Image Captioning Capstone Project: A Two-stage method without recurrent neural network

DI WU
Johns Hopkins University
dwu49@jh.edu

Abstract: *Automatically describing an image with semantically correct sentences is a challenging problem since it requires both computer vision and natural language processing domain knowledge. The generated sentences need to cover a lot of meaningful information such as what are the objects; where are they; what are the relationships between those objects; what colors objects have; what action they made and even what are the object s' emotion. Traditional image captioning models use a seq-to-seq[1] architecture with a Convolutional Neural Network as encoder and Recurrent Neural Network as decoder which already achieved great performances on different benchmark detests. However, both CNN and RNN have some drawbacks that could potentially affect the model performance. Inspired by the most advanced transformer architecture [2][3]. We created a new two stage model that achieved 0.61 BLEU score on flickr8k dataset with only several epoches*

Keywords: Transformer, ViT, Attention, Two-stage, Beam Search, Greedy Search, BLEU Score

1 Introduction

Image captioning is a difficult and a rapidly growing AI research topic. It has been used in many areas such as video content subtitle; helping visually impaired people understand the world. A decent model should be able to generate semantically correct natural languages to describe an image like a human. Most image caption methods can be categorized as three approaches. The first two approaches are template-based caption and retrieval-based caption. Template-based caption uses a series of template slots called black slots to describe the image. The main idea is to detect objects and their relationships, then fill the information into the template description statements. However, those fixed templates are too simple to describe more details in the image. While retrieval-based caption uses a pre-defined set of image-caption pairs, and map the input image to the similar images in the set by using K-Nearst-Neighborhood Method. Then the caption will be generated based on the similar images' captions. This approach is highly dependent on a huge set and usually not produce good descriptions for specificity images.

In recent years, most image captioning models embraced encoder-decoder structure where encoder extracts meaningful high level features from the images and decoder received the features and combined them to learn how to generate semantic nature lan-

guages. The most common encoder is Convolutional Neural Network due to its inductive bias (locality, shift invariant). Most of the researchers tried to use CNN architecture such as InceptionV3, Resnet, VGG, faster RCNN to extract features. And they use Recurrent Neural Network such as LSTM, GRU as decoder. The CNN-RNN based method has achieved significant improvement on Flickr8k, Flickr30k and MS COCO dataset. However, since RNN-based decoder generates words sequentially with auto-regression. The computational time would be slow. Moreover, although CNN would be benefit from convolutional layer, the capacity of learning long-range spatial dependencies is limited because CNN usually scans part of the image.

With the popularity of transformers [3] in the field of vision and language, we found that transformers are also helpful for multi-modal tasks like image captioning. First, transformer makes the parallel computing possible by masking the current caption inputs. With self-attention mechanism, every sequence word will be connected with all other words include the word itself, which makes the long-range dependencies possible in the network. Vision transformer applied the idea into image area where it patches an image into a 16x16 block and flatten the image to act like a sequence in natural language tasks. However, transformer architecture lacks of inductive bias inherent from traditional Convolutional Neural Network which might neglect low level feature representation especially on small datasets.

In order to strength the learning of low-level features, we proposed a new two-stage model, which took advantage of image augmentation, using a Vision Transformer and a simple Convolutional Neural Network with a few layers to extract both high-level and low-level features from an image. The result will be merged through a weighted average and put them into a transformer decoder. We use beam search = 5 to generate the captions. Our model achieved 0.61 BLEU score on the Flickr8k dataset.

The remaining part of the paper will be structured as follow. Section 2 will introduce some related works similar to our methods. The detailed model information will be elaborate on Section 3. We experiment four different network structures (CNN-GRU, CNN-Attentioned GRU, CNN-Transformer, ViT-Transformer) in Section 4, the experimental results and ablation study will be included there. In section 5, we draw the conclusion along with potential future works. Section 6 includes the references and acknowledgements. Other related materials will be put in Appendix on section 7.

