2018 IEEE Winter Conference on Applications of Computer Vision

Fast Self-Attentive Multimodal Retrieval

Jônatas Wehrmann Maurício A. Lopes Martin D. More Rodrigo C. Barros
Machine Intelligence and Robotics Research Group
Pontifícia Universidade Católica do Rio Grande do Sul
Av. Ipiranga, 6681, Porto Alegre, RS, Brazil
{jonatas.wehrmann,mauricio.armani,martin.more}@acad.pucrs.br, rodrigo.barros@pucrs.br

Abstract

Multimodal bidirectional retrieval is a very challenging task that consists of semantically aligning two distinct modalities such as images and textual descriptions, allowing the retrieval of content from one of the modalities given the other. The goal is to find a common semantic space for both modalities in order to discover the correspondences between them. This paper provides a fast and effective attention-based architecture for learning representations for multimodal retrieval, namely SEAM. It is based on a self-attention module that is designed to enhance relevant textual information from word-embeddings while suppressing irrelevant data. We design three incarnations of SEAM, so we can properly assess the performance of the self-attention module when operating over distinct representations, namely: (i) the word-embeddings themselves; (ii) features learned from convolutional layers in distinct granularities; and (iii) features learned from gated recurrent units (GRUs). The output of the self-attention module is projected over a shared multimodal space, so we can map the semantic correspondence between images and descriptions via a contrastive pairwise loss function that minimizes order-violations. We analyze several architectural choices for our approach, and we compare our best models with the current state-of-the-art approaches in the largest and most well-known multimodal retrieval dataset, namely Microsoft COCO. Results show that SEAM outperforms the current state-of-the-art in most cases while being a much faster approach for the task of multimodal retrieval.

1. Introduction

End-to-end fully-differentiable computational graphs (or simply *neural networks*) are known to be the current state-of-the-art approach for most applications that are based on modes like image, video, audio, or text. Examples include supervised image classification [20], object detection [12], semantic segmentation [4], speech recognition [28], video classification [27, 18], text classification [7, 25], text summarization [29], protein function prediction [24], and image captioning [22]. Borrowing concepts from neuroscience, artificial neural networks comprise a mathematical framework capable of assigning meaning to what is seen, heard, or read, being known as an effective method for performing *representation learning* over unstructured data [3].

One of the computer vision problems that has benefited from the recent advances in neural networks research is the so-called multimodal (bidirectional) retrieval, also regarded as bidirectional content retrieval or image-text alignment. In that scenario, the main target is to retrieve content from a modality (e.g., image) given some input content from another modality (e.g., textual description). Most state-of-the-art results for bidirectional retrieval [21, 2, 6, 23] are based on networks trained over word-embeddings [16], encoding the sentences with either Recurrent Neural Networks (RNNs such as LSTMs [5] or GRUs [1]) or with handcrafted non-linear transformations such as Fisher vectors [11]. Those strategies, while often showing good results, are quite costly due to the amount of storage and memory for dealing with the word-embeddings, depending on the size of the dictionary, in which often larger is better in terms of predictive accuracy. A possible approach for reducing the amount of memory is to perform character-level convolutions [26] in order to avoid storing a dictionary of word-embeddings. Despite being lighter in both parameters and memory consumption, character-level convolutions are not fast to train, since the character-based embeddings and their complex relationships have to be learned from scratch.

For properly addressing the problems above-mentioned with an interesting trade-off between memory consumption and training speed, we propose a novel neural network architecture, namely *SEAM* (Self-attentive Embeddings for Aligning Modalities). It considers each input as a $n \times d$ matrix, in which n is the temporal dimension (total number of words in the textual description, indicating the ordered flow of the words) and d is the word-embedding vector that densely projects the respective word onto a high-

978-1-5386-4886-5/18/$31.00 ©2018 IEEE
DOI 10.1109/WACV.2018.00207

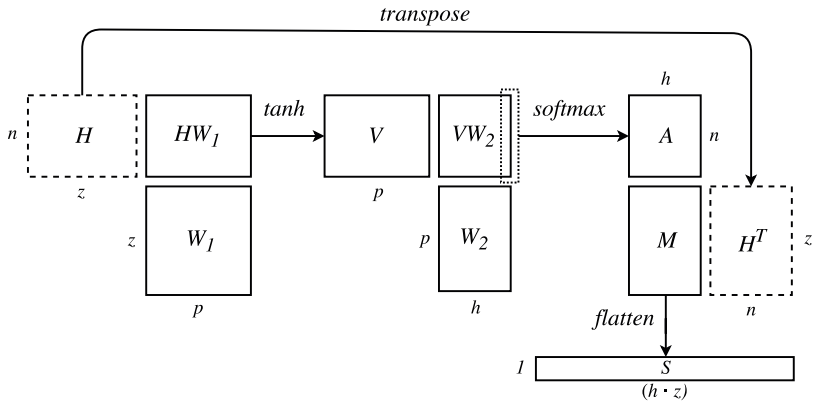

Figure 1. Self-Attention Mechanism

dimensional feature space. We employ a self-attention mechanism so the network can automatically learn the importance of the features from each word in each temporal step. We design three incarnations of *SEAM*, so we can properly assess the performance of the self-attention model when operating over distinct representations, namely: (i) the word-embeddings themselves, (*SEAM*-E); (ii) convolutional features in distinct granularities (*SEAM*-C); and (iii) features learned from Gated Recurrent Units (*SEAM*-G).

