With a Little Help from your own Past: Prototypical Memory Networks for Image Captioning

Manuele Barraco[1] Sara Sarto[1] Marcella Cornia[1] Lorenzo Baraldi[1] Rita Cucchiara[1,2]
[1]University of Modena and Reggio Emilia, Modena, Italy [2]IIT-CNR, Pisa, Italy
{name.surname}@unimore.it

Abstract

Image captioning, like many tasks involving vision and language, currently relies on Transformer-based architectures for extracting the semantics in an image and translating it into linguistically coherent descriptions. Although successful, the attention operator only considers a weighted summation of projections of the current input sample, therefore ignoring the relevant semantic information which can come from the joint observation of other samples. In this paper, we devise a network which can perform attention over activations obtained while processing other training samples, through a prototypical memory model. Our memory models the distribution of past keys and values through the definition of prototype vectors which are both discriminative and compact. Experimentally, we assess the performance of the proposed model on the COCO dataset, in comparison with carefully designed baselines and state-of-the-art approaches, and by investigating the role of each of the proposed components. We demonstrate that our proposal can increase the performance of an encoder-decoder Transformer by 3.7 CIDEr points both when training in cross-entropy only and when fine-tuning with self-critical sequence training. Source code and trained models are available at: https://github.com/aimagelab/PMA-Net.

1. Introduction

Connecting vision and natural language via descriptive expressions is a fundamental human capability and its replication represents a crucial step towards machine intelligence, with applications that range from better human-machine interfaces [25] to accessibility [15]. The task of image captioning [21, 55, 60], which defines such capability, requires an algorithm to describe a visual input with a natural language sentence. As such, it features unique challenges that span from a grounded and detailed understanding of the visual input [8, 46], to the selection of visual objects and semantics that are worth mentioning [27] and their translation into a fluent and coherent sentence.

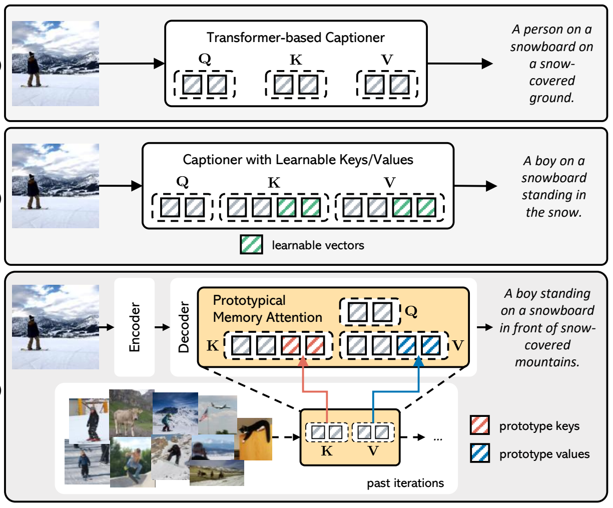
a)
Transformer-based Captioner
Q
K
V
A person on a snowboard on a snow-covered ground.
b)
Captioner with Learnable Keys/Values
Q
K
V
A boy on a snowboard standing in the snow.
learnable vectors
c)
Encoder
Decoder
Prototypical Memory Attention
Q
K
V
A boy standing on a snowboard in front of snow-covered mountains.
K
V
past iterations
prototype keys
prototype values

Figure 1. Comparison between (a) a standard Transformer-based captioner; (b) a captioner with learnable memory vectors [10] and (c) our prototypical memory network.

Image captioning architectures comprise an image encoding part and a language generation approach [49] and focus on developing appropriate connections between the visual and textual modality. Examples of such innovations include the usage of attentive-like structures [4, 36], the incorporation of attributes [26, 66], objects [4, 64] or scene graphs [62, 65]. Regardless of the specific approach used to connect the two modalities, though, almost all of the works developed in the last years share the usage of the Transformer architecture [53]. Such architecture, indeed, is a natural choice for the task, as it can connect two modalities thanks to its encoder-decoder design and the cross-attention operator, and provides unprecedented performance in sequence and set modeling and generation [10, 11, 40].

One of the key properties of attention layers is that the output is computed as a weighted combination of linear projections of the inputs. While this provides an ideal context for both visual understanding and sequence generation, it also leaves the door open for injecting relevant information which can not be directly inferred from the current input sample. An interesting attempt in this direction has been made by the Meshed-Memory architecture [10], which proposed to insert additional learnable key/value vectors in the visual encoder with the objective of integrating a-priori knowledge. While successful, learnable vectors can just store information that is useful for the entire training set and do not really act as a "memory" of past training items. Instead, having access to past training samples at generation time can be a powerful source of information that can ultimately increase the description quality. For instance, given an input image representing a boy snowboarding in a mountain landscape, a model which has access to other training items might retrieve similar images containing boy, snowboard, and mountains even in a different context, and employ this knowledge in a compositional manner to aid the generation of a correct and fluent sentence (Fig. 1).

Following this insight, we devise a *prototypical memory* network, which can recall and exploit past activations generated during training. Our memory is built upon network activations obtained during recent training iterations so that the network has access to a vast set of activations produced while processing other samples from the dataset. In this sense, the memory represents *past knowledge processed by the network itself*. From the point of view of the architectural design, our memory is fully integrated into attention layers through the addition of keys and values which represents activations from the memory. To the best of our knowledge, this is the first attempt of integrating a memory of past training items into an image captioning network.

The key element of our proposal is the computation of "prototypes" from a bank of past activations, which are obtained by modeling the manifold of past activations and clustering its content in a memory with a given fixed size. This is done both at key and value level, by exploiting the mapping between corresponding keys and values. We further justify our prototype generation approach by investigating the resulting attention distribution from a theoretical point of view. Experimentally, we assess the performances of the proposed design on the COCO dataset for image captioning, in comparison with state-of-the-art approaches and carefully-designed ablations to study the role of each component of the proposal. Further, we conduct experiments on the nocaps dataset [1] for novel object captioning and on the robust COCO split [32] for object hallucination.

We believe that the proposed approach can shed light on the effectiveness of employing the space of training items as an additional input to the network, which is currently underexplored and could be in principle applied outside of image captioning. To sum up, the main contribution of this work is the proposal of a prototype memory network for image captioning, which elegantly integrates past activations as an additional input of attention layers. Extensive experiments on COCO, nocaps, and the robust COCO split demonstrate the effectiveness of the proposed approach.

2. Related Work

Image Captioning. Early captioning approaches were constructed by filling pre-defined templates after detecting relevant objects inside of the image [48, 63]. The advent of deep learning then made the use of RNNs and LSTMs a popular choice for the task, and in analogy to the sequence modeling used in machine translation [51], the basic RNN-based encoder-decoder scheme was employed in conjunction with CNNs for encoding the visual content [24,42,55]. This approach was later augmented with the addition of attention [31, 60, 66]. Nowadays, attentive and Transformer-based architectures [12, 39, 52, 53] are often employed both in the visual encoding stage, usually applied to refine features from a CNN [69] or ViT [12], and as language models [10, 16]. Other solutions [33, 35], instead, exploit self-attention to effectively combine visual features coming from multiple backbones (*i.e.* typically a CNN and an object detector), in some cases also finetuning the visual backbones to improve final performances [35].