2 Related Work

In 2014, Orirol Vinyals [4] and his team created a NIC (Neural Image Caption) model inspired by machine translation task [1]. The model took a pre-trained CNN network as an image encoder and used the last layer as an input of the LSTM decoder. Compared to other RNN architecture, they used LSTM to deal with gradient exploding and gradient vanishing. In 2015, Kelvin Xu [5] lead a different team in Canada extended Oriorl's idea by adding attention mechanism [6] to the decoder part. They proposed two attention-based caption generator: soft attention and hard attention. The soft attention used additive attention to find the most relevant features given a caption word. This can be trained through a back-propagation while a hard attention was trained by maximizing lower bound or using reinforcement learning. In [7], Peter Anderson team made

further extension to the attention mechanism and achieved a new state-of-art for MS COCO dataset. They used two attentions (task specific context top-down attention, visual feed-forward bottom-up attention) to selectively extract image local features. They also changed the traditional CNN network to the object detection network F-CNN to increase the model accuracy by filtering out useless image features.

With the remarkable success on Transformer [3] seq-to-seq structure. Lun Huang and his team created a CNN-Transformer based model AoA (Attention on Attention). The multi-head attention worked on both encoder and decoder where encoder first extracted a set of features with a CNN network and then put into a transformer encoder. Both Simao Herdade [9] and Sen He [10] utilized the similar idea like Lun but they used object detection F-RCNN as feature extractor. Simao had changed the vanilla transformer into a Object Relation Transformer to investigate the relative geometry between different bounding boxes, while Sen first used the LSTM to generate a caption token then put them into the transformer.

Most of the above methods used traditional CNN or Object Detection F-CNN as a backbone for image captioning encoder. They either directly put the CNN extracted features into transformer decoder or put them into transformer encoder first then transformer decoder. Inspired by Vision Transformer [2], Wei Liu [11] created a full transformer architecture for image captioning without any CNN and RNN. Their work was simply combining the Vision Transformer with the vanilla transformer decoder. For the encoder part, they gave up the special $[CLS]$ token and send the concatenate patch and positional embedding to the transform encoding layer. Wei claimed that using transformer will be more effective since CNN has the limitation to catch long-range spatial dependencies. Yiyu Wang [12] also created an end-to-end transformer-based image captioning. They used more advanced and powerful vision transformer (Swin Transformer) [13] to replace F-RCNN as a backbone encoder network. Moreover, they also built a refining encoder and decoder to capture intra-relationship in order to increase the interaction between image and word captions. They also calculated the mean pooling for the grid features to increase the model capability. Their model achieved a new state-of-art on MC COCO dataset.

Reinforcement learning had also be implemented on the image captioning task. Steve J Rennie and his team proposed a new idea "self-critical learning" [14]. They treated sequence generator as a reinforcement problem to minimize the negative expected reward where a reward is a CIDEr [15] score of the generated captions compared with the ground truth captions.

3 Model

We have presented an overview of the proposed model architecture in Fig [1]. Before we jumped into the model, first we reshaped the image into a fix resolution image $\in \mathbb{R}^{HxWx3}$ in RGB chanels. Then we transformed the image into a Pytorch tensor (The whole project is built on Pytorch framework) and performed a normalization to all dimension where the dimension had mean = [0.485, 0.456, 0.406] and std = 0.229, 0.224, 0.225]. The ground truth caption will be padding with a special token $< PAD >$ to make sure every caption will have the same length in a batch. The model contains three parts:

