

Image Captioning with Deep Reinforcement Learning

Final Project report | SJSU CMPE-297-03 | Spring 2020

Presented to:

Prof. Jahan Ghofraniha

Team:

Pratikkumar Prajapati (012541477)

Aashay Mokadam (012724998)

Karthik Munipalle (013854867)

Table of Contents

Table of Contents	2
Executive summary	3
Introduction	3
The problem formulation	3
The A2C model of the agent	4
Problem Statement	5
Motivation	6
Differentiators	7
Bidirectional RNN	7
Jump-start with corpus-trained word embeddings	7
Experimenting with curriculum and non-curriculum	7
Methodology	8
Network Pre-training	8
A2C Training	9
Implementation and Results	10
Implementation	10
Results	10
Contributions	12
Conclusions	13
Appendix	15
Code	15
Dataset	15
References	15

Executive summary

Image captioning is the task of an automated description of images. This problem involves multiple goals to achieve: detecting various aspects of an image and generating text that describes the identified aspects. In recent years, there have been many deep learning models developed to tackle this task. One such prominent approach was to use a Convolutional Neural Network (CNN) for image recognition, chained with a Recurrent Neural Network (RNN) to generate the corresponding sequence of text. More recently, policy gradient approaches from Reinforcement Learning were used to further improve the image captioning models.

We have used the Advantage Actor-Critic (A2C) approach, in which our Actor was the traditional policy network with CNN and RNN, and the Critic is a value network that is used to choose the most suitable action. Then we used a visual-semantic embedding reward to estimate the reward. In addition, we used curriculum training to reduce the action space and train the model on smaller caption tasks and gradually increase the caption size. We also experimented with Bidirectional RNN instead of the left-to-right representation to improve the prediction of the sequential context of the captions.

All experiments were evaluated using the MS-COCO image captioning dataset. This is a collection of 123,287 captioned images, each with 4-5 human generated captions.

The A2C model was able to generate captions closer to human-generated captions of an image. We used common Natural Language Processing (NLP) metrics like BLEU, ROUGE_L, METEOR, and the new metric used for image captioning known as CIDEr. The model was able to achieve decent scores on all these metrics using the A2C Paradigm.

Our best model achieved a BLEU 1 score of 0.3051, METEOR score of 0.1126, and a CIDEr score of 0.497.

Introduction

In this section, we explain how the image captioning problem can be expressed as a Reinforcement Learning problem and what advantage we can get by using the RL-based approach.

The problem formulation

The image caption problem can be formulated in terms of the decision-making problem and so we can apply the paradigm of Reinforcement Learning. With the typical setup of Reinforcement Learning, the agent interacts with the environment by taking certain actions by using the current

set of observations. The environment returns the rewards and the next set of observations. Figure 1 shows the main components of the system. With the image captioning problem, the agent's actions can be considered as predicting the captions, and rewards can be the correctness of the captions. The image and its captions can be regarded as an environment. With each action, the environment can return observations in terms of captions generated so far and the image itself.