The output of the self-attention module is projected over a shared multimodal space, so the architecture is capable of mapping the semantic correspondence between images and descriptions via a contrastive pairwise loss function that minimizes order-violations. We compare our best models with the current state-of-the-art approaches that employ either RNNs and word-embeddings, or character-level convolutions for image-text alignment. We make use of the largest and most well-known multimodal retrieval dataset, namely Microsoft COCO [13]. Results show that *SEAM* outperforms the current state-of-the-art in most cases while being a much faster approach for the task of multimodal retrieval.

This paper is organized as follows. Section 2 describes in detail our proposed approach, whereas Sections 3 and 4 present the experimental analysis that was conducted for validating our novel method. Section 5 discusses related work in the area of multimodal retrieval. Finally, we end this paper with our conclusions and suggestions for future work in Section 6. Source code and pre-computed image features are publicly available [1].

2. *SEAM*

In this paper we present *SEAM* (Self-attentive Embeddings for Aligning Modalities), an approach for solving image-text retrieval tasks. *SEAM* relies on the self-attention module proposed in [14], which analyzes different views (or *hops*) to automatically infer the importance of features through time. Our proposed architectures are simple, fast and effective in learning image-text similarity from word-embeddings and image features extracted from deep nets.

The first two incarnations of *SEAM* rely on using self-attention and/or convolutions to extract relevant information from the word-embeddings that represent a sentence. This allows us not to use RNNs, increasing processing speed due to paralellization advantages. In the last incarnation, we attempt to increase the quality of our embedding results by first running the word-embeddings through GRUs and then processing that output with the self-attention mechanism. Next, we detail the self-attention module and the three versions of the proposed approach, namely *SEAM*-[E,C,G].

2.1. Self-attention Module

The self-attention module is the core component within all variations of *SEAM*. It is used to extract different components of a sentence into multiple vector representations while encouraging relevant information and suppressing irrelevant data. Self-attention can be interpreted as being a two-layer neural network that ultimately learns weights within $[0, 1]$ for the features at each time-step (see Figure 1).

Let $H \in \mathbb{R}^{n \times z}$ be a dense representation of the textual input, where n denotes the number of time-steps (e.g., words), and z the size of the feature vectors. Depending on the incarnation, z can be equal to d (the dimension of the word-embedding), or equal to the dimension resulting from a convolutional layer, or equal to the dimension resulting from the application of a recurrent neural network. The first step of the attention module consists in applying a transformation using a fully-connected layer to reduce the dimensionality to p-dimensional feature vectors. Values are processed by a $tanh$ activation function (denoted by ζ), that projects them into the $[-1, 1]$ range, generating $V \in \mathbb{R}^{n \times p}$:

$$V = \zeta(HW_1) \tag{1}$$

[1] https://github.com/jwehrmann/seam-retrieval

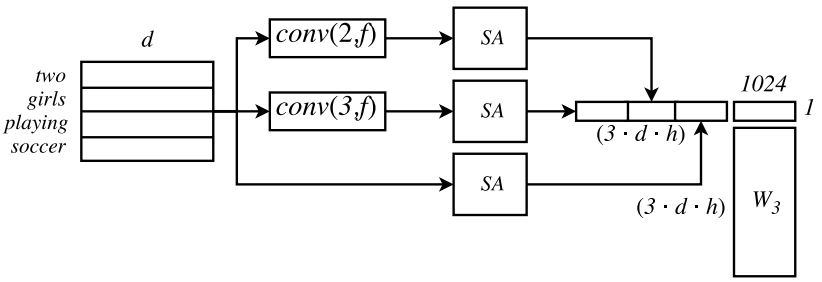

Figure 2. *SEAM*-C

Following, we use an additional fully-connected layer with h neurons, generating an output of size $n \times h$. Softmax is used to ensure that the weighted sum of all features in each temporal step (represented by the columns of the matrix) results in 1. The resulting weight map, denominated annotation matrix, is denoted by $A \in \mathbb{R}^{n \times h}$. Therefore, A can be seen as a matrix that carries the importance of each time-step in h different viewpoints (or *hops*).

$$A = softmax(VW_2) \tag{2}$$

The next step is to use the annotation matrix A and the original dense representation H to compute the weighted feature map $M \in \mathbb{R}^{z \times h}$, which is given by $M = H^T A$. Finally, we can reshape M using a flatten operation to transform it into S, a vector embedding with size $\mathbb{R}^{1 \times (z \times h)}$.