The introduction of Transformer-based models in image captioning has also brought to the development of effective variants of the self-attention operator [10, 14, 16, 19, 30, 36] and to that of vision-and-language early-fusion architectures [18, 26, 70] based on BERT-like models [11]. Recently, a common strategy is that of employing visual features extracted from large-scale cross-modal architectures [6, 9, 34, 47] like CLIP [39]. As done in [27] and other contemporary works [9, 41, 45], these multimodal architectures can also enable the enrichment of predicted textual sentences, by means of retrieval components that can be added to the captioning model.

Memory-Augmented Transformers. External memories have been used in different ways in Transformer-based architectures, mainly in NLP. Khandelwal *et al.* [22] constructed a memory for language generation as a large table of (key, token) pairs, while Sukhbaatar *et al.* [50] replace feed-forward layers with differentiable memory slots. Recently, Wu *et al.* propose a Memorizing Transformer [59] architecture in which they retrieve activations produced over long documents. In vision-and-language, learnable external memories have been successfully employed for image captioning [10], visual relationship recognition [7], and story generation [61].

Summary. Our proposal is different from all the aforementioned approaches, as it employs network activations coming from other training samples as a source of additional knowledge. This research direction might also be seen as an alternative to retrieving information from external knowledge bases, as it represents additional knowledge through

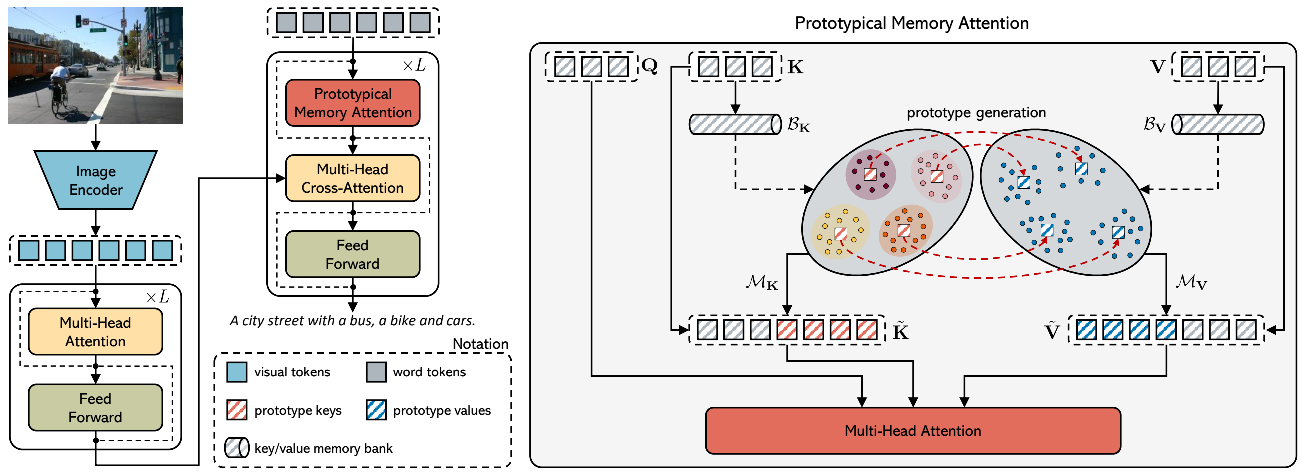

Figure 2. Overview of our approach with Prototypical Memory Attention.

network activations rather than with raw data, and further devises a compression step so that the additional information can be stored directly into the network and avoid the usage of external databases.

3. Proposed Method

3.1. Preliminaries: Memory-augmented Attention

Attention layers operate on triplets of queries, keys and values $(\mathbf{Q}, \mathbf{K}, \mathbf{V})$, which are obtained by linearly projecting items from the same input sequence (self-attention) or from a pair of different input sequences (cross-attention). We are interested in breaking the constraint of operating exclusively on input-dependent data and letting the attention operator consider quantities which are not derived from the current input [10,59]. In memory-augmented attention [10], this is achieved by extending the set of keys and values to include additional memory vectors. As a result, the attention operation can employ both input-dependent and memory-specific keys and values, as follows:

$$\tilde{\mathbf{K}} = [\mathcal{M}_\mathbf{K}; \mathbf{K}(x)], \quad \tilde{\mathbf{V}} = [\mathcal{M}_\mathbf{V}; \mathbf{V}(x)]$$
$$\text{Attention}(\mathbf{Q}, \tilde{\mathbf{K}}, \tilde{\mathbf{V}}) = \frac{\mathbf{Q}\tilde{\mathbf{K}}^\intercal}{\sqrt{d}}\tilde{\mathbf{V}} \tag{1}$$

where, for convenience, the exclusive dependency between regular keys and values and the current input sample x has been made explicit, $[\cdot;\cdot]$ indicates concatenation, $\mathcal{M}_\mathbf{K}$ the set of memory keys and $\mathcal{M}_\mathbf{V}$ the set of memory values.

Previous works which employed memory augmentation [7, 10, 38, 50, 61] have treated $\mathcal{M}_\mathbf{K}$ and $\mathcal{M}_\mathbf{V}$ as learnable parameters and, thus, optimized them directly through SGD during the learning process. Importantly, this imposes a constraint on what can be stored in memory vectors, as memories will be the result of accumulating gradient averaged over sequential mini-batches. This encourages the storage of information which is averagely beneficial to the entire training set and prevents focusing on the peculiarities of the single training items. As a consequence, learning a proper set of disentangled memory vectors turns out to be non-trivial and initialization-dependent [50].

3.2. Memories as banks of past activations

We redefine memory keys and values as a means to let the network attend previous activations produced while processing other training samples. The network, at test time, will be able to attend to its own (past) activations produced while processing similar samples, thus aiding the generation process. Conceptually speaking, we might see this as a more principled design of a *memory*, as in our case memory vectors will be actually storing past experiences of the network instead of being plain learnable parameters.

In our architecture, which we name PMA-Net, we apply memories in a core position of the encoder-decoder structure, *i.e.* inside each self-attention layer of the captioner. This is different from what has been done in previous works (*e.g.* [10] considered only the Transformer encoder), and also represents a privileged placement for vision-and-language architectures, as the self-attention layer is in charge of modeling the temporal consistency of the generation and of integrating it with the result of the previous cross-attention layer, which connects with the visual modality. Figure 2 (left) presents an overview of this design.

