Graph-Augmented OSCAR

Lorenzo Garcia Condoluci
1808446

Alessandra Della Fazia
1970722

Domenico Mattia Cinque
1784965

Michelangelo Saveriano
1823326

Abstract Image captioning is the task of providing a short description of an image. Our project aims to improve the performance of OSCAR [1] by inserting a Graph Convolutional Network [2] in the existing architecture. The main idea is to exploit the relationships between the objects. We trained the model on a subset of the Coco Captions dataset and we found that the original model does not benefit from this approach.

1 Introduction

The task of generating captions for an image is strongly linked to the structure of the relationships between the objects in the image. OSCAR (Object-Semantics Aligned Pre-training for Vision-and-Language Tasks) is a vision-language pre-training (VLP) method that learns generic image-text representations that can be adapted to serve on various downstream tasks (such as VQA, Image-Text Retrieval, and Image Captioning). By introducing a Graph Convolutional Network, we want to learn a more meaningful representation of the image features that exploit the relationships in the image. These new features are then passed trough OSCAR in order to make predictions.

2 Related Work

OSCAR To build an image captioning model, previous works use image-text pair as input, a detection model for process the image, and a pre-trained word embedding for process the text. Those features were concatenated and fed into a multi-layer Transformer. Unlike these models, OSCAR introduces object tags detected into the images: the input is a triplet consisting of a word sequence, a set of object tags, and a set of image region features. The OSCAR model is pre-trained on massive amount of data and can be adapted to the Image Captioning task.

During inference the model encodes the image regions, object tags, and a special token [CLS] as input. [CLS] token is follow by the [MASK] token. The model starts the generation by feeding in a [MASK] token and sampling a token from the vocabulary based on probability. Then the [MASK] token is replaced with the sampled token and a new [MASK] is appended for the next word prediction. The generation process terminates when the model outputs the [STOP] token. A beam search (with beam size = 5) is used.

Graph Convolutional Networks By representing the data as graphs, the structural information can be encoded to model the relations among entities, and provide more promising insights underlying the data. Convolution in Neural Network consists in multiplying the input neurons with a set of weights that are commonly known as filters or kernels. The filters act as a sliding window across the whole image and enable CNNs to learn features from neighboring cells. Within the same layer, the same filter will be used throughout image, this is referred to as weight sharing. GCNs perform similar operations where the model learns the features by inspecting neighboring nodes. The advantage of GNNs is that we can work on irregular non-Euclidean structured data.

3 Proposed method: Graph-Augmented OSCAR

Graph-Creation First of all, the image features are extracted using a Faster R-CNN. For each image we are provided with a image feature vector $\mathbf{v}$ that contains the features and coordinates of the bounding boxes for each object that is detected in the image. We build 3 graphs that are averaged to get a single final graph to be fed to the GCN:

- *Bounding box graph*: this first graph is based on the matrix $\mathbf{D}$ of the pairwise distances $d(B_i, B_j)$ between the bounding boxes. In particular let's define a box as:

  $$B = (l, t, r, b) = (x_{\text{left}}, y_{\text{top}}, x_{\text{right}}, y_{\text{bottom}})$$

  Then the distances are defined as:

  $$\begin{aligned} d(B_i, B_j) &= \sqrt{d_x(B_i, B_j)^2 + d_y(B_i, B_j)^2} \\ \text{where} \quad d_x(B_i, B_j) &= \text{ReLU}(\max(r_i - l_j, r_j - l_i) + \tau_D) \\ \text{and} \quad d_y(B_i, B_j) &= \text{ReLU}(\max(b_i - t_j, b_j - t_i) + \tau_D) \end{aligned}$$

  The parameter τ_D is needed to add a small distance to superimposed bounding boxes.

