69±.00	45.8±.2	73.7±.3	82.3±.2	67.3±.1	444.0±2.1

Table 2: Evaluation scores for the MS-COCO, for the VSE++ and VSRN methods.

Loss function	#	hyper param	i2t R@1	i2t R@5	i2t R@10	i2t average recall	i2t mAP@5	t2i R@1	t2i R@5	t2i R@10	t2i average recall	rsum
			MS-COCO									
			VSE++									
Triplet loss	2.1	$\alpha = 0.2$	22.1±.5	48.2±.3	61.7±.3	44.0±.3	0.30±.00	15.4±.1	39.5±.1	53.2±.1	36.0±.1	240.0±0.9
Triplet loss SH	2.2	$\alpha = 0.2$	**32.5**±.2	**61.6**±.3	**73.8**±.3	56.0±.2	0.41±.00	**21.3**±.1	**48.1**±.1	**61.5**±.0	43.6±.1	**298.8**±0.8
NT-Xent	2.3	$\tau = 0.1$	25.8±.5	53.6±.5	66.1±.2	48.5±.3	0.34±.00	18.0±.1	43.0±.1	56.6±.2	39.2±.1	263.0±0.9
SmoothAP	2.4	$\tau = 0.01$	30.8±.3	60.3±.2	**73.6**±.5	54.9±.3	0.40±.00	20.3±.2	46.5±.2	60.1±.2	42.3±.2	291.5±1.4
			VSRN									
Triplet loss	2.5	$\alpha = 0.2$	42.9±.4	74.3±.3	84.9±.4	67.4±.3	0.52±.00	33.5±.1	65.1±.1	77.1±.2	58.6±.1	377.8±1.2
Triplet loss SH	2.6	$\alpha = 0.2$	**48.9**±.6	**78.1**±.5	**87.4**±.2	71.4±.4	0.57±.01	**37.8**±.5	**68.1**±.5	**78.9**±.3	61.6±.4	**399.0**±2.3
NT-Xent	2.7	$\tau = 0.1$	37.9±.4	69.2±.2	80.7±.3	62.6±.1	0.47±.00	29.5±.1	61.0±.2	74.0±.2	54.6±.1	352.3±0.5
SmoothAP	2.8	$\tau = 0.01$	46.0±.6	76.1±.3	85.9±.3	69.4±.3	0.54±.00	33.8±.3	64.1±.1	76.0±.2	58.0±.2	382.0±1.1

Upshot i

1. The Triplet loss SH results in the best evaluation scores, regardless of dataset, method or task.
2. The Triplet loss SH consistently outperforms the general Triplet loss.
3. The NT-Xent loss consistently underperforms compared to the Triplet loss SH. This is in contrast with findings by Chen et al. [3].
4. Only for the VSE++ method on the i2t task, SmoothAP performs similar to the Triplet loss SH.
5. SmoothAP does not outperform the Triplet loss SH. This is in contrast with the findings by Brown et al. [1].

A Method for Analyzing the Behavior of Loss Functions

Counting Contributing Samples i

- **The question is**: Why do these loss function result in different results, even though the training set-up is the same?

$$\mathcal{L}^{\mathbf{q}}_{TripletSH} = \max(\alpha - s^{+} + s^{-}, 0),$$
$$\mathcal{L}_{TripletSH} = \sum_{\mathbf{q}\in\mathcal{B}} \mathcal{L}^{\mathbf{q}}_{Triplet}.$$

$$\mathcal{L}^{\mathbf{q}}_{Triplet} = \sum_{s^{-}\in S^{q}_{\mathcal{N}}} \max(\alpha - s^{+} + s^{-}, 0),$$
$$\mathcal{L}_{Triplet} = \sum_{\mathbf{q}\in\mathcal{B}} \mathcal{L}^{\mathbf{q}}_{Triplet}.$$