Considering a stream of mini-batches $[\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_t, ...]$ containing randomly sampled training items, for each layer we define two memory banks $\mathcal{B}_\mathbf{K}, \mathcal{B}_\mathbf{V}$ which store all keys and values produced from past training samples, up to a maximum temporal distance of T iterations. Intuitively, the two memory banks model the manifold of keys and values seen over past training iterations. We then define the set of memory keys and values to be employed at the t-th training iteration as a function of the vectors contained in the respective memory banks:

$$\begin{aligned}\mathcal{B}_{\mathbf{K}} &= \left[\mathbf{K}(\mathbf{x}_{t-1}), \mathbf{K}(\mathbf{x}_{t-2}), ..., \mathbf{K}(\mathbf{x}_{t-\mathrm{T}})\right], \\ \mathcal{B}_{\mathbf{V}} &= \left[\mathbf{V}(\mathbf{x}_{t-1}), \mathbf{V}(\mathbf{x}_{t-2}), ..., \mathbf{V}(\mathbf{x}_{t-\mathrm{T}})\right], \\ \mathcal{M}_{\mathbf{K}} &= f\left(\mathcal{B}_{\mathbf{K}}\right), \quad \mathcal{M}_{\mathbf{V}} = f\left(\mathcal{B}_{\mathbf{V}}\right).\end{aligned} \tag{2}$$

In the equations above, for ease of notation, we denote with $\mathbf{K}(\mathbf{x})$ the set of keys produced by a layer while processing all items contained in a mini-batch $\mathbf{x}$. In practice, the temporal window T should be chosen to be sufficiently large to reasonably model the training set distribution (as shown in Sec. 4.3). Also, memory banks need to be updated frequently and in a sufficiently smooth manner, so to follow the evaluation of the keys and values manifold and not to alter the training process, as will be discussed in the following.

Building memory prototypes. Naively placing all keys and values produced during a given time window in the memory (*i.e.* setting $f(\cdot)$ to the identity in Eq. 2) would require, approximately, $\mathrm{T} \cdot \mathrm{B} \cdot \mathrm{h} \cdot \tau$ memory slots per layer, where T represents the number of iterations executed inside the time window, B the mini-batch size, h the number of heads, and τ the average ground-truth sequence length. Under this setting, storing an entire COCO epoch would require storing around 96M memory vectors to both key and value sequences[1], which would make the problem intractable in terms of memory occupation and computational complexity, because of the additional memory required to store vectors and because of the resulting growth of the attention matrix size. Further, as keys and values have been trained to summarize the information contained in a token with respect to other tokens of the same sequence, multiple elements in the memory bank could produce similar attention scores, thus increasing the entropy of the resulting attention distributions.

For this reason, we instead build synthetic key/value pairs as *prototypical memory vectors* which are representative of the distribution of the entire memory bank. In doing this, we satisfy two design requirements: (1) building prototypes should be fast, as we will be performing this on every layer of the architecture and several times during training; (2) memory prototypes should evolve during training to adapt to the changing distribution of keys and values.

In our method, prototype key memory vectors are obtained by clustering the manifold identified by the memory bank of keys and taking the resulting centroids. Value memory vectors are, instead, computed by interpolating between the values corresponding to the keys that lie in each cluster. Formally, being m the target size of the memory, key memory vectors are obtained as follows:

$$\mathcal{M}_{\mathbf{K}} = \left[\mathcal{M}_{\mathbf{K}}^1, \mathcal{M}_{\mathbf{K}}^2, ..., \mathcal{M}_{\mathbf{K}}^m\right] = \texttt{K-Means}_m(\mathcal{B}_{\mathbf{K}}), \tag{3}$$

[1]Considering a network with 8 heads, and BPE tokenization.

where function $\texttt{K-Means}_m(\cdot)$ returns the m centroids obtained by performing a K-Means clustering over the key memory bank. Value memory vectors are computed by taking a linear combination of vectors in the value manifold that correspond to keys that lie close to key prototypes $\mathcal{M}_{\mathbf{K}}^i$, according to a distance function $d(\cdot)$ which compares items in the key manifold:

$$\begin{aligned}\mathcal{M}_{\mathbf{V}} &= \left[\mathcal{M}_{\mathbf{V}}^1, \mathcal{M}_{\mathbf{V}}^2, ..., \mathcal{M}_{\mathbf{V}}^m\right], \\ \mathcal{M}_{\mathbf{V}}^i &= \sum_{(\mathbf{K}^j, \mathbf{V}^j) \in \texttt{top-k}(\mathcal{M}_{\mathbf{K}}^i)} e^{-d\left(\mathcal{M}_{\mathbf{K}}^i, \mathbf{K}^j\right)} \mathbf{V}^j,\end{aligned} \tag{4}$$

where, here, function $\texttt{top-k}(\mathcal{M}_{\mathbf{K}}^i)$ returns the closest (key, value) pairs in the memory bank with respect to $\mathcal{M}_{\mathbf{K}}^i$.

It shall be noted that the $\texttt{top-k}$ operation can be implemented by fitting a k-NN index on the keys memory bank ($\mathcal{B}_{\mathbf{K}}$) and using the centroid $\mathcal{M}_{\mathbf{K}}^i$ as query. The resulting list of keys close to $\mathcal{M}_{\mathbf{K}}^i$ can then be paired with their corresponding values to compute Eq. 4. In practice, we use the L_2 distance as distance function inside both the key and value manifolds, as we found it to perform favorably compared to the inner product during preliminary experiments.

Discussion. With the strategy defined above, we obtain a set of m memory keys and values, where m can be controlled a-priori. Taking prototypes as centroids ensures, when m is sufficiently high, that memory keys model the memory bank distribution properly. Further, the distance between centroids and cluster members in the key manifold is small, which has a positive effect on the resulting attention distribution, compared to the one obtainable by setting $f(\cdot)$ to the identity – *i.e.*, when storing all keys and values from the memory bank in the self-attention layer. Similar keys in an L_2 space, indeed, result in similar attention distributions, as we show in the following.

Proposition – Given a query q and a set of keys $\mathbf{K}$, if a key $k \in \mathbf{K}$ is replaced with $\tilde{k}$ such that $\|k - \tilde{k}\|_2 \leq \varepsilon$ to form $\tilde{\mathbf{K}}$, then $\|\text{softmax}(q\mathbf{K}^\intercal) - \text{softmax}(q\tilde{\mathbf{K}}^\intercal)\|_2 \leq \varepsilon \|q\|_2$.