In order to prevent redundancy problems (i.e., similar summation weights inside *hops*), the self-attention mechanism introduces a penalization term, presented in Eq. 3, that seeks to encourage diversity across time-steps. The practical effect is that the probability mass for each vector in the annotation Softmax output will be focused on as few words as possible.

$$P = ||(AA^T - I)||_F^2 \tag{3}$$

The $||\bullet||_F$ term stands for the Frobenius norm of a matrix. This penalization term is multiplied by a coefficient λ within the interval $[0, 1]$ and is minimized alongside the retrieval loss function.

2.2. *SEAM*-E

Our first approach makes use of the self-attention module directly over word-embedding representations. This assumes that the word-embedding itself already contains all information needed for interpreting the textual description. Since the attention module already offers different viewpoints for weighting words over time, we believe it to be enough to aggregate the necessary temporal semantics for the actual task of generating representative embeddings for sentences. Even if the amount of temporal information is constrained, this strategy is significantly faster and reduces model complexity due to having less trainable parameters. The extracted vector of the attention mechanism ($1 \times d.h$) is projected to match the dimensions of the image embedding representation via a linear layer with 1024 neurons.

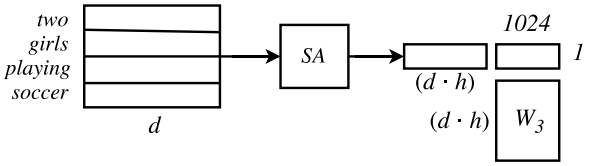

Figure 3. *SEAM*-E

2.3. *SEAM*-C

Our second approach is a somewhat modified version of the architectures introduced in [9]. In summary, *SEAM*-C applies convolutional layers directly over word-embeddings and then processes each individual result with the self-attention module to generate the final embedding (see Figure 2). Since the convolutional layers have distinct filter sizes fs, this allows us to learn multiple n-gram-like features (specifically, bigrams and trigrams).

SEAM-C employs two convolutional layers with filter sizes $fs \in \{2, 3\}$ on a zero-padded input matrix H. The padding is particularly important so that we can generate feature maps with matching dimensions, which will in turn be processed by the self-attention module individually. Let $\psi(H) = C$ be the computation of a convolutional layer applied over the input H with a resulting dimension $\mathbb{R}^{n \times f}$, where f represents the number of filters. We apply the self-attention scheme individually for each C, obtaining $\mathbb{R}^{n \times (d \cdot h)}$ feature matrices. Unlike [9], we also process the input matrix H with self-attention, which allows us to select relevant information not only from temporal data, but also from individual words. The final embedding is the concatenation of the results of applying the self-attention mechanism to the convolutional features and the original matrix, which is a vector with dimensions $1 \times 3(d \cdot h)$.

2.4. *SEAM*-G

Our third method uses a gated recurrent unit network (GRU) and processes the result with the self-attention

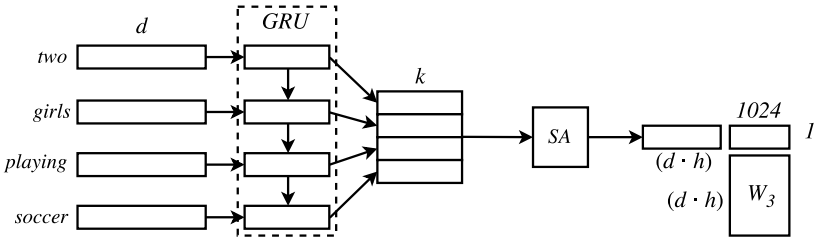

Figure 4. *SEAM*-G

mechanism (see Figure 4). The GRU provides k-long output vectors (considering k neurons in the hidden layer) for each of the n word-vectors given as inputs. Each output contains information on all previous inputs. GRUs (or similar RNNs) are particularly well-suited for this scenario since they are capable, together with the self-attention module, to weigh different text snippets and to infer which snippets are more relevant to the image-text alignment process.

Let $\mathbf{h}_i$ be the i-th hidden layer activations for a n-sized sentence. $\mathbf{G} \in \mathbb{R}^{n \times k}$ represents the concatenation of [$\mathbf{h}_0$, $\mathbf{h}_1$, ..., $\mathbf{h}_n$] through time. $\mathbf{G}$ is then used as input in the self-attention mechanism to compute the ultimate feature vector representation, as shown in Figure 4.

2.5. Overall Architecture

We approximate two encoding functions, $f_t(t)$ and $f_i(i)$, whose goals are to project both description t and image i into the same embedding space. In such a space, correlated image-text pairs should be close to each other, and the distance of non-correlated pairs should necessarily be larger than the correlated ones.

For the image encoding function $f_i(i)$, we extract image features from the global-pooling layer of a VGG-19 (or Inception-ResNet-C [IRC]) [19, 20] pre-trained in the ImageNet dataset [17]. Each image i is then represented by 4096 (1536)-dimensional vectors, extracted using the 10-crop strategy. Let $\mathcal{C}(i)$ be features extracted from image i by the convolutional neural network; images are projected onto the $\mathcal{R}^d_+$ embedding-space based on a linear mapping:

$$f_i(i) = |W_i \cdot \mathcal{C}(i)| \tag{4}$$

where $W_i \in \mathcal{R}^{d \times 4096(1536)}$ is a learned weight matrix and d is the number of dimensions of the embedding space.