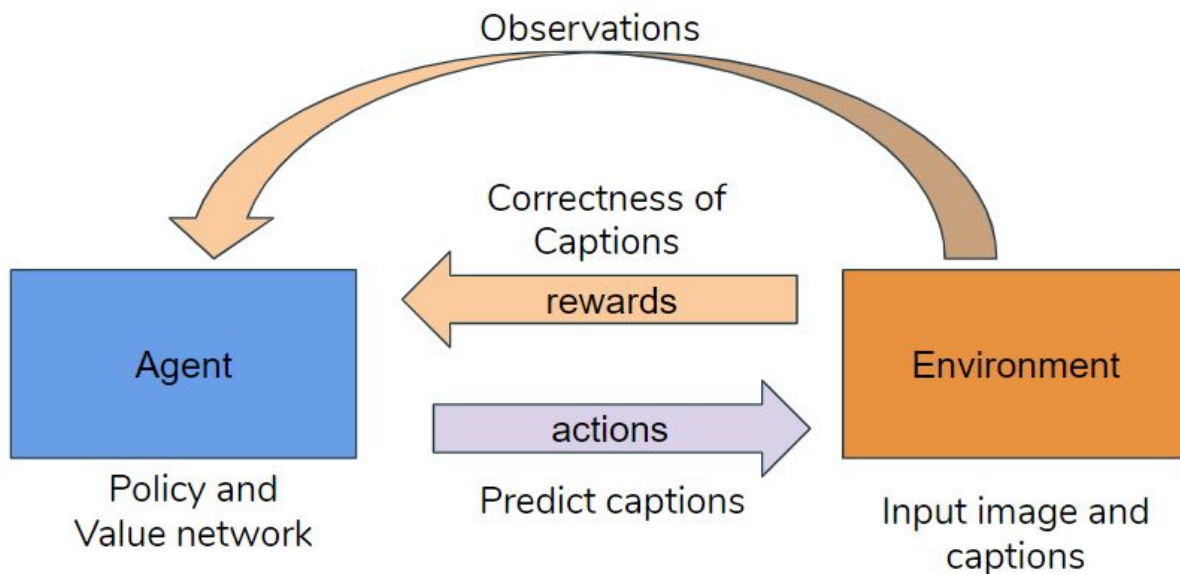


Figure 1. Representing the problem of Image captioning in the Reinforcement Learning paradigm

The A2C model of the agent

In this case, the agent is the pair of neural networks to make the decision on the right captions for the given image. We use an Actor-critic model to learn about the image features holistically. Policy network and Value network work together to predict the whole caption. The Policy network works as an actor and can learn the policy gradient to predict the probability distribution of the matching captions. This interface can give a local view of the image as learned by the policy network. Many times by only using the policy network can work to predict the captions of the image but most of the time it would just look at the local view of the image as only some of the highest probability captions would get selected by the policy network. By introducing the Value network as the critic the overall model can visualize the global image view. The Critic network would use beam search to look-ahead the overall captions and give value to the current state i.e., image and the captions chosen so far.

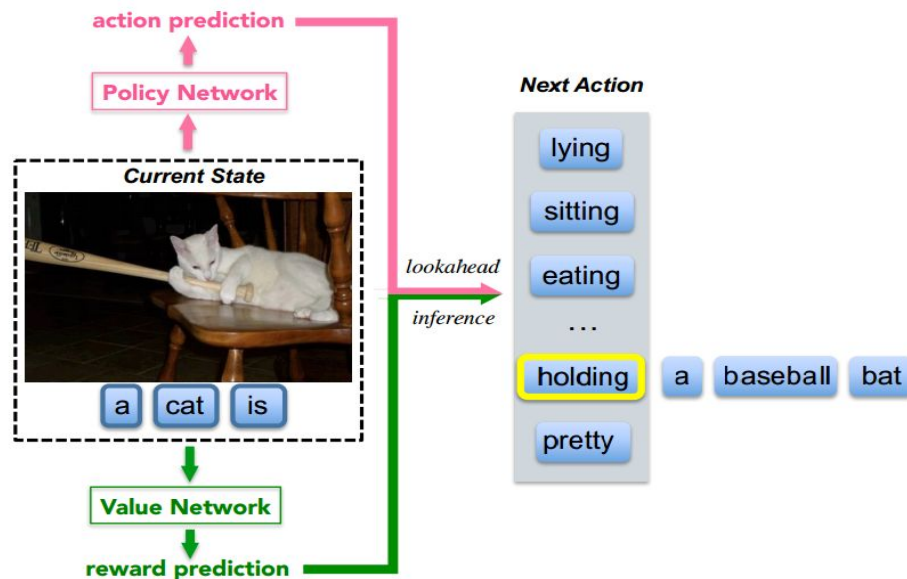


Figure 2. Example of Policy and Value network as A2C model [1]

Let's consider an image of a cat holding the baseball bat and sitting on a chair, as shown in Figure 2. The Policy network would generate a probability distribution of possible words. In this example, the agent would choose either of "lying", "sitting", "eating", "holding", etc. words. There is a good chance that "lying" gets the highest probability and the "lying" words get selected as an action by the agent. But when value networks get activated, it would look-ahead and find next of possible words and give the value of the current action based upon that. The value network is trained such that it gives high value to captions matching the overall visual-semantics of the image. In this given example the value network gives high value to the word "holding" as it would capture the global view of the image. In a nutshell, both the policy and value network works in harmony to maximize the rewards by predicting captions to capture the global view of the image. This would be the most important advantage of using the Reinforcement Learning-based A2C model.