Proof. As the softmax operator has Lipschitz constant less than 1 [13, 56] and because the L_2 matrix norm is subordinate, $\|\text{softmax}(q\mathbf{K}^\intercal) - \text{softmax}(q\tilde{\mathbf{K}}^\intercal)\|_2 \leq \|q\mathbf{K}^\intercal - q\tilde{\mathbf{K}}^\intercal\|_2 \leq \|q\|_2 \|(\mathbf{K} - \tilde{\mathbf{K}})^\intercal\|_2$. Recalling that, for any matrix A, $\|A\|_2 \leq \|A\|_F$, $\|(\mathbf{K} - \tilde{\mathbf{K}})^\intercal\|_2 \leq \|k - \tilde{k}\|_2$, from which the thesis follows. $\square$

Memory bank update. To ensure that memories are refreshed during training while lowering the total cost of prototypes generation, we adopt a strided sliding window approach to update the memory banks, which is visually depicted in Fig. 3. Having defined a maximum length T for the banks (Eq. 2), at regular intervals we take the last T batches from the key/value stream produced by a layer, create a memory bank with those and generate prototype vectors to be placed in $\mathcal{M}_{\mathbf{K}}$ and $\mathcal{M}_{\mathbf{V}}$ (Eq. 3, 4). In practice, the process is repeated twice per epoch and memory banks

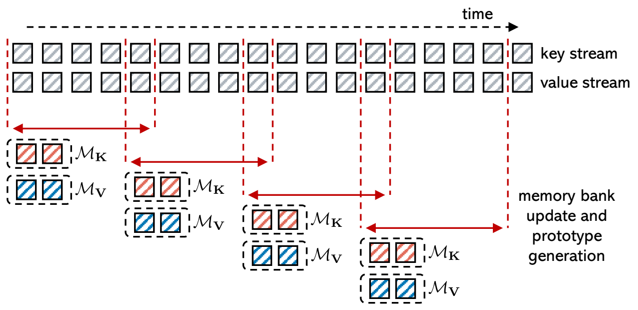

Figure 3. Memory bank update approach.

store around two epochs of samples, so to have a significant overlap between the memory banks obtained at two consecutive update steps, which helps to stabilize the training. This is also illustrated in pseudo-code in Algorithm 1.

Segment embeddings. As the final set of keys of the layer is a concatenation of memory-specific and input-specific keys (Eq. 1), we add two different, learnable, segment embeddings to $\mathbf{K}$ and $\mathcal{M}_\mathbf{K}$, to help the network distinguish between the two key types.

Computational complexity. Computing memory prototypes requires executing a K-Means clustering over the key memory bank ($\mathrm{T} \cdot \mathrm{B} \cdot \mathrm{h} \cdot \tau$ datapoints, m clusters) and a kNN search over the key memory bank (which contains the same number of items) and is executed every s training steps, being s the stride employed over the key/value stream to update the memory banks (Fig. 3). Further, the addition of the m memory vectors to a mini-batch having a sequence length of T implies growing the attention matrix from $T \times T$ to $(T + m) \times T$. As prototype generation is only required at training time, the latter is the only cost that is added at test time with respect to a standard attention layer.

In practice, adding prototypical memories does not increase inference times significantly with respect to a naive Transformer as the increase of the attention matrix is well amortized by the GPU parallelism. During training, computing the K-Means clustering and the k-NN index for solving Eq. 4 requires around 10s with a V100 GPU every time the memory needs to be refreshed. As the memory occupied for this can be de-allocated after prototypes computation, we did not need to decrease the batch size with respect to a baseline with learnable memory vectors.

4. Experimental Evaluation

4.1. Datasets and evaluation protocol

We analyze the effectiveness of our PMA-Net on the widely used COCO benchmark [29] employing the splits defined in [21]. We also evaluate on the COCO online test server composed of more than 40k images for which ground-truth captions are not publicly available.

Additionally, we perform experiments on robust COCO,

Algorithm 1 PMA-Net pseudocode

```
# m: number of prototypes
# T: maximum length of the memory bank
# stride: memory bank update stride
# bank_k, bank_v: key/value memory banks
bank_k = [], bank_v = []
for img, caption in dataloader:
  output, act_k, act_v = net(img, caption)
  bank_k.append(act_k)
  bank_v.append(act_v)
  if len(bank_k) == T:
    compute_prototypes(m, bank_k, bank_v) # Eq. 3, 4
  bank_k = bank_k[stride:]
  bank_v = bank_v[stride:]
  loss = loss_fn(output, caption)
  loss.backward()
```

a different split of the COCO dataset introduced in [32] to verify sensitivity to object hallucination and nocaps [1] for novel object captioning. The former dataset guarantees that object pairs mentioned in captions of different sets do not overlap (with 110,234, 3,915, and 9,138 images for training, validation, and test), while the latter contains images annotated with 10 human-written captions, that can be further divided in in-domain, near-domain and out-of-domain pairs depending on their nearness to COCO.

To evaluate our results, we employ all standard captioning metrics, namely BLEU [37], METEOR [5], ROUGE [28], CIDEr [54], and SPICE [2], and some more recent evaluation scores like BERT-S [68], CLIP-S [17], and PAC-S [44] in both their reference-free and reference-based versions. When evaluating our results on robust COCO, we also employ the CHAIR metric [43] that measures which fraction of objects mentioned in the generated sentences is hallucinated (CHi) and the portion of sentences that includes a hallucinated object (CHs).

4.2. Implementation details

Both the encoder and decoder are constructed with $L = 6$ Transformer layers, with a hidden size of 512 and 8 attention heads. Unless otherwise specified, we employ 1,024 memory vectors and a size of the memory banks T equal to 1,500. We use a CLIP [39] ViT-L/14 image encoder over the input image. The rationale behind this choice is that they have higher quality, adaptability to different tasks, and lower computational load compared to detection-based ones. To ensure fair comparison, we re-train recent and publicly-available models using the same features.

Our source code is based on Huggingface [57], using the GPU-based implementations of K-Means and k-NN search from FAISS [20]. At training stage, the overall objective of PMA-Net is the typical cross-entropy loss for sentence generation. Next, following [42], PMA-Net can be further optimized with sentence-level reward, using the CIDEr score. Specifically, we first pre-train with the LAMB optimizer [67], a batch size of 1,024 and for 20,000 steps. We use the following learning rate schedule: we linearly

	m	T	B-4	M	R	C	S
Transformer [53]	-	-	37.4	30.3	58.9	127.8	23.3
Transformer (w/ learnable mem.)	64	-	37.7	30.2	58.1	127.9	23.4
Transformer (w/ learnable mem.)	1024	-	37.2	30.1	58.3	127.6	23.3
PMA-Net	256	1500	38.8	30.1	59.4	129.4	23.5
PMA-Net	512	1500	39.0	30.1	59.5	130.0	23.5
PMA-Net	1024	500	37.8	30.3	59.0	128.6	23.5
PMA-Net	1024	1000	38.2	**30.5**	59.5	129.4	23.5
PMA-Net (w/o mem. in 1st layer)	1024	1500	38.3	30.2	59.0	129.2	23.3
PMA-Net (w/o segment emb.)	1024	1500	38.6	30.4	59.4	130.1	23.4
PMA-Net	1024	1500	**39.5**	30.4	**59.6**	**131.5**	**23.6**

Table 1. Ablation study (m is the number of memory vectors and T is the size of the memory banks).

warmup for 1,000 steps, then keep a constant learning rate of $2.5 \cdot 10^{-4}$ until 10,000 steps, then sub-linearly decrease until 15,000 steps to 10^{-5} and keep the value constant for the rest of the training. For the second stage, we further optimize PMA-Net using the Adam optimizer [23] and with $1 \cdot 10^{-6}$ as learning rate, for 50,000 steps using a batch size of 64. We employ a beam size equal to 5.