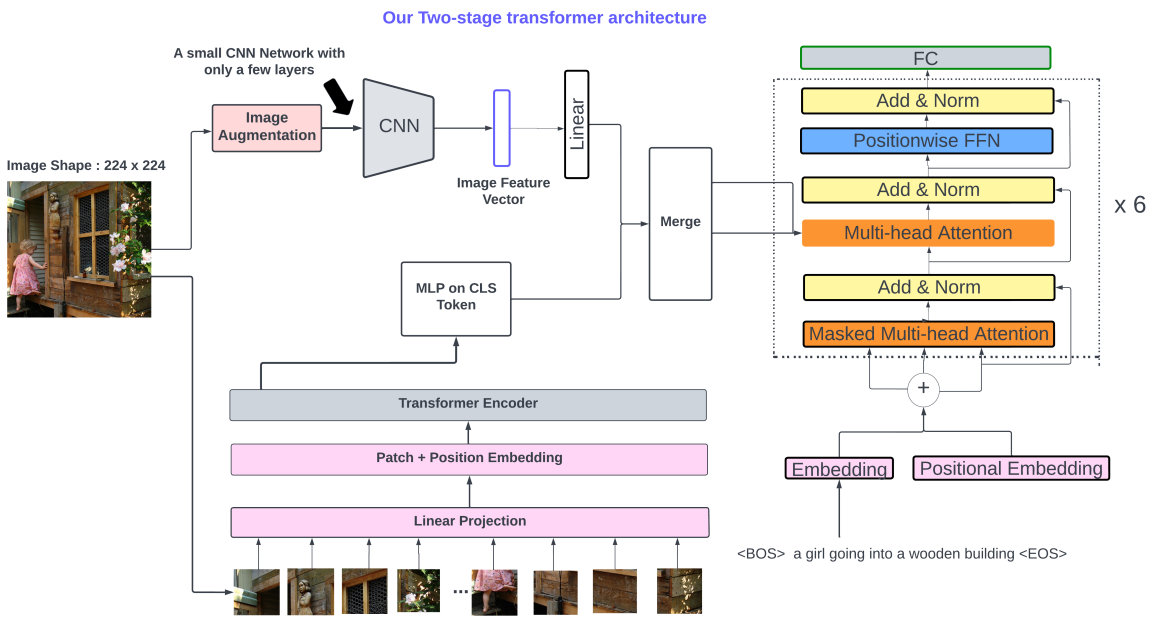

Figure 1: The overall our proposed model. It contains a pretrained CNN with only a few shallow layers and a ViT transformer as encoder, a 6 layer transformer decoder with learnable positional embedding and additive attention

1. Image augmentation: includes random vertical and horizontal flip, Gaussian noise and Gaussian Blur.
2. A encoder contains both a simple and small pretrained Convolutional Neural Network and a pre-trained ViT, the result will be merged and send into the decoder
3. A transformer decoder with additive attention instead of dot-product attention

Attention Mechanism

Starting from 1964, people began to form a idea that mimicking humans' sensory attention using an attention mechanism. Nadaraya and Watson proposed a non-parameter techniques to estimate the conditional expecation of a random variable. $f(x) = \sum_{i=1}^{n} \frac{K(x-x_i)}{\sum_{j=1}^{n} K(x-x_j)} y_i$ Inspired by their ideas, a general attention expression were used:

$$f(\mathbf{q}, (\mathbf{k}_1, \mathbf{v}_1), \ldots, (\mathbf{k}_m, \mathbf{v}_m)) = \sum_{i=1}^{m} \alpha(\mathbf{q}, \mathbf{k}_i)\mathbf{v}_i \tag{1}$$

$$\alpha(\mathbf{q}, \mathbf{k}_i) = \text{softmax}(a(\mathbf{q}, \mathbf{k}_i)). \tag{2}$$

where $q \in \mathbb{R}^q$, $k \in \mathbb{R}^k$, $v \in \mathbb{R}^v$, and $a()$ is attention scoring function. It contained three important parts: query and key-value pair. In our image captioning task, each word acts as a query, the image extracted features will be used as a key-value pair where key is used for calculating attention distribution and value is used to generate selected features. Unlike vanilla transformer from the paper attention is all you need [3], our model using additive attention rather than dot-product attention.

$$a(\mathbf{q}, \mathbf{k}) = \mathbf{w}_v^\top \tanh(\mathbf{W}_q\mathbf{q} + \mathbf{W}_k\mathbf{k}) \in \mathbb{R}, \tag{3}$$

In transformer layer, we used multi-head attention which contains h parallel attention layers(heads).

$$MHA(\mathbf{q}, \mathbf{k}) = [h_1, ..., h_h]W_o \in \mathbb{R}^{p_o}, \tag{4}$$

where h_i is the additive attention defined above equation (3).

3.1 Encoder

The encoder contains two parts: One is a Vision Transformer network. The image $X \in \mathbb{R}^{H \times W \times C}$ had been divided into sequence of flattened N patches image where each patch has the shape $\mathbb{R}^{N(P^2C)}$ and $N = (H/P) * (W/P)$. The whole patches went through a linear embedding. An special token $< CLS >$ will be added to the patch embedding along with a learnable position embedding. Then the whole embedding will be throwing to the transformer encoder layer. The special token $< CLS >$ will be acted as a image representation inspired from BERT [17] because self-attention has attention information to all other embedded tokens. However, since attention mechanism has the property: same dimension in same dimension out. We can also NOT use the special token. Instead, we can perform a global average to the whole output or use the entire output without CLS token. Our experiments showed that there aren't significant different between the three encoding method for ViT part.