For embedding text, we use the proposed attention-based approach $f_t(\cdot)$ with the three main variations previously discussed. Our models provide a 1024-long vector representation that carries the textual semantic information, which is the result of linearly projecting the output of the self-attention module onto $\mathcal{R}^d_+$ by using a learned $W_t \in \mathcal{R}^{d \times 1024}$ weight matrix.

2.6. Loss function

Let $f_t(t) = \mathcal{T}$ represent the text embedding vector and $f_i(i) = \mathcal{I}$ be the image embedding vector. To obtain the similarity between $\mathcal{T}$ and $\mathcal{I}$, we first scale both to have unit norm, so that the inner product of both results in the cosine distance. However, instead of directly optimizing the cosine distance, as in [10], we follow the adaptations proposed by [21] and instead learn order-embeddings by optimizing the alignment of text and images while preserving the order relationships among the visual-semantic hierarchy, given that asymmetric distances are naturally more well-suited for image-sentence alignment. This translates to applying the following order-violation penalty for an ordered pair: $S(t,i) = -||max\{0, \mathcal{I} - \mathcal{T}\}||^2$. These order violation penalties are used as a similarity distance and optimized by the following contrastive pairwise ranking loss:

$$\begin{aligned}\mathcal{L}_c = &\sum_{\mathcal{T}}\sum_{k} max\{0, \alpha - S(\mathcal{T},\mathcal{I}) + S(\mathcal{T},\mathcal{I}_k)\} \\ &+ \sum_{\mathcal{I}}\sum_{k} max\{0, \alpha - S(\mathcal{I},\mathcal{T}) + S(\mathcal{I},\mathcal{T}_k)\}\end{aligned} \tag{5}$$

where $\mathcal{T}_k$ and $\mathcal{I}_k$ are the sentence and image contrastive examples (i.e., uncorrelated). This loss function encourages the similarity $S(t,i)$ for proper image-text pairs to be larger than the contrastive pairs by a margin of at least α. Since we are using self-attention in *SEAM*, we include its penalization term, described in Sec. 2.1, which is given by Eq. 3. Our final loss function, therefore, is given by $\mathcal{L}_f = \mathcal{L}_c + \lambda\mathcal{P}$.

3. Experimental Setup

3.1. Dataset

For analyzing the performance of our proposed approach, we make use of the Microsoft COCO dataset [13]. It contains over 100,000 images with at least 5 descriptions per image. We have used the same data splits from [8]: 113,287 images for training, 5,000 images for validation, and 5,000 images for testing.

MS COCO has been extensively employed in the recent years for image-text retrieval challenges. Note that, for the $5k$ images in the test set, there are three distinct evaluation

protocols employed by the research community, because the test images were further divided into 5 folds of $1k$ images each. Some studies present results on the entire test set of $5k$ images, a protocol we refer to as COCO-$5k$; and others present results only for a subset of $1k$ images, which we refer to as COCO-$1k$.

3.2. Hyper-Parameters and Training Details

We choose hyper-parameters via non-exhaustive random search based on the results over the validation data. We employ Adam for optimization, given its capacity in adjusting per-weight learning rates during training. We use Adam's default initial learning rate of 1×10^{-3}. In addition, we found it was beneficial to reduce the learning rate by $10\times$ in the 15th epoch. Inspired by [21], we use a batch size of 128 (127 contrastive examples) and margin $\alpha = 0.05$. Neither weight decay nor dropout were used, since we believe the loss function itself is enough to regularize the model by including several contrastive examples which naturally injects some amount of noise during training.

For all versions of *SEAM*, both embedding size d and the intermediate dimension of the self-attention module p are set to 300. For *SEAM*-E, we vary the number of hops $h \in \{10, 15, 20, 30\}$. For *SEAM*-C, $f = 100$ for both convolutional layers. There are three attention modules whose outputs are concatenated, hence we vary the number of hops $h \in \{5, 7, 10, 15, 20\}$ for each module. Recall that z can be either 100 — when the input to the attention module is the output of a convolution — or 300 for the attention over the word-embeddings. For *SEAM*-G, we vary the number of neurons of the GRU in $\{256, 512, 1024\}$, and fix the number of hops $h = 30$.

3.3. Evaluation Measures

For evaluating the results, we use the same measures as those in [21]: $R@K$ (reads "Recall at K") is the percentage of queries in which the ground-truth term is one of the first K retrieved results. The higher its value, the better. We also show the results of *Med* r and *Mean* r, which represent respectively the median and mean of the ground-truth ranking. Since they are ranking-based measures, the smaller their values the better.

4. Experimental Analysis

In this section, we provide a thorough analysis of the performance of our proposed approach. First, we analyze the impact of different architectural choices for *SEAM* by looking exclusively to results on validation data. Then, we compare our best approach with the state-of-the-art in bidirectional retrieval (results over the test set).