4.3. Comparisons and ablation studies

We firstly conduct an ablation study to investigate how each design choice in our PMA-Net influences the overall performances on COCO dataset. Table 1 details the performance comparisons among different ablated runs. Note that all the results reported here are without self-critical training strategy. We start from a base Transformer encoder-decoder architecture, which is also a degraded version of PMA-Net without memory banks and prototype vectors. Subsequently, we compare by adding learnable memory vectors as defined in [10, 50] but in the same position of PMA-Net, *i.e.* in place of the self-attention layer in the sentence decoder instead of the visual encoder. Then, we add memory banks and prototype vectors and vary the number of clusters and the size of the memory banks.

Firstly, we notice that the basic learnable memory vectors do not give a significant contribution when placed in the sentence decoder, outlining that in this core position of the captioner, in which activations coming from both modalities are merged, learning appropriate memory vectors becomes more complex. Instead, the proposed prototype vectors increase caption quality significantly, up to 3.7 CIDEr points, highlighting the appropriateness of the proposed strategy. We notice that increasing the number of clusters and the size of the memory banks exhibits better performances, as we hypothesize that this provides a better estimation of the key and value manifold and more fine-grained prototypes.

In our architecture, the sliding window contains the last T$\cdot$B captions seen, which in our best configuration amounts to 1.5M samples. Being COCO 0.6M image-text pairs, this models the training set distribution and its evolution across more than two epochs. We notice that increasing T further

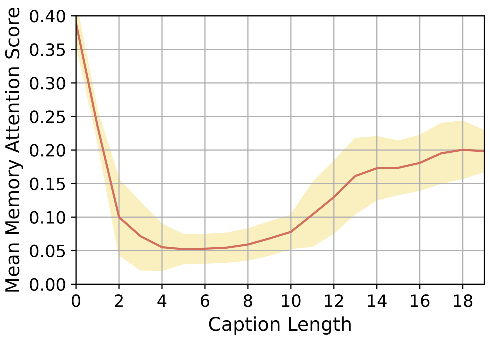

Figure 4. Average magnitude of attention on prototype memories over time on the COCO Karpathy validation split.

does not enhance performance; reducing it, especially to less than one epoch, is instead detrimental.

In the lower part of Table 1, we run additional ablations on two design choices: the use of segment embeddings to distinguish prototypes from input-dependent keys, and the incorporation of prototypical vectors in the first layer of the captioner. The second experiment arises from the fact that the first layer is not influenced by cross-attention results and, therefore, by multimodal connections with the input image. It can be observed that the segment embeddings provide a relevant contribution, and that memory prototypes have an impact both on the first layer and on the other layers, outlining that its advantage is inherently multimodal.

Memory attention visualization. We show how the prototype memories are employed during the generation of the captions. To this aim, we compute a memory attention score that represents the percentage of the attention that is applied on the memories when the model has to decide which token to generate. For each generated token and a layer l, we look at the attention scores with respect to past keys ($\boldsymbol{a}_p^l$) and the memory ($\boldsymbol{a}_m^l$). We then compute the memory attention score for the layer as mean$(\boldsymbol{a}_m^l)/(\text{mean}(\boldsymbol{a}_m^l)+\text{mean}(\boldsymbol{a}_p^l))$, then average across layers. Figure 4 shows the average attention scores over the COCO test set. As can be seen, the memory is employed during the generation of the entire sentence. Specifically, we observe a strong peak in the initial part of the generation, in which we speculate that the network retrieves a-priori information from the memory. Attention scores further increase in the final part, while the network describes details and less relevant objects which can benefit from the retrieval of additional knowledge.

4.4. Comparison with the State-of-the-Art

We then compare PMA-Net with different state-of-the-art approaches. The models we compare to include the classic Up-Down [4] approach, GCN-LSTM [65] that uses graph convolutional networks to encode visual relationships, SGAE [62] which is based on scene graphs, AoANet [19], X-Transformer [36] and RSTNet [69] that propose attention variants, DLCT [33], DIFNet [58] that

	Cross-Entropy Loss								CIDEr Optimization							
	B-1	B-2	B-3	B-4	M	R	C	S	B-1	B-2	B-3	B-4	M	R	C	S
Up-Down [4]	77.2	-	-	36.2	27.0	56.4	113.5	20.3	79.8	-	-	36.3	27.7	56.9	120.1	21.4
GCN-LSTM [65]	77.3	-	-	36.8	27.9	57.0	116.3	20.9	80.9	-	-	38.3	28.6	58.5	128.7	22.1
SGAE [62]	77.6	-	-	36.9	27.7	57.2	116.7	20.9	81.0	-	-	39.0	28.4	58.9	129.1	22.2
AoANet [19]	77.4	-	-	37.2	28.4	57.5	119.8	21.3	80.2	-	-	38.9	29.2	58.8	129.8	22.4
$\mathcal{M}^2$ Transformer [10]	-	-	-	-	-	-	-	-	80.8	-	-	39.1	29.2	58.6	131.2	22.6
X-Transformer [36]	77.3	61.5	47.8	37.0	28.7	57.5	120.0	21.8	80.9	65.8	51.5	39.7	29.5	59.1	132.8	23.4
DLCT [33]	-	-	-	-	-	-	-	-	81.4	-	-	39.8	29.5	59.1	133.8	23.0
RSTNet [69]	-	-	-	-	-	-	-	-	81.8	-	-	40.1	29.8	59.5	135.6	23.3
DIFNet [58]	-	-	-	-	-	-	-	-	81.7	-	-	40.0	29.7	59.4	136.2	23.2
CaMEL [6]	78.3	-	-	39.1	29.4	58.5	125.7	22.2	82.8	-	-	41.3	30.2	60.1	140.6	23.9
COS-Net [27]	**79.2**	63.8	50.2	39.2	29.7	58.9	127.4	22.7	82.7	68.2	54.0	42.0	**30.6**	60.6	141.1	**24.6**
GRIT* [35]	-	-	-	-	-	-	-	-	84.2	-	-	42.4	30.6	60.7	144.2	24.3
Transformer†	76.4	61.0	47.9	37.4	30.3	58.9	127.8	23.3	83.4	68.6	54.2	42.0	30.0	60.6	140.3	23.5
$\mathcal{M}^2$ Transformer† [10]	78.8	63.3	49.5	38.7	29.6	58.9	127.8	23.3	83.7	69.2	54.8	42.3	30.5	61.0	141.2	23.6
CaMEL† [6]	78.8	63.5	50.3	39.2	30.0	59.3	129.9	23.4	83.6	69.0	54.7	42.4	**30.6**	60.9	142.4	23.6
PMA-Net	79.0	**64.2**	**50.7**	**39.5**	**30.4**	**59.6**	**131.5**	**23.6**	**83.8**	**69.3**	**55.0**	**43.0**	**30.6**	**61.1**	**144.1**	24.0

Table 2. Comparison with the state of the art on the Karpathy test. The † marker indicates models re-trained with the same visual features used by our approach, while $*$ indicates finetuning of the visual backbone.