Counting Contributing Samples ii

$$\frac{\partial \mathcal{L}^{\mathbf{q}}_{TripletSH}}{\partial \mathbf{q}} = \begin{cases} \mathbf{v}^{+} - \mathbf{v}^{-}, & \text{if } s^{+} - s^{-} < \alpha \\ 0, & \text{otherwise.} \end{cases}$$

$$\frac{\partial \mathcal{L}^{\mathbf{q}}_{Triplet}}{\partial \mathbf{q}} = \sum_{\mathbf{v}^{-} \in \mathcal{N}_{\mathbf{q}}} \mathbb{1}\{s^{+} - s^{-} < \alpha\} \left(\mathbf{v}^{+} - \mathbf{v}^{-}\right).$$

- Remember: $s^{+} - s^{-} = \mathbf{q}\mathbf{v}^{+} - \mathbf{q}\mathbf{v}^{-}$
- Apparently, the number of triplets (i.e. samples) is causing the difference in evaluation score.
- **Intuition**: The number of triplets/samples should influence the final evaluation scores.

Counting Contributing Samples iii

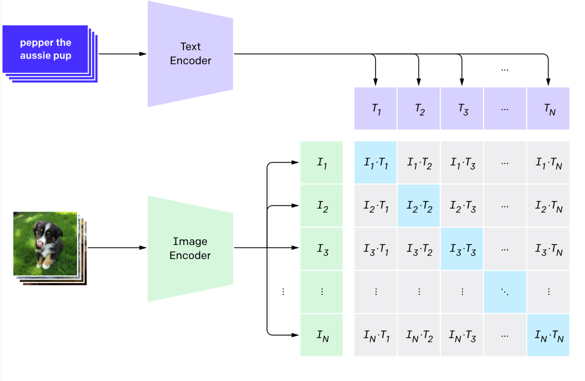

Figure 4: Contrastive Image-Caption Retrieval framework. Source: `https://openai.com/blog/clip/`.

Counting Contributing Samples iv

$$C^{\mathbf{q}}_{Triplet} = \sum_{s^- \in \mathcal{S}^{\mathbf{q}}_{\mathcal{N}}} \mathbb{1}\{s^+ - s^- < \alpha\}$$

$$C^{\mathcal{B}}_{TripletSH} = \textstyle\sum_{\mathbf{q} \in \mathcal{B}} \mathbb{1}\{s^+ - s^- < \alpha\},$$

1. By counting the number of candidates that contribute to the gradient w.r.t. **q**, we aim to get a better understanding of why a certain loss function performs better than others.
2. We propose *counting contributing samples* (COCOS).
3. We hypotheses that there is correlation between the evaluation score and the number of triplets that contribute to the gradient.

Counting Contributing Samples v

- We take the model checkpoint we also use for evaluation.
- We freeze all model parameters.
- We randomly iterate over the train set and count the values for $C^{\mathbf{q}}_{Triplet}$ and $C^{\mathcal{B}}_{TripletSH}$
    - We need to sample batches, to compute $C^{\mathbf{q}}_{Triplet}$ and $C^{\mathcal{B}}_{TripletSH}$

Table 3: COCOS w.r.t. query **q**, for the Triplet loss and the Triplet loss SH.

				i2t			t2i		
			#	$C^{\mathbf{q}}$	$C^{\mathcal{B}}$	C^{o}	$C^{\mathbf{q}}$	$C^{\mathcal{B}}$	C^{o}
Flickr30k	VSE++	Triplet loss	1.1	6.79±0.83	768.92±96.87	14.78±3.52	6.11±0.75	774.67±98.05	1.14±1.22
		Triplet loss SH	1.2	1±0.0	98.74±4.83	29.23±4.81	1±0.0	98.22±4.66	29.75±4.62
	VSRN	Triplet loss	1.5	1.39±0.12	60.96±10.30	84.29±5.80	1.28±0.10	61.21±10.01	80.15±6.35
		Triplet loss SH	1.6	1±0.0	45.59±5.93	82.39±5.92	1±0.0	44.98±5.70	82.99±5.70
MS-COCO	VSE++	Triplet loss	2.1	3.51±0.49	353.82±52.71	27.09±4.60	2.94±0.36	341.64±50.80	12.24±4.92
		Triplet loss SH	2.2	1 ±0.0	88.17±5.25	39.82±5.24	1±0.0	87.24±5.34	40.75±5.33
	VSRN	Triplet loss	2.5	1.21±0.13	29.88±7.46	103.33±5.22	1.15±0.10	30.25±7.49	101.70±5.58
		Triplet loss SH	2.6	1±0.0	33.24±5.39	94.73±5.45	1±0.0	32.90±5.35	95.08±5.4