  The adjacency matrix $\mathbf{A}^D$, for the bounding box graph, is then calculated as

  $$\mathbf{A}^D = 1 - \mathbf{D} \iff \mathbf{A}^D_{i,j} = 1 - d(B_i, B_j)$$

  A simpler graph could be based on the objects' position within the image and the distance between the centers of the bounding boxes. However we found that this metric poses some problems when considering two big objects: even in the case they overlap, the distance between their centers may be quite distant.

- *Thresholded absolute cosine distance graph*: In this case we use the *cosine similarity* $d(\mathbf{v}'_i, \mathbf{v}'_j)$ between the image features $\mathbf{v}'$ to make the adjacency matrix $\mathbf{A}^C$

  $$\mathbf{A}^C_{i,j} = \begin{cases} \left|d(\mathbf{v}'_i, \mathbf{v}'_j)\right| & \left|d(\mathbf{v}'_i, \mathbf{v}'_j)\right| \geq \tau_C \\ 0 & \left|d(\mathbf{v}'_i, \mathbf{v}'_j)\right| < \tau_C \end{cases} \quad \text{where } 0 \leq \tau < 1$$

- *Random Erdős–Rényi graph*: Since we noticed that a great portion of the images features were fixed to 0, because output of a padding operation, we decided to use them to fit more information into the image features tensor. To do so we created a new graph defined as a Random Erdős–Rényi graph with probability $p = 0.2$. This allows the GCN to exploit also these empty nodes which were unused before.

Graph Convolutional Network Our GCN is made of 3 layers, each one with residual connections and dropout. We pre-trained this model without modifying the input. This is done to avoid worsening the performance of the original OSCAR model.

Training Since OSCAR is a model that can be used for a variety of tasks, the first thing we did was fine-tuning it for Image Captioning. Due to our computational limits, we performed the fine-tuning for 50 epochs on a subset composed by 10k images. This represents our starting baseline. Finally, we trained the whole Augmented-OSCAR for 3 epochs on the same 10k images.

4 Results

As mentioned before, we trained and tested using Coco Captions.

	`Bleu1`	`Bleu2`	`Bleu3`	`Bleu4`
Baseline	62.9	45.8	32.9	23.8
Augmented OSCAR	62.5	44.0	30.6	21.4

5 Conclusion and future work

The focus of this work was to test if a SoTA model such as OSCAR could benefit from *augmented* features derived from a graph.

As we can see from the results, the performance of the model is worse than the baseline. We could see this also during fine-tuning, since the loss was augmenting at each step. This could be caused by a variety of reasons, one of them is the ER graph that could add too much randomness to the process. Another possible reason is that OSCAR may be needing more time to adapt to the new features, creating some sort of *domain shift*.

Since the GNN does not change the connectivity structure of the graph, the construction of the graph itself plays a fundamental role in the quality of these features. Therefore, this work is prone to exploration of new methods for the construction of the graph, such as

- The use of object tags embeddings $\mathbf{q}$ to make a new graph based on the similarities between every pair of object tags in the image. This new graph could be combined with the object features graph that we used.
- The use of specific methods for scene graph generation, such as [3][4].

References

[1] Xiujun Li et al. "Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks". In: *CoRR* abs/2004.06165 (2020). arXiv: 2004.06165. URL: https://arxiv.org/abs/2004.06165.

[2] Thomas N. Kipf and Max Welling. "Semi-Supervised Classification with Graph Convolutional Networks". In: *CoRR* abs/1609.02907 (2016). arXiv: 1609.02907. URL: http://arxiv.org/abs/1609.02907.

[3] Kaihua Tang et al. "Unbiased Scene Graph Generation from Biased Training". In: *CoRR* abs/2002.11949 (2020). arXiv: 2002.11949. URL: https://arxiv.org/abs/2002.11949.

[4] Mohammed Suhail et al. "Energy-Based Learning for Scene Graph Generation". In: *CoRR* abs/2103.02221 (2021). arXiv: 2103.02221. URL: https://arxiv.org/abs/2103.02221.