Computer Vision (EC-353)
A Modern Approach to Image Captioning

Ramansh Grover
Department of Computer Science and Engineering
Delhi Technological University
Delhi, India
`ramanshgrover_2k18co281@dtu.ac.in`

December 1, 2020

1 Introduction

In the past few years, recent developments in Image Captioning Systems have been inspired by advancements in object detection and machine translation. The task of image captioning involves two main aspects: (1) resolving the object detection problem in computer vision and (2) creating a language model that can accurately generate a sentence describing the detected objects.

Seeing the success of encoder-decoder models with "soft" attention, I use soft alignment (Bahdanau et al., 2016) and modern approaches to object detection (Ba et al., 2015) as a baseline model. To extend this work, I investigate the effect of pre-trained embeddings by integrating GloVe (Pennington et al., 2014) and BERT (Devlin et al., 2019) context vectors to enhance the models performance and reduce training time. Through the medium of this project, I contribute the following:

- Provide a fresh pyTorch implementation of the soft deterministic attention mechanism with an encoder-decoder architecture for image captioning as described in (Xu et al., 2016).
- Integrate context vectors from BERT and GloVe embeddings into the baseline model and enhance its performance.
- Finally, visualize the results and quantitatively validate all three models with the validation dataset.

The source code is publicly available on Github.

2 Problem Description

Given a single raw image, the goal is to generate a caption **y** encoded as a one-hot vector corresponding to the vocabulary.

$$y = \{y_1, ..., y_C\}, y_i \in R^V$$

Where V is the size of the vocabulary and C is the length of the caption.

3 The Dataset

For image captioning, there are several publicly available datasets for training and validating models (MS COCO, Flickr8k, Flickr30k). For the scope of this project, I opted for the Microsoft's Common Objects in Context (COCO) 2014 dataset (Lin et al., 2015) for both training and validation readily utilizing the pyCOCO API for cleaning and structuring scripts (Vinodababu, 2019; Park, 2018) to parse the captions, extract the vocabulary, and batch the images to optimize the training process for the models. Table 1 shows an overview of its descriptive statistics.

Split	**Training**	**Validation**	**Testing**
Examples	82,783	40,504	40,775

Table 1: Descriptive Statistics, MS COCO 2014 Dataset

After re-sizing and normalizing all the images to 224x224 pixels, I extracted and tokenized the captions with the NLTK tokenizer. Following that, I built a vocabulary with all the training dataset words, which came to be a list of 8,856 words.

4 Models and Algorithms

As suggested in (Xu et al., 2016), I use an encoder-decoder architecture to generate captions. Here, the encoder is a Convolutional Neural Network (CNN) which acts as a feature descriptor taking in a single image to generate a vector that describes the detected objects. This vector of objects is then passed to the decoder, which is a Long Short-Term Memory Network (LSTM) that attends to the image and outputs a descriptive caption one word at each time step.

In the subsections below, I describe the encoder used in the models and the three variants of the attention-based decoders. The first decoder is an exact replica of the soft attention model with optimized hyper-parameters. This initial model will act as our baseline. The second and third ones are an extension on the baseline model integrating GloVe embeddings and BERT's pre-trained context vectors and are hence, resized.

4.1 Encoder

Similar to the encoder described in (Xu et al., 2016), I used a CNN to extract feature vectors from images. The Encoder produces L vectors, where each vector has D-dimensions that represent part of the image.

$$a = \{a_1, ..., a_L\} \in R^D$$

Although it's possible to create and train a CNN of my own, for this project, I used the pretrained ResNet-101 CNN as our encoder to reduce the training time and focus on enhancing the performance of the decoder. To use ResNet-101, simply discard the pooling and linear layers - the last two layers - as we only need the image encoding, rather than the image classification. Following this, pass the output of the modified ResNet onto an adaptive pooling layer to create a fixed size output vector –

fixed L – that can be easily passed to the decoder. The Encoder is kept intact and no fine-tuning is performed in the ResNet-101.

4.2 Decoder: Baseline Attention Model

I use a Long Short-Term Memory network to generate caption words one step at a time by conditioning on the previous step's hidden state, the context vector, and the previously generated words. This implementation directly follows that of (Xu et al., 2016; Vinyals et al., 2015).