	Training	BERT-S	CLIP-S	RefCLIP-S	PAC-S	RefPAC-S
Transformer†	XE	0.947	0.741	0.804	0.819	0.865
$\mathcal{M}^2$ Transformer† [10]	XE	0.946	0.744	0.806	0.815	0.864
CaMEL† [6]	XE	0.947	0.745	0.807	0.818	0.865
PMA-Net	XE	**0.948**	**0.754**	**0.812**	**0.821**	**0.868**
Transformer†	SCST	**0.947**	0.749	0.807	0.818	0.864
$\mathcal{M}^2$ Transformer† [10]	SCST	0.946	0.749	0.809	0.817	0.865
CaMEL† [6]	SCST	0.945	0.751	0.810	0.818	0.865
PMA-Net	SCST	0.946	**0.755**	**0.814**	**0.821**	**0.869**

Table 3. Comparison with additional metrics. † indicates models re-trained with the same visual features used by our approach.

GT: A group of horse mounted police standing in front of a crowd.
Transformer: A group of police officers standing in a street.
PMA-Net: A group of police officers on horses in a street.

GT: A man takes a picture of snowy mountains with his cell phone.
Transformer: A man taking a picture of the mountains.
PMA-Net: A man taking a picture of mountains with a cell phone.

GT: A Subway sandwich with chips raisins and a coffee cup.
Transformer: A sandwich and a bag of chips on a table.
PMA-Net: A table with a sandwich and chips and a cup of coffee.

Figure 5. Qualitative results on COCO sample images.

combine features extracted from multiple backbones, COS-Net [27] which retrieves knowledge from an external base, and GRIT [35] that uses a DETR-based detector and a grid feature network for image encoding, which are both finetuned during training. For fairness, we also compare with methods re-trained using our visual features, namely CaMEL [6] which employs a mean teacher learning approach, and the $\mathcal{M}^2$ Transformer [10].

Karpathy test split. In Table 2 we report results on the standard Karpathy test split, in a single-model setting. The upper part of the table shows the results reported by the compared approaches, using their original features. In the lower part, instead, we re-train different approaches on the same CLIP grid features we employ for training PMA-Net. Specifically, in addition to a Transformer, we re-train the $\mathcal{M}^2$ Transformer [10] and CaMEL [6], which both represent recent and complementary approaches which could also be integrated with our proposal. With respect to a standard Transformer, PMA-Net exhibits a margin of 3.8 CIDEr points also under CIDEr optimization, similarly to the XE training setting. Further, when compared with recent approaches using the same features, PMA-Net also provides better performance. As shown in the table, with the exception of GRIT [35] that differently from us finetunes the visual backbone, PMA-Net consistently outperforms all the state-of-the-art approaches according to all metrics. Notably, our model is still competitive even compared to GRIT, despite the latter uses more powerful visual features. In Table 3, we also compare against baselines trained on the same features using more recent learnable metrics, *i.e.* BERT-S, CLIP-S and, PAC-S. To qualitatively validate the effectiveness of our solution, we report sample images and corresponding predicted captions in Fig. 5.

COCO Test Server. We also report the performances of our approach obtained on the official COCO test split, through the online test server[2]. Table 4 reports the performance with respect to 5 reference captions (c5) and 40 reference captions (c40). Following previous literature [10, 27], we report the results using an ensemble of four models. As it can

[2]https://codalab.lisn.upsaclay.fr/competitions/7404

	BLEU-1		BLEU-2		BLEU-3		BLEU-4		METEOR		ROUGE		CIDEr	
	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40	c5	c40
Up-Down [4]	80.2	95.2	64.1	88.8	49.1	79.4	36.9	68.5	27.6	36.7	57.1	72.4	117.9	120.5
SGAE [62]	81.0	95.3	65.6	89.5	50.7	80.4	38.5	69.7	28.2	37.2	58.6	73.6	123.8	126.5
AoANet [19]	81.0	95.0	65.8	89.6	51.4	81.3	39.4	71.2	29.1	38.5	58.9	74.5	126.9	129.6
$\mathcal{M}^2$ Transformer [10]	81.6	96.0	66.4	90.8	51.8	82.7	39.7	72.8	29.4	39.0	59.2	74.8	129.3	132.1
X-Transformer [36]	81.9	95.7	66.9	90.5	52.4	82.5	40.3	72.4	29.6	39.2	59.5	75.0	131.1	133.5
RSTNet [69]	82.1	96.4	67.0	91.3	52.2	83.0	40.0	73.1	29.6	39.1	59.5	74.6	131.9	134.0
DLCT [33]	82.4	96.6	67.4	91.7	52.8	83.8	40.6	74.0	29.8	39.6	59.8	75.3	133.3	135.4
COS-Net [27]	83.3	96.8	68.6	92.3	54.2	84.5	42.0	74.7	30.4	40.1	60.6	76.4	136.7	138.3
CaMEL [6]	83.2	97.3	68.3	92.7	53.6	84.8	41.2	74.9	30.2	39.7	60.2	75.6	137.5	140.0
PMA-Net	**84.7**	**97.9**	**70.2**	**93.8**	**55.7**	**86.5**	**43.4**	**77.1**	**30.5**	**40.3**	**61.3**	**76.8**	**141.5**	**143.4**

Table 4. Leaderboard of various methods on the online COCO test server.

	B-1	B-4	M	R	C	S	CHs	CHi
Att2In [42]	-	-	24.0	-	85.8	16.9	14.1	10.1
Up-Down [4]	-	-	24.7	-	89.8	17.7	11.3	7.9
Transformer [27]	76.9	36.3	27.4	56.1	109.3	20.5	7.9	5.1
COS-Net [27]	78.0	37.3	27.9	56.8	112.1	21.2	6.2	3.9
Transformer†	77.4	37.8	**29.4**	58.1	119.6	22.3	4.6	2.8
PMA-Net	**79.5**	**39.3**	**29.4**	**58.7**	**122.0**	**22.5**	**4.3**	**2.6**

Table 5. Results on robust COCO test set. The † marker indicates a model re-trained with the same visual features of our approach.