Problem Statement

The high-level problem statement is to predict the captions of the given images. The goal is to design a system that automatically describes the given images as accurate as a living person might describe. The overall gain of the problem is useful for the reasons mentioned in the Motivation section, later in this document.

The image captioning problem molded into a Reinforcement Learning paradigm can be seen as a decision-making problem. The Agent needs to make decisions to predict correct captions of

the given images. The actions of the agent would be to predict captions word-by-word and at the word, the prediction is considered as an action. The environment of the problem set is the input images and captions associated with each image. The environment returns the rewards based on actions taken by the agent. The environment also returns the observations as the image and captions generated so far. The Agent needs to learn to predict the correct captions by maximizing the rewards.

Motivation

Automatic image captioning has many use cases. In social media applications where inferring content from images can be really helpful to flag images for inappropriate content. An automatic description of satellite imagery can also be helpful in creating reports from images. Automatic image captioning can help visually impaired people. Text generated from the image can be fed to a text-to-speech model that reads out the description for the visually impaired user. It can be used in many applications that help visually impaired users. One more interesting application is in healthcare that can help create automated reports from scanning images, that reduce the human effort and increases the efficiency of health care services. It can be used in the frame by frame video description and can find its applications in security logging and monitoring applications.

Automatic image captioning can be very helpful in extracting important information from images and stored as text. Text-based information is easy to disseminate compared to the storage overhead caused by complex files like images and videos. We wanted to explore the application of Reinforcement learning in this task, as recent studies proved that the image captioning models can benefit a lot from decision-making frameworks like reinforcement learning. Moreover, image captioning involved both NLP and Computer Vision, two fields of study with many applications in machine learning. We wanted to explore and solve this problem using adversarial learning like Generative Adversarial Networks (GANs), then we looked into policy gradient applications in image captioning. As GANs and A2C approach are similar to each other, we built an A2C model for image captioning.

Differentiators

We implemented three different approaches compared to the literature available on image captioning: (1) We used bidirectional RNNs instead of unidirectional RNNs, (2) We used pre-trained word vectors instead of the PyTorch's embedding layer, and (3) We experimented by trying non-curriculum and curriculum-based training to analyze the results.

Bidirectional RNN

Bidirectional RNNs have outperformed the traditional unidirectional RNNs in language modeling tasks in recent years. They combine left-to-right and right-to-left representations of the input. Unlike unidirectional RNNs that predict the probabilities of the words based on the words in the past, the bidirectional RNNs capture both past and future context of the sequence and can improve the overall prediction accuracy of the language model [3]. We used bidirectional RNNs for the same reason because they improve the overall performance of language models as words can be predicted more efficiently when both past and future contexts are known. Moreover, the bidirectional inference of context can also help in the lookahead inference we are using for caption generation. Therefore, we implemented all the networks with Bidirectional RNNs, including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU).

Jump-start with corpus-trained word embeddings

Words must be represented in numbers for the neural network to understand, words can be expressed as vectors either by using the one-hot vectors or neural network-based models to create word vectors. Word2Vec is one such model that predicts a center word from a given context of words. Many word embedding models are available that can be loaded onto the network as word embeddings. Training word embeddings often provide a jump-start to NLP models that require vector representations of the words. Pretrained embeddings often help the models converge faster and improve the overall performance of the language model. Therefore, we trained and used fasttext word vectors (Skip-Gram training) with 300 dimensions as word embeddings for our model.

Experimenting with curriculum and non-curriculum

Curriculum learning is the process of training the machine learning model on smaller subtasks with an increasing difficulty similar to how humans learn from easier to hard tasks. We used curriculum learning as suggested by [1] to decrease the action space of the agent to improve the overall performance of the model, albeit in a rudimentary manner. In addition to this, we also tried different step levels of training. Initially, we experimented with a step size of 2, then we tried the step size of 3. The results obtained from these experiments are shown in the Implementation and Results section.

Methodology

The approach we follow is heavily inspired by the methodologies stated in [1], which is using an actor-critic training paradigm in order to model text based on a Visual-Semantic Embedding Reward function. The agent is a trained actor-critic model, the action space consists of predictions for the next word in a caption, and the observation is an image along with the caption generated so far. Hence for a full prediction cycle, the trained agent would receive an image and a <START> token as the initial observation. It would then predict the caption one word at a time, receiving a reward (see Reward Network below) and the updated caption so far as the next observation.