$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ tanh \end{pmatrix} T_{D+m+n,n} \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ h_{t-1} \\ z_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot tanh(c_t)$$

Where, i, f, o, and g are the input, forget, memory, and output states of the LSTM.
h is the hidden state, c is the cell state (that keeps long term memory), and h_t denotes the hidden state at timestep t.
Additionally, $T_{n,m}$ defines an affine transformation from dimension n to m and $\odot$ is an element-wise multiplication.

E represents an embedding matrix that is used and z_t denotes the context vectors of the relevant part of the image at time step t that is generated through soft-attention. To perform soft-attention in generating z_t, the "soft" attention mechanism (detailed in (Xu et al., 2016)) is used. In this baseline model, the caption embedding, **E**, is learned alongside training the model.

In the next two decoder descriptions, I will explain how **E** is optimized to enhance the model's performance.

4.3 Decoder: GloVe Attention Model

Recent methods for extracting and learning vector space representations for words have proven successful in capturing fine-grained semantic and syntactic regularities in words. In particular, GloVe: Global Vectors for Word Representation (Pennington et al., 2014) created a global log-bilinear regression model that generates word vector representations that enable Machine learning models to utilize these pre-trained embeddings. These embeddings are helpful because they can be pre-trained on vast amounts of text data instead of being trained alongside the task specific model, which usually has a much smaller dataset leading to class biases and overfitting.

As an extension to the baseline decoder explained in the previous section, GloVe embeddings are integrated into our decoder by applying them to the images captions. Further, these embeddings were fine-tuned along-side the model (as it trained) to increase it's accuracy and make it better fit the MS COCO dataset captions.

I opted for the 6B token based 300-dimensional word vectors pre-trained on Wikipedia Data introduced in (Pennington et al., 2014) and built a weights matrix that contains a GloVe embedding for every word in our vocabulary. The decoder embeddings are then initialized with these weights and fine-tuned alongside the model by propagating back the gradients.

4.4 Decoder: BERT Attention Model

In using the GloVe vector representations, each word is represented by a single unique vector no matter what context the word is used in. This raised concerns for several researchers (Devlin et al., 2019) as they realized that each word could have multiple meanings depending on where the word is used. Rather than having one representation for each word, BERT uses a Transformer to generate a bi-directional contextualized word embeddings conditioned on the context of the word in a sentence.

BERT has two distinct models, BERT base and BERT large. The base version has 12 encoder layers in its Transformer, 768 hidden unites in its feedforward-network, and 12 attention heads. On the other hand, the large version has 24 encoder layers in it's Transformer, 1024 hidden unites in it's feedforward-network, and 16 attention heads. Throughout this implementation, I used BERT base to generate the caption's contextualized word vectors due to the increase in training time the large model introduces.

The decoder takes a batch of captions as our input $\mathbf{c} = \{c_1, ..., c_B\}$, where B is the size of the batch and c_i is a full text representation of the caption. It then iteratively takes each caption c_i and perform the following steps to it:

1. Tokenize each caption with BERT's wordPiece tokenizer to enable BERT to digest the caption and add the special '[CLS]' BERT token to the beginning of the caption
2. Pass the wordPieces into BERT base
3. Retrieve the output of the 12th layer (last layer) and discard the embedding of the special '[CLS]' token
4. Detokenize the embeddings by summing the BERT context vectors of wordPieces that belong to the same original word.

After doing the steps above to each caption in the batch we will have caption embeddings $\mathbf{b} = \{b_1, ..., b_B\}$, where b_i is a tensor of size (caption size x 768) as each word has a vector of size 768 as it's contexualized embedding. **b** can then directly replace the GloVe embeddings and the trained embeddings used in the baseline model and the GloVe model respectively.

5 Experiments and Results

The Baseline, GloVe, and BERT models were trained and validated using the MS COCO 2014 dataset with the following optimized hyper-parameters obtained from (Xu et al., 2016). Table 2 consists of these hyper-parameters with possible justifications as to why I used what I did.