Another one is a simple and small Convolutional Neural Network. This could use a lot of frame work such as VGG, ResNet, AlexNet, Inception V3, etc with all deeper layer removed. So the small network only contains a few shallow layers. The reason is that we found ViT is pretty good at extracting high level features but could potential neglect some low level features (color, texture). We notice that by simply use ViT as an encoder, many generated captions would have wrong color descriptions. Since traditional Convolutional Nerual Network did great jobs on capturing low-level features on the image at the first couple layers. So We added a small network to extract the color information accurately. Then we merge the Convolutional Neural Network result and Vision Transformer result together to feed into decoder layer. In our model, we use a pretrained Resnet50 [1] and only kept the first layer of the network. Notice that we also found that using a non-pretrained Resnet50 model with all layers training from the start would have the similar results. Other than simply merge the result, we found that using weighted average sum would also achieve great performance.

3.2 Decoder

The decoder part will be used for both training and inference (image caption generator). First, two special token $< BOS >$ and $< EOS >$ will be put on the begin and end of the ground truth captions as "Begin of Sequence" and "End of Sequence". We use sinusoid positional embedding along with the word embedding to embed the ground truth captions. Similarly, we also tried a learnable positional embedding, which doesn't have significant different. The embedded captions will be processed by a masked multi-head attention with all query, key and value the same. The reason why we used mask was because we wanted to feed the entire captions into the model for efficiency. However, the

[1] For both pretrained model, we freeze all but the last layer of the models for fine tuning use.

whole process still need to be auto-regressive $P(X_i|X_{j<i};\theta)$. Therefore, we don't want our model to look ahead when compute $P(X_i|X_{j<i};\theta)$.

After masked multi-head attention, the output will take a layer norm and skip-connection which is quite the same as discussed in ResNet [18]. Then the output along with the encoder output will be thrown into a vanilla transformer layer. The last layer of the output of the transformer will be used to predict the next words by a fully connection layer that returns a dimension equal to vocabulary size.

3.3 Loss Function

For this image captioning task, we tried to minimize the cross-entropy loss:

$$L_{XE} = -\sum_{t=1}^{T} log(p_\theta(y_t^*|y_{1:t-1}^*)) \tag{5}$$

where $y_{1:T}^*$ is the ground truth caption and y_t^* is the prediction of the model. Since we add the special token $< PAD >$ to the captions in order to make them in the same length for the model. The cross-entropy loss will ignore the $< PAD >$ index.

4 Experiment

In this image caption task, we used Flickr 8k dataset. This dataset contains 8091 images and each image has 5 ground truth captions. Therefore, the total number of image-caption pair is 40,455. We use 6,091 images as our training dataset, 1,000 iamges as validation dataset and 1000 for testing dataset. The vocabulary is created with additional sepeical tokens included: $< BOS >$, $< EOS >$, $< PAD >$ and $< UNK >$ indicating: begin of sequence, end of sequence, padding and unknown. We dropped the words with less than or equal to 1 frequency. The batch size for the training data and validation data was 64 and 32 respectively. We used BLEU [19] score to evaluate our model performance. The BLEU score is defined as follow:

$$\exp\left(\min\left(0, 1 - \frac{\text{len}_{\text{label}}}{\text{len}_{\text{pred}}}\right)\right) \prod_{n=1}^{k} p_n^{1/2^n} \tag{6}$$

where $\text{len}_{\text{label}}$ is the number of tokens in the ground truth caption and len_{pred} is the number of token in the predicted caption. The p_n is the ratio of [number of matched n-gram in the predicted caption] and [number of n-gram in ground truth caption]. For example, if the predict caption is [ABBC] and the ground truth caption is [ABCD] then we have $p_1 = \frac{3}{4}$, $p_2 = \frac{2}{3}$, $p_3 = 0$, $p_4 = 0$. In our task, we used BLEU score with 1-gram to 4-gram.

We trained our model with a pretrained ViT and a pretrained ResNet50. The output dimension was set to be 256. For pretrained ResNet50 model, we did the same thing as for Vision Tranaformer, however, we also set the sequential layer and layer 2, 3, 4 to be identical which is same as removing these layers. We only wanted the first couple layers to learn the low-level features for color representation. For decoder transformer, we used additive attention with 8 heads. The ground truth caption is embedded into dimension

256, and the number of transformer layer is 6. The feed-forward dimension first go up to 2048 then get back to 256. The entire work use dropout value = 0.5.