	In		Out		Overall	
	C	S	C	S	C	S
NBT [1]	62.1	10.1	62.4	8.9	60.2	9.5
Up-Down [1]	80.0	12.0	66.4	9.7	73.1	11.1
Transformer [10]	78.0	11.0	29.7	7.8	54.7	9.8
$\mathcal{M}^2$ Transformer [10]	85.7	12.1	38.9	8.9	64.5	11.1
GRIT* [35]	105.9	13.6	72.6	11.1	90.2	12.8
Transformer†	105.9	13.3	73.9	11.3	90.9	12.6
PMA-Net	**107.5**	**13.7**	**75.9**	**11.4**	**92.6**	**12.8**

Table 6. Results on nocaps validation set. The † marker indicates a model re-trained with the same visual features of our approach, while $*$ indicates finetuning of the visual backbone.

be seen, also in this setting PMA-Net surpasses the compared approaches by a large margin, further demonstrating its effectiveness on the COCO dataset.

Robust COCO split and sensitivity to hallucination. As our approach relies on the memorization of other training samples, we verify whether the proposed strategy has an impact in terms of object hallucination. We perform this analysis by employing the robust COCO splits defined in [32] and report the results in Table 5, comparing with state-of-the-art approaches and with a Transformer trained with the same visual features. Both PMA-Net and the Transformer baseline are re-trained from scratch on this dataset using cross-entropy loss only. In addition to standard evaluation metrics, we employ the CHAIR score, in its variants CHi and CHs, to measure object hallucination. From this analysis, it can be seen that the addition of prototypes memory vectors reduces the hallucination rate with respect to a Transformer, and that PMA-Net performs favorably with respect to previous approaches also in this case.

Novel object captioning. We also evaluate PMA-Net on the nocaps dataset [1] for novel object captioning. It shall be noted that our approach does not leverage components which are explicitly designed to deal with the naming of novel objects, still the nocaps dataset provides a relevant test bed to compare PMA-Net with other approaches from the literature. To conduct this analysis, we employ our model and the Transformer-based baseline trained on the standard COCO dataset. Results are reported in Table 6, both in the in-domain and out-of-domain splits of the datasets and without employing constrained beam search [3]. We observe that PMA-Net achieves the best performance among all the compared approaches and with respect to the base Transformer which does not employ memory prototypes. In this setting, PMA-Net also outperforms the results of GRIT, which employs a finetuned visual backbone. This highlights that the addition of prototypes memory vectors improves the description of novel objects.

5. Conclusion

We presented PMA-Net, a novel architecture for image captioning that is based on novel prototypical memory vectors which are integrated into standard attention layers to summarize activations produced during recent training iterations. Noticeably, the exploitation of past activations and the construction of memory prototypes is unprecedented in vision-and-language architectures. Experimental results demonstrate the effectiveness of PMA-Net on the COCO dataset and show that it can alleviate hallucination effects and describe novel objects better than competitors.

Acknowledgments

We thank CINECA for providing computational resources. This work has been supported by the PNRR-M4C2 project (PE00000013) “FAIR - Future Artificial Intelligence Research” funded by the European Commission and the PRIN “CREATIVE: CRoss-modal understanding and gEnerATIon of Visual and tExtual content” co-funded by the Italian Ministry of University and Research (CUP B87G22000460001).

References

[1] Harsh Agrawal, Karan Desai, Xinlei Chen, Rishabh Jain, Dhruv Batra, Devi Parikh, Stefan Lee, and Peter Anderson. nocaps: novel object captioning at scale. In *ICCV*, 2019. 2, 5, 8

[2] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. SPICE: Semantic Propositional Image Caption Evaluation. In *ECCV*, 2016. 5

[3] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Guided open vocabulary image captioning with constrained beam search. In *EMNLP*, 2017. 8

[4] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, 2018. 1, 6, 7, 8

[5] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *ACL Workshops*, 2005. 5

[6] Manuele Barraco, Matteo Stefanini, Marcella Cornia, Silvia Cascianelli, Lorenzo Baraldi, and Rita Cucchiara. CaMEL: Mean Teacher Learning for Image Captioning. In *ICPR*, 2022. 2, 7, 8

[7] Jun Chen, Aniket Agarwal, Sherif Abdelkarim, Deyao Zhu, and Mohamed Elhoseiny. Reltransformer: A transformer-based long-tail visual relationship recognition. In *CVPR*, 2022. 2, 3

[8] Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Show, Control and Tell: A Framework for Generating Controllable and Grounded Captions. In *CVPR*, 2019. 1

[9] Marcella Cornia, Lorenzo Baraldi, Giuseppe Fiameni, and Rita Cucchiara. Universal Captioner: Inducing Content-Style Separation in Vision-and-Language Model Training. *arXiv preprint arXiv:2111.12727*, 2022. 2

[10] Marcella Cornia, Matteo Stefanini, Lorenzo Baraldi, and Rita Cucchiara. Meshed-Memory Transformer for Image Captioning. In *CVPR*, 2020. 1, 2, 3, 6, 7, 8

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *NAACL*, 2018. 1, 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021. 2

[13] Bolin Gao and Lacra Pavel. On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning. *arXiv preprint arXiv:1704.00805*, 2017. 4

[14] Longteng Guo, Jing Liu, Xinxin Zhu, Peng Yao, Shichen Lu, and Hanqing Lu. Normalized and Geometry-Aware Self-Attention Network for Image Captioning. In *CVPR*, 2020. 2

[15] Danna Gurari, Yinan Zhao, Meng Zhang, and Nilavra Bhattacharya. Captioning Images Taken by People Who Are Blind. In *ECCV*, 2020. 1

[16] Simao Herdade, Armin Kappeler, Kofi Boakye, and Joao Soares. Image Captioning: Transforming Objects into Words. In *NeurIPS*, 2019. 2

[17] Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, and Yejin Choi. CLIPScore: A Reference-free Evaluation Metric for Image Captioning. In *EMNLP*, 2021. 5

[18] Xiaowei Hu, Zhe Gan, Jianfeng Wang, Zhengyuan Yang, Zicheng Liu, Yumao Lu, and Lijuan Wang. Scaling Up Vision-Language Pre-training for Image Captioning. In *CVPR*, 2022. 2

[19] Lun Huang, Wenmin Wang, Jie Chen, and Xiao-Yong Wei. Attention on Attention for Image Captioning. In *ICCV*, 2019. 2, 6, 7, 8

[20] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 5

[21] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *CVPR*, 2015. 1, 5

[22] Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through Memorization: Nearest Neighbor Language Models. In *ICLR*, 2020. 2

[23] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *ICLR*, 2015. 6

[24] Federico Landi, Lorenzo Baraldi, Marcella Cornia, and Rita Cucchiara. Working Memory Connections for LSTM. *Neural Networks*, 144:334–341, 2021. 2

[25] Xinghang Li, Di Guo, Huaping Liu, and Fuchun Sun. Robotic indoor scene captioning from streaming video. In *ICRA*, 2021. 1