When it came down to implementing the embedding extensions, I used embedding dimension of 512 for the baseline model, 300 for GloVe, and 768 for BERT. All the models were trained and

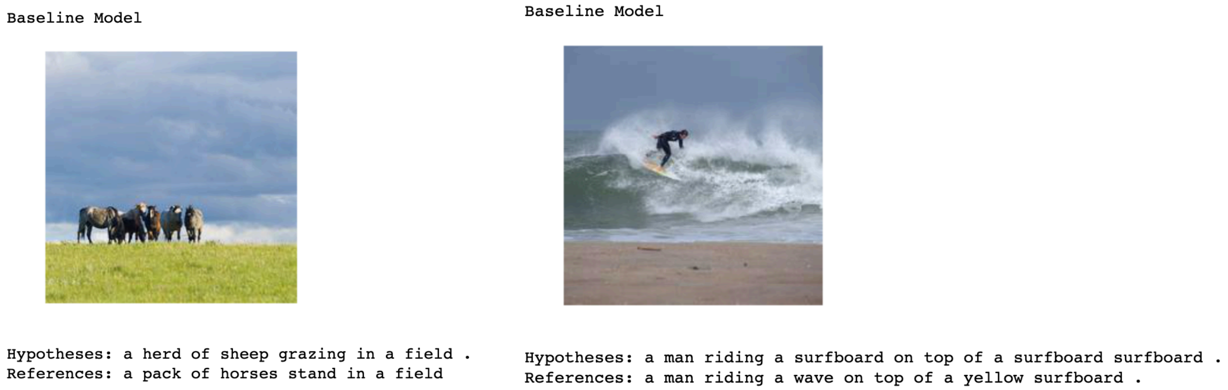
Hyper-Parameter	Value	Justification (if needed)
Gradient Clip	5	To avoid gradient explosion.
Epochs	4	Limited to 4 epochs due to GPU accessibility.
Batch Size	32	as per Author
Decoder Learning Rate	0.0004	as per Author
Dropout Rate	0.5	as per Author
Vocabulary Size	8856	For the MS COCO 2014 Dataset
Encoder Dimension	2,048	Based on RESNET-101's output size
Attention Dimension	512	as per Author
Weights Initialization	in range of [-0.1, 0.1]	In order to keep a uniform distribution

Table 2: Experiment Hyper-Parameters

validated on the same dataset splits with the same vocabulary to enable an accurate performance comparison. Each epoch in the baseline and GloVe model took around 3.5 hours to train on an NVDIA GTX 1070 Ti GPU, while the BERT model's epoch took around 4.2 hours.

5.1 Baseline Attention Model

Figure 1 shows sample results obtained from the baseline model. Qualitatively analyzing the results, it can be seen that the image on the left has an accurate, grammatically correct hypotheses although the animal classifications are incorrect. Additionally, the model used the word"herd" where is should have been "flock" due to it's limited vocabulary. The image on the right shows a hypothesis that is more similar to the average hypotheses generated by the baseline model where many word repetitions occur. This means that the model correctly learned some representations, but didn't finish training yet due to GPU restrictions.



Baseline Model

```
Hypotheses: a herd of sheep grazing in a field .
References: a pack of horses stand in a field
```

Baseline Model

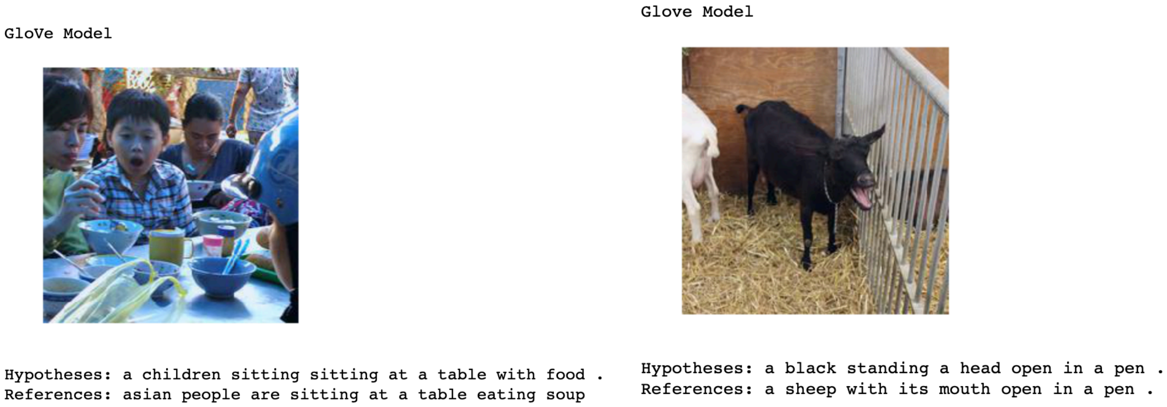
```
Hypotheses: a man riding a surfboard on top of a surfboard surfboard .
References: a man riding a wave on top of a yellow surfboard .
```