Network Pre-training

There are three major components (reward network, policy network, value network) that implement this approach. Training is performed in two phases - first, the Reward, Policy, and Value networks are trained independently (in that order), and then the Policy and Value networks are trained using Deep Reinforcement Learning in an A2C paradigm using an advantage-weighted loss function.

1) Reward Network

The objective of this network is to act as an environment that provides feedback to the agent in the form of a visual-semantic embedding similarity. This is done in two steps; first, the sequence of tokens (i.e. image caption) is put through an embedding layer followed by a Gated Recurrent Unit layer - these jointly capture the content and sequential interdependencies of the caption. Then, both the image features (obtained from a pre-trained VGG16 network) and the caption data (taken as the final hidden state output of the recurrence-based network) are projected onto a shared vector space.

This network is trained by minimizing the Visual-Semantic Embedding Loss [4] between the two embedded components.

This is summarized in the image below, where:

$f_e(v)$ - a linear layer projection of image features (from CNN) 'v'

$h'_T(S)$ - linear layer projection of last RNN hidden state for caption features 'S'

$$L_e = \sum_v \sum_{S^-} \max(0, \beta - \underbrace{f_e(v) \cdot h'_T(S)}_{\text{Maximize similarity between image 'v' and caption 'S'}} + \underbrace{f_e(v) \cdot h'_T(S^-)}_{\text{Minimize similarity between image 'v' and false caption 'S-'}}) + \sum_S \sum_{v^-} \max(0, \beta - \underbrace{h'_T(S) \cdot f_e(v)}_{\text{Maximize similarity between caption 'S' and image 'v'}} + \underbrace{h'_T(S) \cdot f_e(v^-)}_{\text{Minimize similarity between caption 'S' and false image 'v-'}})$$

Minimize Loss

Once the semantic and visual information has been captured, we calculate the reward as the similarity between their embeddings via a cosine similarity.

2) Policy Network

This network is the 'actor' part of the agent and learns a policy function to predict the immediate next word for a caption. It accomplishes this through language modeling (i.e. predicting the next word given a sequence of words). The network consists primarily of an LSTM layer, whose hidden state is initialized with image features (after a linear layer projection to the appropriate dimension).

Given an image and its caption, this network is trained by minimizing the Cross-Entropy Loss between the probability distribution of predicted words and the actual caption.

3) Value Network

This network acts as the 'critic' of the agent and is responsible for predicting the expected long-term reward for an action (next-word prediction) given an observation (image + caption so far). The network first obtains a sequence representation of the caption (via an LSTM layer following embedding the words into vectors), which is then combined with the image feature representation (from the pre-trained VGG16 CNN embedding). The image and caption embeddings are then combined into a single vector, which is then fed into a feed-forward multi-layer perceptron that converts the input features to a scalar value. This is the total expected reward.

Once we have a trained policy network - providing local, "short-term" guidance on what the best immediate action (i.e. next word prediction) for a caption would be, as well as a trained reward network - representing an acceptable simulation of an environment which rewards the agent based on how correctly a caption describes an image, we train the value network. We start off by creating a caption via greedily selecting actions predicted by the policy network, i.e. repeatedly selecting the most probable next word given an image. We then use the trained reward network to obtain a reward for the caption. Finally, we sample the first 'n' words of the greedily generated policy network caption - 'n' selected randomly, corresponding to a random state in the sequence generation - and use it along with the image features to train the value network by minimizing the Mean Squared Error of the output with respect to the obtained reward.

A2C Training

Once the agent (Policy network + Value network) and environment (reward network) have been trained, we freeze the reward network and jointly train the agent using deep reinforcement learning on advantage-weighted gradient losses for the value function and the logarithm of the policy function. Additionally, a few different variants of the A2C model have been trained, with different degrees of effectiveness. These are discussed in the next section. Once the entire

model was trained, captions were generated for test/validation set images using an iterative beam-search lookahead approach based on the policy and value network predictions.

Implementation and Results

In this section, we would discuss implementation details, results, and contributions of each team member.