Upshot: The Triplet loss takes way more negatives into account than the Triplet loss SH. Hence, lower evaluation scores.

How to compute COCOS for the other loss functions?

Counting Contributing Samples viii

$$\mathcal{L}_{NT\text{-}Xent} = -\frac{1}{|\mathcal{B}|} \sum_{\mathbf{q} \in \mathcal{B}} \log \frac{\exp(s^+/\tau)}{\sum_{s_i \in \mathcal{S}_\Omega^{\mathbf{q}}} \exp(s_i/\tau)},$$

$$\frac{\partial \mathcal{L}^{\mathbf{q}}_{NT\text{-}Xent}}{\partial \mathbf{q}} = \left(1 - \frac{\exp(s^+/\tau)}{Z(\mathbf{q})}\right) \tau^{-1} \mathbf{v}^+ - \sum_{s^- \in S_{\mathcal{N}}^q} \left(\frac{\exp(s^-/\tau)}{Z(\mathbf{q})}\right) \tau^{-1} \mathbf{v}^-,$$

Counting Contributing Samples ix

$$C^{\mathbf{qv}^-}_{NT-Xent} = \sum_{s^- \in \mathcal{S}^{\mathbf{q}}_{\mathcal{N}}} \mathbb{1}\{\frac{\exp(s^-/\tau)}{Z(\mathbf{q})} > \epsilon\} \tag{6}$$

- **Intuition**: We count the number of negative candidates with a weight value bigger than $\epsilon = 0.01$.

Counting Contributing Samples x

Table 4: COCOS w.r.t. query **q**, for the NT-Xent loss [3].

			i2t			t2i		
		#	$C^{\mathbf{qv}^-}_{NT-Xent}$	$W^{\mathbf{qv}^-}_{NT-Xent}$	$W^{\mathbf{qv}^+}_{NT-Xent}$	$C^{\mathbf{qv}^-}_{NT-Xent}$	$W^{\mathbf{qv}^-}_{NT-Xent}$	$W^{\mathbf{qv}^+}_{NT-Xent}$
Flickr30k	VSE++	1.3	9.88±0.51	0.42±0.02	0.56±0.02	9.65±0.51	0.42±0.02	0.56±0.02
	VSRN	1.7	2.45±0.23	0.13±0.02	0.20±0.02	2.46±0.23	0.13±0.02	0.20±0.02
MS-COCO	VSE++	2.3	5.59±0.40	0.36±0.02	0.46±0.02	5.33±0.38	0.36±0.02	0.46±0.02
	VSRN	2.7	1.10±0.14	0.10±0.02	0.14±0.02	1.11±0.14	0.09±0.02	0.14±0.02

Upshot: The NT-Xent loss takes also more than 1 negative into account. Hence, lower evaluation scores.

Counting Contributing Samples xii

$$C^{\mathbf{q}}_{Smooth} = \frac{1}{|\mathcal{S}^{\mathbf{q}}_{\mathcal{P}}|} \sum_{i \in \mathcal{S}^{\mathbf{q}}_{\mathcal{P}}} \left(\sum_{j \in \mathcal{S}^{\mathbf{q}}_{\mathcal{N}}} \mathbb{1}\left\{ \frac{sim(D_{ij})}{\mathcal{R}(i, \mathcal{S}^{\mathbf{q}}_{\Omega})^2} > \epsilon \right\} + \sum_{j \in \mathcal{S}^{\mathbf{q}}_{\mathcal{P}}, j \neq i} \mathbb{1}\left\{ \frac{sim(D_{ij})}{\mathcal{R}(i, \mathcal{S}^{\mathbf{q}}_{\Omega})^2} > \epsilon \right\} \right). \tag{7}$$

- **Intuition**: We count the number of samples that are very close to each other in terms of similarity score (i.e. which can change the ranking).