Figure 1: (left) successful hypothesis from the baseline model. (right) incorrect hypothesis from the baseline model

Additionally, this model is unable to generate sentences that have the same meaning as the reference sentences while using synonymous words, it seemed that the model is attempting to copy the reference sentence word by word. This can be explained by the fact that no pre-trained embeddings were used in this model. Therefore, it was difficult for the model to learn accurate word representations that would allow it to switch similar words.

5.2 GloVe Attention Model

Although the Glove model has similar quantitative results to the baseline model in validation loss and BLEU scores, the GloVe model proved to be able to generate captions that use a different style of writing than the reference captions by using different words that are synonymous in meaning. This change can be explained by the fact that GloVe embeddings offer the model the ability to pick and choose the best possible word from a cluster of similar words. The left image in Figure 2 shows can example where the words 'Asian people' was translated to 'children' in the generated caption. However, this model has similar repetition problems as the baseline model due to limited training.



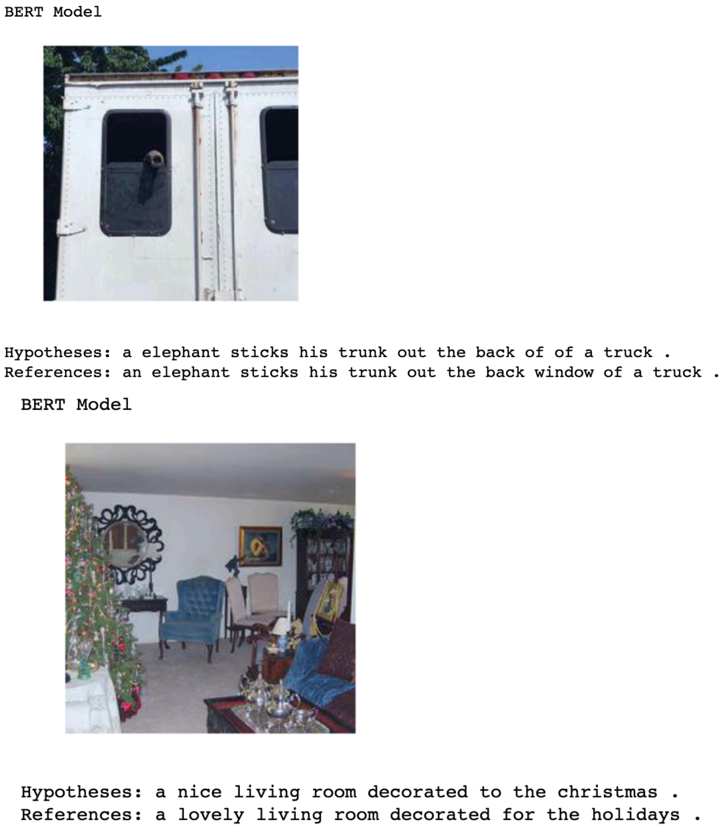
```
Hypotheses: a children sitting sitting at a table with food .
References: asian people are sitting at a table eating soup
```

```
Hypotheses: a black standing a head open in a pen .
References: a sheep with its mouth open in a pen .
```

Figure 2: (left) decent hypothesis from the GloVe model. (right) incorrect hypothesis from the GloVe model

5.3 BERT Attention Model

Although it was expected that the BERT model would outperform both the baseline and GloVe models because of its reliance on contextualized word embeddings, it was surprising to see the extent of increase in BLEU score it could achieve. The validation loss (Cross Entropy) decreased much faster in the BERT model, which clearly shows that the BERT embeddings are very accurate in representing contextualized words. Additionally, this shows that context is very important in generating image captions, which makes a lot of sense because captions are supposed to relate objects in an image together and make sense of them.



```
BERT Model

Hypotheses: a elephant sticks his trunk out the back of of a truck .
References: an elephant sticks his trunk out the back window of a truck .
```

```
BERT Model

Hypotheses: a nice living room decorated to the christmas .
References: a lovely living room decorated for the holidays .
```