Implementation

Each of the individual networks (Reward, Policy, Value) were pre-trained on their respective loss functions for 50 epochs. We used Adam optimizer for all networks with a learning rate of 0.001.

The following variants of models are trained:

- With and without word embedding (FastText [5]) initializations
- With vanilla and bidirectional variants for all recurrence-based networks[#] involved
- With and without basic curriculum learning

[#]As mentioned in the methodology section, these networks refer to a GRU layer for the Reward Network and LSTM layers for the Value and Policy Networks.

Results

Configuration [#]			Number of Training Epochs	Scores	
				BLEU_1	CIDEr
No Word Vector Initialization	Simple LSTMs and GRUs	Full RL Training	10	0.305	0.497
		Curriculum Learning	5	0.111	0.186
	Bidirectional LSTMs and GRUs	Full RL Training	5	0.010	0.0004
		Curriculum Learning	2	0.0051	0.0010
FastText Vector Initialization	Simple LSTMs and GRUs	Full RL Training	10	0.042	0.018
		Curriculum Learning	5	0.042	0.017
	Bidirectional LSTMs and GRUs	Full RL Training	5	0.015	0.0091
		Curriculum Learning	2	0.0064	0.0056

Table 1. Results of the training with various configurations. Best model configuration is highlighted in bold typeface

Best model configuration is highlighted in bold typeface in Table 1. The sample images with real and generated captions with our best performing model can be seen in Figure 3.




	<p>Real caption: the slice of pizza has large <UNK> of tomatoes on it</p> <p>Generated cap: plate with a slice of pizza covered in it</p>
	<p>Real caption: young girl about to throw a frisbee in a park</p> <p>Generated caption: a young girl in a white dress holding a surfboard</p>
	<p>Real caption: brown teddy bear with a bow sitting on a chair</p> <p>Generated caption: a brown teddy bear wearing a blue shirt with a laptop</p>
	<p>Real caption: a man on a snowboard in the mountains</p> <p>Generated caption: a person carrying a snowboard on a snowy surface</p>

Figure 3. Sample images with real and generated captions with our best performing mode.

Metrics Used

- BLEU_1 is the overlapping n-grams in real and generated captions.
- CIDEr is the average cosine similarity between generated captions and real captions (human-generated captions).

One caveat to these metrics is that they rely on similarity (n-gram or embedding) of machine generated captions to human generated captions. While these are a good rough estimator of results, they do not necessarily indicate the quality of text generated by a model.

Generated

Gen: <START> an image of family having dinner and posing <END>

Real: <START> several <UNK> sitting around a table in a restaurant <END>

Gen: <START> an apple and orange in an <UNK> <UNK> <END>

Real: <START> a orange <UNK> that is on a wooden table <END>

Note : <UNK> : unknown token , <START> : special start token, <END> : special end token.

Contributions

Worked as a team to evaluate multiple projects, investigated various approaches, and the possible inclusion of Reinforcement Learning in the problem. Designed the overall flow of the image captioning project. Also, we worked on bug fixing and documentation.

Pratikkumar

- Developed the core pipeline for our project.
- Implemented the value network and policy network as Actor-Critic
- Developed training and testing routines.
- Implementing post-processing of the model outputs to find the best performing caption.

Aashay

- Enhanced the networks by adding Bidirectional RNNs in the Policy and Value networks.
- Core bug fixes for memory leaks.
- Added the visual embedding based reward network.
- Training fasttext word embeddings to jump-start RNN inputs.
- Beam search and further optimized A2C training.

Karthik

- Investigated and procured the dataset.
- Enhanced the agent training with a curriculum learning-based approach.
- Bug fixes and refactoring code.
- Worked with Pratikkumar in implementing the validation phase.
- Worked on scoring metrics for validation of the generated captions.

Conclusions

We have implemented multiple variants of an A2C based Deep Reinforcement Learning approach for automatic image captioning. A major trend in results is that more complex models tend to perform quite poorly, and need more training. This is evident in bidirectional and curriculum learning models. Constraints on time and computing resources limited the amount of training that could be accommodated. Moreover, bidirectional RNNs take more time to train compared to their unidirectional variants.