Table 5: COCOS w.r.t. query **q**, for the SmoothAP [1] loss.

			i2t		t2i	
		#	$C^{\mathbf{q}}_{SmoothAP}$	$C^{\mathrm{o}}_{SmoothAP}$	$C^{\mathbf{q}}_{SmoothAP}$	$C^{\mathrm{o}}_{SmoothAP}$
Flickr30k	VSE++	1.4	1.27±0.06	2.15±1.51	1.47±0.83	636.72±18.72
	VSRN	1.8	2.33±0.07	0.00±0.00	1.62±0.95	636.49±18.65
MS-COCO	VSE++	2.4	1.48±0.07	0.80±0.90	1.41±0.74	637.10±20.28
	VSRN	2.8	1.67±0.07	0.14±0.37	1.42±0.76	637.23±20.35

Counting Contributing Samples xiv

Upshot:

1. The gradient for (most) metric learning functions is just a sum over positive and negative candidates.
2. The number of negative samples that is taken into account when computing the gradient has an effect on the final evaluation score(s)

Discussion and Conclusions

Discussion i

- **Limitation:** Can we just use loss functions as an off-the-shelf tool, without any additional hyper-parameter tuning?
  - Musgrave er al. [7] also show that metric learning functions generalize quite badly to different training settings.
- **Limitation:** Counting samples that contribute to the gradient based on a weight value is quite non-trivial.
- **Future research:** Design loss functions using the principle using the COCOs principles.
- **Future research:** The moment of counting during training also matters a lot.
- **Future research:** Extend the idea of COCOS to include more loss functions, or to other domains (such as DPR).

Conclusions i

1. We tried three different loss functions for the ICR task.
   - The Triplet loss with semi hard-negatives still results in the highest evaluation performances.
2. We introduce COCOS.
   - **Underlying idea**: most metric learning functions, in the end, are a weighted sum of positive and negative samples.
   - **Goal**: An approach to analyze and unify metric learning functions.
3. We have shown that the best performing loss function only focuses on one (hard) negative sample when computing the gradient.
4. This suggests that the underperforming loss functions take too many (non-informative) negatives into account, and therefore converge to a sub-optimal point.

https://github.com/MauritsBleeker/ecir-2022-reproducibility-bleeker

Thanks for your attention!
Are there any questions?

{m.j.r.bleeker,m.derijke}@uva.nl

@MauritsBleeker

mauritsbleeker.github.io

References

[1] A. Brown, W. Xie, V. Kalogeiton, and A. Zisserman. Smooth-ap: Smoothing the path towards large-scale image retrieval. In *European Conference on Computer Vision*, pages 677–694. Springer, 2020.

[2] T. Chen, J. Deng, and J. Luo. Adaptive offline quintuplet loss for image-text matching. *arXiv preprint arXiv:2003.03669*, 2020.

[3] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

[4] F. Faghri, D. J. Fleet, J. R. Kiros, and S. Fidler. Vse++: Improving visual-semantic embeddings with hard negatives. 2018. URL `https://github.com/fartashf/vsepp`.

[5] C. Jia, Y. Yang, Y. Xia, Y.-T. Chen, Z. Parekh, H. Pham, Q. V. Le, Y. Sung, Z. Li, and T. Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. *arXiv preprint arXiv:2102.05918*, 2021.

[6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[7] K. Musgrave, S. Belongie, and S.-N. Lim. A metric learning reality check. In *European Conference on Computer Vision*, pages 681–699. Springer, 2020.

[8] A. v. d. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[9] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics*, 2:67–78, 2014.

[10] L. Yuan, D. Chen, Y.-L. Chen, N. Codella, X. Dai, J. Gao, H. Hu, X. Huang, B. Li, C. Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.