4.1. Impact of the Self-Attention Coefficient λ

Table 1 shows the impact of the attention coefficient on the results over validation data. Such value is responsible for improving the diversity in the self-attention mechanism. We vary $\lambda \in \{0, 0.25, 0.50, 0.75, 1\}$ for both *SEAM*-E and *SEAM*-C. Note that there is no single value for λ that provides the best results for all evaluation measures. Intermediate values like 0.5 and 0.75 seem to be a solid choice, though $\lambda = 0$ is also a reasonable option. A large penalization such as $\lambda = 1$ does not yield good results, mostly due to the fact that λ may constraint the training process, since it can force the optimization to focus more on diversity over the alignment quality.

Table 1. Impact of the attention coefficient λ. Bidirectional results on COCO validation set ($1000k$ images). Bold values indicate the best results per method.

		Image to text			Text to image		
Method	λ	R@1	R@10	Mean r	R@1	R@10	Mean r
SEAM-E	0.00	**53.30**	92.00	**4.40**	41.80	87.80	7.20
SEAM-E	0.25	51.40	91.80	**4.40**	41.10	88.00	6.80
SEAM-E	0.50	51.60	**92.30**	4.50	41.20	**88.30**	**6.40**
SEAM-E	0.75	50.90	91.40	**4.40**	**42.00**	88.00	**6.40**
SEAM-E	1.00	52.70	91.70	4.50	41.00	88.20	6.80
SEAM-C	0.00	52.90	92.40	**4.00**	42.20	**89.20**	6.30
SEAM-C	0.25	54.30	92.20	4.20	42.20	88.80	6.70
SEAM-C	0.50	**55.30**	90.80	4.20	42.60	89.10	6.50
SEAM-C	0.75	53.50	**92.70**	4.30	**42.90**	88.60	6.60
SEAM-C	1.00	50.70	90.80	4.50	39.30	88.80	**5.80**

4.2. Impact of h

Next we analyze the impact of the number of hops in the attention module. We show in Table 2 the performance of both *SEAM*-E and *SEAM*-C with different h values.

Table 2. Impact of h. Bidirectional results on COCO-$1k$ test set. Bold values indicate the current state-of-the-art results. Underlined values outperform the best published results.

		Image to text		Text to image	
Method	#Params	R@1	Mean r	R@1	Mean r
SEAM-E (h=10)	6,574,024	52.90	4.50	41.80	**6.50**
SEAM-E (h=15)	8,111,524	50.50	4.60	41.60	6.90
SEAM-E (h=20)	9,649,024	52.30	4.80	**42.20**	7.00
SEAM-E (h=30)	12,724,024	**53.30**	**4.40**	41.80	7.20
SEAM-C (h=5)	6,273,724	**54.60**	**4.00**	42.00	6.50
SEAM-C (h=7)	7,299,524	52.40	4.30	42.50	**6.10**
SEAM-C (h=10)	8,838,224	52.90	**4.00**	42.20	6.30
SEAM-C (h=15)	11,402,724	53.50	4.10	**42.60**	6.70
SEAM-C (h=20)	13,967,224	53.30	**4.00**	42.30	**6.10**

Recall that the hops h represent different viewpoints as you weigh the words throughout the sentence. Results show that the performance of *SEAM*-E for the image-to-text task improves with a larger number of hops ($h = 30$), though there is no clear pattern in the text-to-image task. For *SEAM*-C, a small number of hops suffice for the image-to-text task, and once again there does not seem to exist a clear

Table 3. Bidirectional results on COCO-1k test set. Bold values indicate the current state-of-the-art results. Underlined values outperform the best published results.

Method	ConvNet	Image to text					Text to image				
		R@1	R@5	R@10	Med r	Mean r	R@1	R@5	R@10	Med r	Mean r
UVS [10]	VGG-19	43.40	-	85.80	2.00	-	31.00	-	79.90	3.00	-
Embedding Network [23]	VGG-19	50.40	79.30	89.40	-	-	39.80	75.30	86.80	**2.00**	-
sm-LSTM [6]	VGG-19	52.40	81.70	90.80	**1.00**	-	38.60	73.40	84.60	**2.00**	-
2WayNet [2]	VGG-19	**55.80**	-	75.20	-	-	39.70	-	63.30	-	-
Order [21] [Ours]	VGG-19	49.30	78.50	89.40	2.00	5.60	39.50	75.00	86.20	**2.00**	7.50
SEAM-E	VGG-19	52.70	83.20	91.20	**1.00**	4.40	41.40	76.00	86.80	**2.00**	7.20
SEAM-E($\lambda = 0.5$)	VGG-19	50.60	82.80	91.50	**1.00**	4.50	41.70	76.20	87.40	**2.00**	7.20
SEAM-C	VGG-19	51.80	81.70	91.00	**1.00**	4.70	41.50	76.70	88.00	**2.00**	6.90
SEAM-C($\lambda = 0.5$)	VGG-19	54.30	82.70	91.90	**1.00**	4.80	41.80	76.70	87.80	**2.00**	7.20
SEAM-G (512)	VGG-19	52.10	83.10	91.00	**1.00**	4.60	41.80	76.80	87.60	**2.00**	7.00
Order [21][Ours]	IRv2	50.10	84.50	92.80	**1.00**	4.30	40.00	76.30	88.20	**2.00**	6.70
SEAM-E	IRv2	54.20	85.60	92.70	**1.00**	3.60	43.30	78.80	90.00	**2.00**	6.70
SEAM-E($\lambda = 0.5$)	IRv2	55.20	84.70	93.90	**1.00**	**3.50**	43.90	79.40	89.50	**2.00**	7.10
SEAM-C	IRv2	54.50	85.40	93.70	**1.00**	3.70	43.70	79.60	89.60	**2.00**	6.60
SEAM-C($\lambda = 0.5$)	IRv2	55.40	**86.40**	**94.40**	**1.00**	3.60	43.80	**79.80**	90.10	**2.00**	6.90
SEAM-G (512)	IRv2	53.60	85.60	93.20	**1.00**	3.70	**44.60**	**79.80**	**90.30**	**2.00**	**6.50**