These shortcomings can be more closely observed by analysing the generated captions of one such model. It is evident that these networks fall short in language modelling capabilities, and likely would perform much better with more training:

- Real: <START> a pair of giraffes <UNK> an <UNK> <UNK> <END>
Generated: <START> a giraffe standing next food food as food as food food food food food food food
- Real: <START> a few people on skis skiing on a snowy mountain <END>
Generated: <START> a skier coming as as as she as she as she as she as she as

Based on the above observations, it is evident that there is scope for future improvement. Solutions to some of the major shortcomings in performance are:

I. **Longer Training - More Epochs**

Considering that some of the poorly performing models are not able to model language effectively, it is clear that some of our models suffer from inadequate training. Some of the models in question were trained on very few epochs too (as seen in the Results table) due to time and computing restrictions. Given the complexity of the deep learning networks used as well as the overall size of the dataset (over 400k training examples), increasing the overall training epochs is the most obvious avenue for improvement.

II. **Better Curriculum Learning**

While curriculum learning is a valuable approach for complex tasks such as this one, our approach suffers from poor convergence and is perhaps a little too rudimentary. A notable improvement would be to implement a more sophisticated variant of curriculum learning, like MIXER [6], where at each curriculum level, we add cross-entropy training on the portion of the sequence being omitted by reinforcement training.

III. **Better Language Modelling**

A long-standing critique of recurrence based networks has been their inability to model long sequences effectively. This shortcoming is certainly felt by some of our models,

where the generated captions are coherent (and often, surprisingly detailed) up to a certain point, beyond which they descend into gibberish. This is the case with the curriculum learning model without a bidirectional enhancement. Given recent improvements in Natural Language Processing which omit recurrence altogether, notably the Transformer Network [7] and its pre-trained transfer-learning cousins (e.g., BERT, T5), exploring these techniques to improve our action-space encoding and policy formulation are promising directions for future work.

IV. **Flexible and Pretrained Embeddings**

One major original addition was the use of FastText Word Embeddings trained on the corpus of all image captions. This was intended to jump-start language modeling and allow the policy, value, and reward functions to converge faster. However, these models ostensibly performed badly. One possible explanation is that in our experiments so far, once the embeddings were trained and the vectors were fed into our recurrent networks, we froze their weights in a bid to leverage their features fully. Perhaps this hindered performance by causing a mismatch in alignments of vector spaces for word embeddings and image feature embeddings. Therefore, allowing the trained word vectors to be mutable by gradient updates during the training of the RNNs might potentially alleviate this issue.

V. **Exploring Deep Q-Learning**

Given that our problem is a partially observable Markov Decision Process, the approaches specified in [8] could be a worthwhile direction for exploration. The model is said to perform well when only observing one frame (Atari image data) at a single time step, and benefits from training on partial observations and evaluation on full observations. These aspects of the algorithm could be modified to fit the image-captioning task and benefit our performance.

Appendix

Code

The code of the project can be found at: [github repo](#)

Dataset

Follow this [link](#) to get the zipped version of the data. You can extract all the files from the zip file and place them in the **datasets/** folder.

References

- [1] Zhou, R., Wang, X., Zhang, N., Lv, X., & Li-Jia, L. (2017). Deep Reinforcement Learning-based Image Captioning with Embedding Reward. *ArXiv.org*, ArXiv.org, Apr 12, 2017.
- [2] Shi, Haichao, Li, Peng, Wang, Bo, & Wang, Zhenyu. (2018). Image Captioning based on Deep Reinforcement Learning. *ArXiv.org*, 13.
- [3] Chen, X., Liu, X., Wang, Y., Ragni, A., Wong, J., & Gales, M. (2019). Exploiting Future Word Contexts in Neural Network Language Models for Speech Recognition. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 27(9), 1444-1454.
- [4] Frome, Andrea, et al. "Devise: A deep visual-semantic embedding model." *Advances in neural information processing systems*. 2013.
- [5] Bojanowski, Piotr, et al. "Enriching word vectors with subword information." *Transactions of the Association for Computational Linguistics* 5 (2017): 135-146.
- [6] Ranzato, Marc'Aurelio, et al. "Sequence level training with recurrent neural networks." *arXiv preprint arXiv:1511.06732* (2015).
- [7] Vaswani, Ashish, et al. "Attention is all you need." *Advances in neural information processing systems*. 2017.