Figure 3: Accurate captions by the BERT model

Figure 3 shows two example of the BERT model predicting captions, both captions make sense and are grammatically correct. Interestingly, the predicted captions (hypothesis) predicted similar sentences to the reference caption but use different word variants to explain similar things. Notice that 'back window' was translated to 'back' and 'holidays' was translated to 'Christmas'. This is interesting as it shows that the model offers correct captions that are not exactly the same as the reference captions, which is essentially the goal of image captioning. Overall, BERT captions were very well written with the exception of a few hiccups with repetitions.

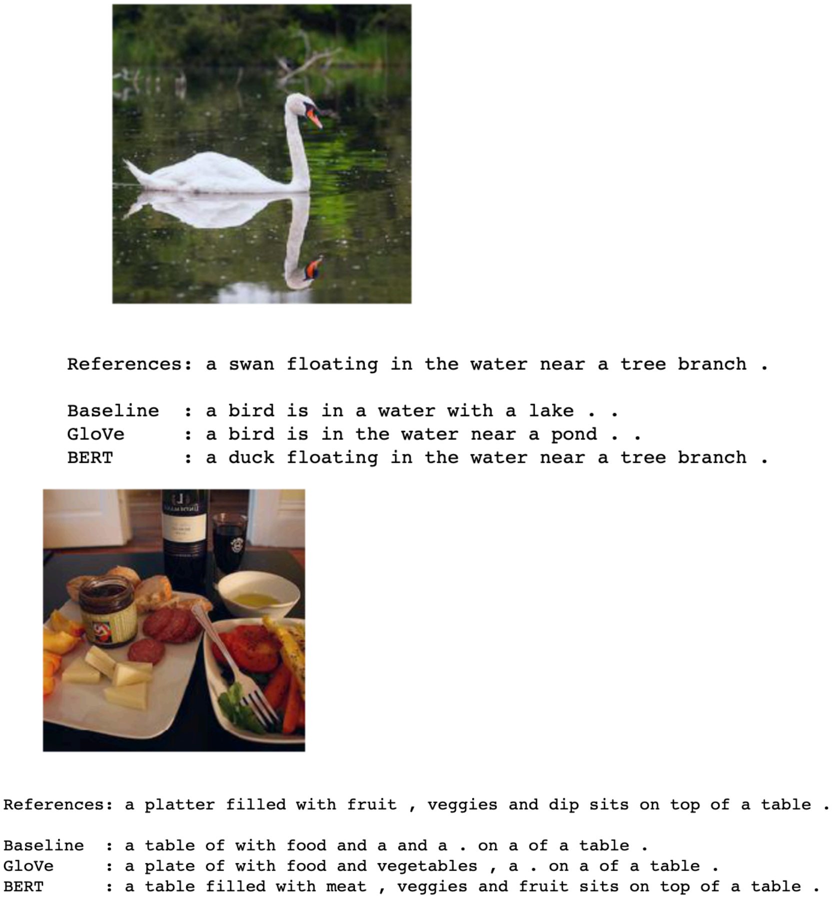
Model	Val (loss)	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Baseline Model	3.452	48.51	18.84	7.406	3.097
GloVe Model	3.325	49.70	20.07	8.214	3.552
BERT Model	**1.901**	**78.27**	**59.53**	**46.22**	**36.53**

Table 3: Model validation loss and BLEU scores on the validation dataset

5.4 Summary of findings

After running a full single epoch on the MS COCO 2014 validation set and comparing each hypothesis to 5 reference captions, Table 3 shows a full summary of the results. For performance metrics, I considered validation loss, BLEU-1, BLEU-2, BLEU-3, and BLEU-4 which give an accurate indication of how well each model performs quantitatively.

Baseline and GloVe yielded very similar results to each other, having only a slight improvement with Glove due to the introduction of pre-trained embeddings that were trained on vast amounts of data. BERT, on the other hand, had much better results in all aspects as shown in Table 3. The BERT model results outperformed the results obtained by (Xu et al., 2016) while being trained on fewer epochs (Vinodababu, 2019).