Table 4. Bidirectional results on COCO-5cv test set. Bold values indicate the current state-of-the-art results.

Method	ConvNet	Image to text					Text to image				
		R@1	R@5	R@10	Med r	Mean r	R@1	R@5	R@10	Med r	Mean r
Order [21]	VGG-19	46.70	-	88.90	2.00	-	37.90	-	85.90	2.00	-
OECC [26]	VGG-19	47.20	78.60	88.90	2.00	5.60	37.50	74.60	87.00	2.00	7.30
Order [Ours]	VGG-19	47.20	82.40	91.60	2.00	4.40	37.90	74.70	87.00	2.00	7.30
SEAM-E	VGG-19	51.20	81.40	90.50	1.20	5.00	39.70	75.30	86.70	2.00	7.60
SEAM-E($\lambda = 0.5$)	VGG-19	49.40	80.70	90.30	1.80	5.10	39.70	75.30	87.00	2.00	7.40
SEAM-C	VGG-19	50.00	81.10	90.60	1.60	5.10	40.20	75.60	87.10	2.00	7.20
SEAM-C($\lambda = 0.5$)	VGG-19	50.70	81.40	90.90	1.40	4.90	40.30	75.70	87.40	2.00	7.40
SEAM-G (512)	VGG-19	50.90	81.80	90.50	1.40	4.90	40.20	75.70	87.20	2.00	7.40
Order [Ours]	IRv2	48.30	82.20	91.20	1.80	4.50	37.90	74.40	86.90	2.00	7.20
OECC [26]	IRv2	49.50	81.70	91.30	1.60	4.50	40.40	77.40	88.60	2.00	**6.80**
SEAM-E	IRv2	51.80	83.90	92.00	1.40	4.30	42.00	77.20	88.40	2.00	7.40
SEAM-E($\lambda = 0.5$)	IRv2	51.90	83.20	92.20	1.20	**4.00**	41.60	77.30	88.00	2.00	7.40
SEAM-C	IRv2	**52.90**	83.50	**92.60**	**1.00**	4.30	**42.30**	77.90	88.40	2.00	7.70
SEAM-C($\lambda = 0.5$)	IRv2	52.80	83.50	**92.60**	1.20	4.10	41.90	77.50	88.50	2.00	7.70
SEAM-G (512)	IRv2	52.10	**84.00**	92.50	1.20	4.10	41.90	**78.20**	**89.10**	2.00	7.10

pattern in the text-to-image task. We believe that the convolutional layers already introduce different viewpoints with the number of filters, f, so increasing those values does not seem to help. Note that a large number of hops directly affects the amount of trainable parameters, increasing memory consumption and processing time. Therefore, for lighter models one should consider using $h \in \{5, 10\}$ that perform quite similar than the heavier models.

4.3. *SEAM vs.* State-of-the-art

For comparing our models with the state-of-the-art, we selected the models from each *SEAM* variation that presented the best performance on validation data. We compare to the state-of-the-art approaches for multimodal retrieval, namely UVS [10], DVSA [8], FV [11], Order-Embeddings (OE) [21], Embedding Network [23], sm-LSTM [6], 2WayNet [2], and OECC [26]. To provide a fair comparison, we replicated the Order-Embedding [21] and the Character-level Convolutions with Order Embeddings (OECC) [26] results using their own source code.

Table 3 presents the comparison over the COCO-1k test set. We show results of our approach and OE [21] by using two distinct convolutional networks (VGG and IRv2). In general, our methods outperform all baselines in both image retrieval and text retrieval tasks. For all evaluation metrics, the three proposed methods present superior performance both for VGG-19 image features and IRv2. The only exception is for $R@1$ in the image-to-text task, where 2WayNet outperforms all methods. However, note that 2WayNet is only slightly better than *SEAM* $- C(\lambda = 0.5)$, and it performs poorly regarding $R@10$. In addition, all proposed methods easily outperform it in the text-to-image task. Our methods are about $15\times$ faster than the RNN-based methods (see 4.4). In test time, they are much faster than 2WayNet, while presenting one order of magnitude fewer parameters. The same behavior is observed in Table 4, where the comparison is over the COCO-5cv data. Finally, we noticed that the use of a better feature representation (IRv2) highly benefits *SEAM*, while for OE it is quite modest and even marginal in some evaluation metrics.

