Image Captioning via CLIP Prefix Tuning

Peiyan Song, Veronika Grytsai, Patrick Rui De Jing

Introduction

As for our final project, we aimed to explore how powerful pre-trained vision and language models like CLIP and GPT-2 can be used for image captioning without fine-tuning the large models themselves. This project is a reimplementation of “ClipCap: CLIP Prefix for Image Captioning” by Ron et al.s We chose this paper because it proposes an efficient solution to bridge vision and language using minimal training. The idea behind it is to train a neural network that can successfully connect two models and provide meaningful results.

Methodology

1. We start with extracting the embedding vector from images using **TFCLIP**(OpenAI’s CLIP-like model presented by HuggingFace that is specialized for TensorFlow).

    - CLIP (Contrastive Language-Image Pre-training) is a multimodal model developed by OpenAI that can understand and relate images and textual descriptions in a unified manner. Here in our model, CLIP takes an image as input and outputs a single embedding vector.

2. To connect TFCLIP and GPT-2 we are using MLP/Transformer to convert those embeddings to a sequence of prefix tokens. This part is being trained to successfully convert data between two frozen models - TFCLIP ang GPT-2.
    - Fine-tuning CLIP does not benefit resulting quality, but does increase training time and complexity. To keep the model lightweight and minimize the training time we keep CLIP frozen.
    - MLP is sufficient for mapping CLIP embeddings to GPT-2 space when the language model is fine-tuned, because both components can adapt together. However, when the language model is frozen, a transformer is preferred due to its greater expressive power and ability to leverage global attention.
3. GPT-2 Language Model takes the prefix as input and generates the caption word by word during inference.
4. The loss function used is cross-entropy loss, computed between the predicted token logits and the ground-truth token sequences.

The datasets involved in training were a subset of COCO and full Flickr30. COCO Captions Dataset(subset taken: 56674 captions 33,841 for train 12,505 for val), which contains over 330K images, five independent human generated captions are provided for each image. Flickr30k dataset consists of 31,783 images that capture people engaged in everyday activities and events. Each image has a descriptive caption.

Results

- For evaluation metrics we chose BLEU-4. BLEU-4(Bilingual Evaluation Understudy with 4-gram) is a metric for evaluating machine-generated text by comparing it to reference translations using 1-gram to 4-gram precision. It penalizes short translations and outputs a score between 0 and 1, with higher scores indicating better quality. The evaluation compares BLEU-4 scores for captions generated from 193 images using two models. The MLP model with fine-tuned GPT-2 achieved a higher average BLEU-4 score of 0.287, indicating better alignment with reference captions, while the Transformer model with frozen GPT-2 scored 0.274, suggesting slightly lower performance.

- As for the loss, we observe that using a Transformer mapper with a frozen GPT-2 leads to faster convergence — the model reaches a low training loss more quickly. This is likely because the frozen GPT-2 provides stable and consistent language generation, allowing the mapper to adapt rapidly without destabilizing the overall model. While on the other hand, the MLP mapper combined with a fine-tuned GPT-2 achieves a lower eventual loss. Although it converges more slowly, fine-tuning GPT-2 allows the model to better adapt to our specific dataset, leading to improved performance in the long run.

To sum up, our MLP version performed slightly better than the Transformer one. It was also proved by the manual inspection of some captions created by both of them. It seems like the model that uses Transformers used basic words, produced repetitive captions (e.x. “a dancer dancing”), and sometimes missed the logical background of the image. While a model with MLP tended to use different words and be more descriptive.

In the paper, they got better results while using the Transformer version. It is expected because the Transformer has greater expressive power and uses global attention. In our case, MLP performed better probably because it has more flexibility as being able to fine-tune the language model. Also, MLP likely overfits better to the dataset’s captioning style, increasing n-gram overlap and improving BLEU-4. While Transformer can miss specific phrases or word orders found in references, reducing BLEU-4 despite producing high-quality captions.

We reached both our base and target goals from the initial project write-up. As we successfully trained two models, evaluated the results and compared their performances. Even though our results do not match exactly the paper, we find it to be a possible outcome of using less data and trying to minimize the training even more.

Challenges

One of the biggest challenges we faced was reimplementing the paper using a different package. Original paper uses PyTorch and the original CLIP, which was produced by OpenAI in PyTorch. We had to look for alternatives and the best one we found was - HuggingFace TFClip (CLIP TensorFlow version). It seemed to be enough for our project, but it might be not as powerful as the original CLIP, which led to worse results and potentially has more hidden errors in it.

As for other challenges, training the model took time and we had to accommodate for it as we were running it on a local machine. Also, the original paper uses several evaluation metrics, but we decided to choose only one, which ended up being BLEU-4.

If we were to start over the project, we would be more structured. We started with having many ideas and not enough understanding of how to implement it. During the course of the project our base goal slowly switched to stretch goal as we realized that it is harder and more time-consuming than we initially expected. Moreover, we would structure our group work differently as expectations from each member were not clear at all times.

Reflection

We are overall satisfied with the outcome of this project. If we had more time, we would implement video captioning for short videos ~5s. The simplest approach is just to treat each frame separately and then combine it to an embedding vector. As for other ideas, we could train the model more to be able to support more than one sentence output and maybe storytelling ability. This project taught us group work, resilience and patience. It was a great opportunity to be close to a real-life experience doing a project. We found ourselves at first thrown into the great variety of papers to reimplement, and even after finding the one, we underestimated how hard and time-consuming it would be. However, it taught us to be structured, search for information and solve the problems as soon as they appear. Also, a part of the project was compromising and supporting each other, which we succeeded in.