[26] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks. In *ECCV*, 2020. 1, 2

[27] Yehao Li, Yingwei Pan, Ting Yao, and Tao Mei. Comprehending and ordering semantics for image captioning. In *CVPR*, 2022. 1, 2, 7, 8

[28] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *ACL Workshops*, 2004. 5

[29] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *ECCV*, 2014. 5

[30] Fenglin Liu, Xuancheng Ren, Xian Wu, Shen Ge, Wei Fan, Yuexian Zou, and Xu Sun. Prophet Attention: Predicting Attention with Future Attention. In *NeurIPS*, 2020. 2

[31] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *CVPR*, 2017. 2

[32] Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Neural Baby Talk. In *CVPR*, 2018. 2, 5, 8

[33] Yunpeng Luo, Jiayi Ji, Xiaoshuai Sun, Liujuan Cao, Yongjian Wu, Feiyue Huang, Chia-Wen Lin, and Rongrong Ji. Dual-Level Collaborative Transformer for Image Captioning. In *AAAI*, 2021. 2, 6, 7, 8

[34] Ron Mokady, Amir Hertz, and Amit H Bermano. ClipCap: CLIP Prefix for Image Captioning. *arXiv preprint arXiv:2111.09734*, 2021. 2

[35] Van-Quang Nguyen, Masanori Suganuma, and Takayuki Okatani. GRIT: Faster and Better Image captioning Transformer Using Dual Visual Features. In *ECCV*, 2022. 2, 7, 8

[36] Yingwei Pan, Ting Yao, Yehao Li, and Tao Mei. X-Linear Attention Networks for Image Captioning. In *CVPR*, 2020. 1, 2, 6, 7, 8

[37] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *ACL*, 2002. 5

[38] Mengshi Qi, Jie Qin, Di Huang, Zhiqiang Shen, Yi Yang, and Jiebo Luo. Latent memory-augmented graph transformer for visual storytelling. In *ACM Multimedia*, 2021. 3

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models From Natural Language Supervision. In *ICML*, 2021. 2, 5

[40] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners, 2019. 1

[41] Rita Ramos, Bruno Martins, Desmond Elliott, and Yova Kementchedjhieva. SmallCap: Lightweight Image Captioning Prompted With Retrieval Augmentation. In *CVPR*, 2023. 2

[42] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-Critical Sequence Training for Image Captioning. In *CVPR*, 2017. 2, 5, 8

[43] Anna Rohrbach, Lisa Anne Hendricks, Kaylee Burns, Trevor Darrell, and Kate Saenko. Object Hallucination in Image Captioning. In *EMNLP*, 2018. 5

[44] Sara Sarto, Manuele Barraco, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Positive-Augmented Contrastive Learning for Image and Video Captioning Evaluation. In *CVPR*, 2023. 5

[45] Sara Sarto, Marcella Cornia, Lorenzo Baraldi, and Rita Cucchiara. Retrieval-augmented transformer for image captioning. In *CBMI*, 2022. 2

[46] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural Machine Translation of Rare Words with Subword Units. In *ACL*, 2016. 1

[47] Sheng Shen, Liunian Harold Li, Hao Tan, Mohit Bansal, Anna Rohrbach, Kai-Wei Chang, Zhewei Yao, and Kurt Keutzer. How Much Can CLIP Benefit Vision-and-Language Tasks? In *ICLR*, 2022. 2

[48] Richard Socher and Li Fei-Fei. Connecting modalities: Semi-supervised segmentation and annotation of images using unaligned text corpora. In *CVPR*, 2010. 2

[49] Matteo Stefanini, Marcella Cornia, Lorenzo Baraldi, Silvia Cascianelli, Giuseppe Fiameni, and Rita Cucchiara. From Show to Tell: A Survey on Deep Learning-based Image Captioning. *IEEE Trans. PAMI*, 45(1):539–559, 2022. 1

[50] Sainbayar Sukhbaatar, Edouard Grave, Guillaume Lample, Herve Jegou, and Armand Joulin. Augmenting self-attention with persistent memory. *arXiv preprint arXiv:1907.01470*, 2019. 2, 3, 6

[51] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NeurIPS*, 2014. 2

[52] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021. 2

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2, 6

[54] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. CIDEr: Consensus-based Image Description Evaluation. In *CVPR*, 2015. 5

[55] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 1, 2

[56] Apoorv Vyas, Angelos Katharopoulos, and François Fleuret. Fast transformers with clustered attention. In *NeurIPS*, 2020. 4

[57] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Transformers: State-of-the-Art Natural Language Processing. In *EMNLP*, 2020. 5

[58] Mingrui Wu, Xuying Zhang, Xiaoshuai Sun, Yiyi Zhou, Chao Chen, Jiaxin Gu, Xing Sun, and Rongrong Ji. DIFNet: Boosting Visual Information Flow for Image Captioning. In *CVPR*, 2022. 6, 7

[59] Yuhuai Wu, Markus N Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing Transformers. In *ICLR*, 2022. 2, 3

[60] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015. 1, 2

[61] Dizhan Xue, Shengsheng Qian, Quan Fang, and Changsheng Xu. Mmt: Image-guided story ending generation with multimodal memory transformer. In *ACM Multimedia*, 2022. 2, 3

[62] Xu Yang, Kaihua Tang, Hanwang Zhang, and Jianfei Cai. Auto-Encoding Scene Graphs for Image Captioning. In *CVPR*, 2019. 1, 6, 7, 8

[63] Benjamin Z Yao, Xiong Yang, Liang Lin, Mun Wai Lee, and Song-Chun Zhu. I2t: Image parsing to text description. *Proceedings of the IEEE*, 98(8), 2010. 2

[64] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Incorporating Copying Mechanism in Image Captioning for Learning Novel Objects. In *CVPR*, 2017. 1

[65] Ting Yao, Yingwei Pan, Yehao Li, and Tao Mei. Exploring Visual Relationship for Image Captioning. In *ECCV*, 2018. 1, 6, 7

[66] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *CVPR*, 2016. 1, 2

[67] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Dem-

mel, Kurt Keutzer, and Cho-Jui Hsieh. Large Batch Optimization for Deep Learning: Training BERT in 76 minutes. In *ICLR*, 2020. 5

[68] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. BERTScore: Evaluating Text Generation with BERT. In *ICLR*, 2020. 5

[69] Xuying Zhang, Xiaoshuai Sun, Yunpeng Luo, Jiayi Ji, Yiyi Zhou, Yongjian Wu, Feiyue Huang, and Rongrong Ji. RST-Net: Captioning With Adaptive Attention on Visual and Non-Visual Words. In *CVPR*, 2021. 2, 6, 7, 8

[70] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. Unified Vision-Language Pre-Training for Image Captioning and VQA. In *AAAI*, 2020. 2