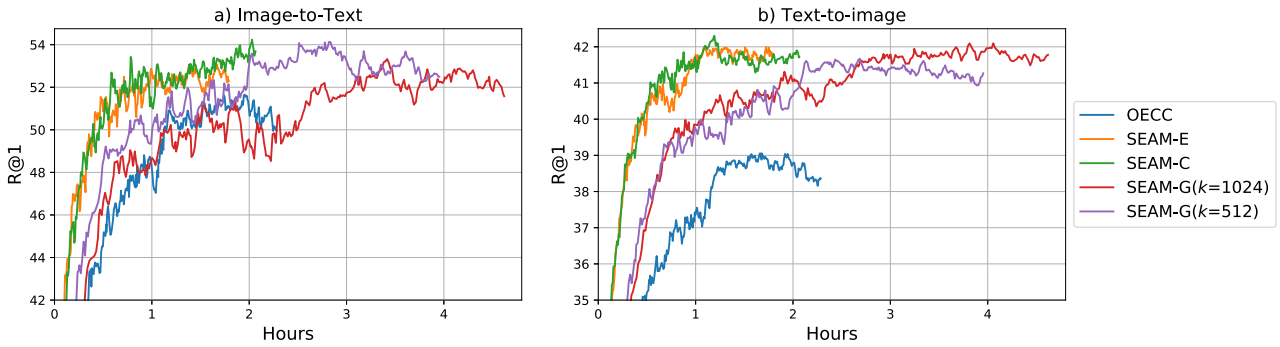

Figure 5. R@1 per hours of training. Note that all methods were executed for the exact same amount of epochs.

4.4. Execution Time

All experiments were executed in a server with four NVIDIA GTX 1080ti GPUs, Intel(R) Core(TM) i7-5930K CPU @ 3.50GHz, CUDA 8, CUDNN 5.1, and 128GB RAM. For providing a fair time comparison, we implemented all methods described in this section using the PyTorch framework. Even though *SEAM*-E could be highly-optimized (as in [7]) for achieving a much faster performance, we used the default PyTorch implementation for allowing direct comparison.

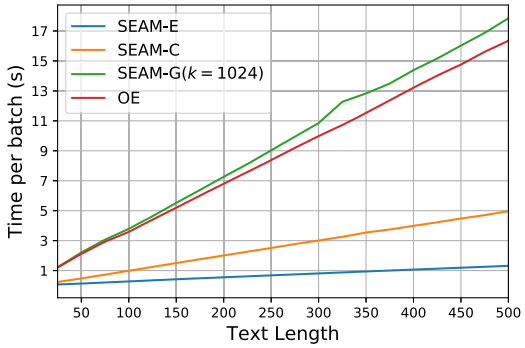

Figure 6. Time (in seconds) taken to perform a forward pass of a 100-instance batch with descriptions of varying sizes.

Figure 5 depicts values of R@1 for both image-to-text (a) and text-to-image (b) tasks across hours of training. Note that the fastest methods, namely *SEAM*-[E,C], reach convergence roughly in two hours, which is about $3\times$ faster than *SEAM*-G (our GRU-based approach, which requires training time similar to OE [21]). They are also the best trade-off in performance (*SEAM*-C is, in fact, the best performing method in terms of R@1). OECC is also much faster than *SEAM*-G and OE, but it is outperformed by all methods in terms of R@1.

Figure 6 shows the time (in seconds) required for encoding textual instances with increasing sizes. We evaluate the impact of the sentences length by varying the number of words $\in \{25, 50, 75, ..., 475, 500\}$. We report the average values of 10 runs of 100-instance batches. Note that *SEAM*-E is capable of encoding a batch with 500-word descriptions in approximately 1 second, whereas OE [21] takes $16\times$ more time to encode similar sentences.

5. Related Work

Karpathy and Fei-Fei [8] propose an architecture that makes use of features from detection-based systems, aligning image regions with a proper sentence fragment. Ma et al. [15] propose a multimodal ConvNet for aligning image and text by jointly convolving word-embeddings and image features. The learned similarity score predicts whether a pair is correlated or not.

Vendrov et al. [21] propose sentence order-embeddings, which aim to preserve the partial order structure of a visual-semantic hierarchy. It allows learning ordered representation by applying order-penalties, and they show that asymmetric measures are better suited for image-sentence retrieval tasks. Their architecture is virtually the same of the one introduced in [10], only modifying the loss function to consider the order violations.

Wang et al. [23] introduce a two-branch neural network for learning a multimodal embedding space. They encode text based on 300-d word-embeddings, where they apply ICA and construct a codebook with 30 centers using first and second-order information, resulting in a $18k$-dimensional representation. Next, they apply PCA to reduce the representation to $6k$ dimensions in order to reduce memory requirements and training time.