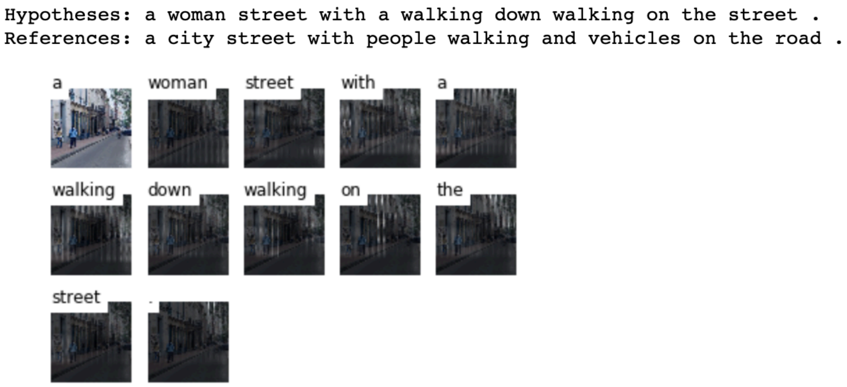


```
References: a swan floating in the water near a tree branch .

Baseline  : a bird is in a water with a lake . .
GloVe     : a bird is in the water near a pond . .
BERT      : a duck floating in the water near a tree branch .
```

```
References: a platter filled with fruit , veggies and dip sits on top of a table .

Baseline  : a table of with food and a and a . on a of a table .
GloVe     : a plate of with food and vegetables , a . on a of a table .
BERT      : a table filled with meat , veggies and fruit sits on top of a table .
```

Figure 4: Direct comparison of the three main models proposed

Figure 4 shows a direct a comparison between the three models implemented. The results clearly show that Baseline and GloVe yield similar results, while BERT outperforms them by a large margin.



```
Hypotheses: a woman street with a walking down walking on the street .
References: a city street with people walking and vehicles on the road .
```

Figure 5: Failed attempt to visualizing attention

Figure 5 shows a failed attempt of implementing a program that visualizes the attention and maps it onto the image in order to see what the model is attending to while predicting a specific word. The issue here persists because of how I map the attention values onto the image.

6 Conclusions and Future Work

This project proposes two extensions to the self-attention based approach of Neural Image Captioning (introduced in (Xu et al., 2016)) that enhanced the performance of the model and reducing training time alongside. The BERT approach surpasses the MS COCO validation scores obtained in the original publication while being trained on fewer epochs with the same hyper-parameters. Our experiments outline the importance of word embeddings in Computational Linguistics and offers a new method of integrating BERT with the already developed models to enhance their performance. The implementation is publicly available on Github[1].

It would be an exciting endeavour to see how FastText and other Skipgram-based word embedding generation approaches perform for this task. This also includes training a new model with BERT large as opposed to BERT base, and experimenting with other autoencoder variants such as RoBERTa, ALBERT (Liu et al., 2019; Lan et al., 2020) and with autoregressive language models such as XLNet, GPT-2 (Yang et al., 2019; Radford et al., 2019). Other possible extensions to this work would also include utilization of beam search in validation, and training the models until the training loss converges (unlike this case).

7 Acknowledgements

At the end, I would like to express my sincere gratitude to my Professor and Project Supervisor, Dr. Rajiv Kapoor for his enthusiasm, patience, insightful comments, practical advice, and unceasing ideas throughout the course. This project (and course) taught me countless nuances and most of all, gave me a chance to contribute to the Computer Vision Community as a whole for the novel

[1] https://github.com/ramanshgrover/A-Modern-Approach-To-Image-Captioning

task of Image Captioning. I would also like to extend this gratitude to the Rahul Sir for conducting my presentation and offering me discerning yet intuitive advice for the same.

References

Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention, 2015.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2016.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations, 2020.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context, 2015.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach, 2019.

Sunwoo Park. A pytorch implementation of show attend and tell. https://github.com/parksunwoo/show_attend_and_tell_pytorch, 2018.

Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL https://www.aclweb.org/anthology/D14-1162.

A. Radford, Jeffrey Wu, R. Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Sagar Vinodababu. A pytorch tutorial to image captioning. https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning, 2019.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator, 2015.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention, 2016.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763, 2019.