Huang et al. [6] propose a selective multimodal LSTM (sm-LSTM). They introduce a multimodal context-modulated attention scheme at each time-step, which is capable of focusing on a text-image pair by predicting pairwise instance-aware saliency maps. Sentences are processed by a bidirectional LSTM that runs over word-embeddings. Image features are selected by using a strategy of instance candidates, which extracts local information from a $512 \times 14 \times 14$ tensor.

In [2], the authors introduce a 2-Way-Network for map-

ping a modality into another. They concatenate the Fisher Vector encoding (GMM) and the Fisher Vector extracted form $word2vec$ vectors of the HGLMM distribution, resulting in a $36k$-dimensional vector per sentence.

6. Conclusions

We presented a fast and effective architecture for bidirectional multimodal retrieval capable of learning self-attentive textual embeddings, namely *SEAM*. Even though it is conceptually a much simpler architecture than those found in related work, our approach achieves state-of-the-art results in both *text-to-image* and *image-to-text* tasks considering the most well-known retrieval dataset, namely MS COCO [13], while being orders of magnitude faster. As future work, we intend to make use of the attention module for retrieving the image features as well.

7. Acknowledgments

We would like to thank Google, Motorola and the Brazilian research agencies CAPES, CNPq, and FAPERGS for funding this research. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research.

References

[1] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio. Gated feedback recurrent neural networks. In *ICML*, ICML'15, pages 2067–2075, 2015.

[2] A. Eisenschtat and L. Wolf. Linking image and text with 2-way nets. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[3] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. The MIT Press, 2016.

[4] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick. Mask R-CNN. *CoRR*, abs/1703.06870, 2017.

[5] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997.

[6] Y. Huang, W. Wang, and L. Wang. Instance-aware image and sentence matching with selective multimodal lstm. In *CVPR*, July 2017.

[7] A. Joulin, E. Grave, . P. Bojanowski, and T. Mikolov. Bag of tricks for efficient text classification. *CoRR*, abs/1607.01759, 2016.

[8] A. Karpathy and L. Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR 2015*, pages 3128–3137, 2015.

[9] Y. Kim. Convolutional neural networks for sentence classification. In *In EMNLP*. Citeseer, 2014.

[10] R. Kiros, R. Salakhutdinov, and R. S. Zemel. Unifying visual-semantic embeddings with multimodal neural language models. *CoRR*, abs/1411.2539, 2014.

[11] B. Klein, G. Lev, G. Sadeh, and L. Wolf. Associating neural word embeddings with deep image representations using fisher vectors. In *CVPR*, pages 4437–4446, 2015.

[12] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár. Focal loss for dense object detection. *CoRR*, abs/1708.02002, 2017.

[13] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. In *European Conference on Computer Vision*, 2014.

[14] Z. Lin, M. Feng, C. N. d. Santos, M. Yu, B. Xiang, B. Zhou, and Y. Bengio. A structured self-attentive sentence embedding. *arXiv preprint arXiv:1703.03130*, 2017.

[15] L. Ma, Z. Lu, L. Shang, and H. Li. Multimodal convolutional neural networks for matching image and sentence. In *International Conference on Computer Vision (ICCV)*, 2015.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013.

[17] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[18] G. S. Simões, J. Wehrmann, R. C. Barros, and D. D. Ruiz. Movie genre classification with convolutional neural networks. In *IJCNN*, pages 259–266, 2016.

[19] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[20] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. 2017.

[21] I. Vendrov, R. Kiros, S. Fidler, and R. Urtasun. Order-embeddings of images and language. In *ICLR*, 2016.

[22] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan. Show and tell: A neural image caption generator. In *CVPR*, pages 3156–3164, 2015.

[23] L. Wang, Y. Li, and S. Lazebnik. Learning deep structure-preserving image-text embeddings. In *CVPR*, pages 5005–5013, 2016.

[24] J. Wehrmann, R. C. Barros, and R. Cerri. Hierarchical multi-label classification with chained neural networks. In *SAC*. ACM, 2017.

[25] J. Wehrmann, W. Becker, H. E. Cagnini, and R. C. Barros. A character-based convolutional neural network for language-agnostic twitter sentiment analysis. In *IJCNN*, 2017.

[26] J. Wehrmann, A. M. Silva, and R. C. Barros. Order embeddings and character-level convolutions for multimodal alignment. *Pattern Recognition Letters*, 102, 2018.

[27] J. Wehrmann, G. S. Simões, R. C. Barros, and V. F. Cavalcante. Adult content detection in videos with convolutional and recurrent neural networks. *Neurocomputing*, 272, 2018.

[28] W. Xiong, L. Wu, F. Alleva, J. Droppo, X. Huang, and A. Stolcke. The microsoft 2017 conversational speech recognition system. *CoRR*, abs/1708.06073, 2017.

[29] W. Zeng, W. Luo, S. Fidler, and R. Urtasun. Efficient summarization with read-again and copy mechanism. *CoRR*, abs/1611.03382, 2016.