We used Adam [20] as our optimizer. The whole model was first trained with a cross-entropy loss for 15 epoches. The initial learning rate was 0.0003 where we used a scheduler to decrease the learning rate by 0.1 for every 5 epochs. Beam search = 5 was used in the task to increase the BLEU score.

4.1 Model Comparison

Table 1: Compare BLEU score for different models

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
CNN+GRU	47.19	23.88	11.76	6.15
CNN+Attention GRU	48.41	23.97	11.54	6.34
CNN+Transformer	55.65	31.89	18.97	10.97
ViT+Transformer	58.41	35.95	22.40	13.91
ViT+GPT-2	59.88	41.06	26.79	17.12
Two Stage Transformer	61.55	40.40	27.00	18.28

We created 5 different models and each model was created upon the previous one for better BLEU score. The first model was a pretrained CNN+GRU. For this model, the output of the encoder was both added to the input of GRU and also treated as a hidden state of the GRU. Notice that we could replace GRU [21] with LSTM. However, since LSTM needs additional state "cell state", we could also replicate the output of encoder to be both hidden state and cell state for LSTM. The second model was CNN + Attention GRU where we included attention [5] mechanism to the decoder so that the caption knows where to look from the image. We tried both additive attention and dot-product attention for the second model. For the third model, we replaced GRU to a fully transformer decoder so that our model would gain the parallel capacity which increase the efficiency. [2] For the fourth model, we built a ViT + Transformer (a fully transformer encoder-decoder) model which made the model learn more about long-range spatial dependencies. Then we compared the whole pretrained ViT + GPT-2 model from the Hugging Face to see the current state-of-art model performance. The last model was our proposed model.

The comparison for the above 6 models was showed on Table 1. Our model had better performance than all other models in BLEU-1,3,4 score, and also pretty closed to the pretrained model from Hugging Face in BLEU-2.

4.2 Ablation Study

In this project, we conducted a lot of ablation studies to make our model more robust. The three main directions are: Data Augmentation, Encoder models, Decoder architectures and hyper-parameters.

[2]Start from model three, we used additive attention to replace dot-product attention.

4.2.1 Data Augmentation

We have tried 4 different data augmentations on image part including random image vertical flip, random image horizontal flip, Gaussian noise and Gaussian blur and many combinations among the four image augmentation method. We also tried to first resize the image into [232, 232] then random crop the image in shape [224, 224] for augmentation. Beside random crop, we tried to use a convolution layer with kernel size = 9 to reduce the size into [224,224]. We Gaussian blur method had better performance than others. We also Notice that using random crop to augment image will confuse the model and generate weird captions since random crop might miss some useful information.

Table 2: Compare different augmentation method

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Without Augmentation	58.41	35.95	22.40	13.91
Random Vertical Flip	60.28	39.32	25.44	16.71
Random Horizontal Flip	60.04	38.69	24.97	16.08
Random Flip + Gaussian Blur	58.41	37.16	23.90	15.34
Random Crop	44.48	21.86	11.01	6.00
Excat same input	60.06	39.07	25.42	16.21
Gaussian Noise	59.07	37.96	24.42	15.54
Gaussian Blur	61.55	40.40	27.00	18.28

Different from choosing various data augmentation methods, we also considered where to put the data augmentation. Instead of putting it into the encoder, we also tried to put them into the data-loader as we pre-processed the image data. Therefore, the image will have a direct two processing: a small and simple CNN for adding inductive bias and focusing on low-level feature representation and a ViT for learning long-range spatial dependencies. During inference, we will not use data augmentation part. However, the result for this approach was not good enough.

4.2.2 Encoder model

Table 3: Compare different encoder methods

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Pretrained YOLO-V5	58.50	37.51	25.02	16.70
Pretrained F-RCNN	59.29	37.39	23.77	15.13
Non-pretrained Resnet	61.24	39.61	25.53	16.33
Pretrained Resnet	61.55	40.40	27.00	18.28
ViT without special token	61.04	39.60	26.25	17.41

For encoder part, we also tried to use some object detection methods as a backbone [24] instead of Vision Transformer or Convolutional Neural Network. We tried to take advantage of F-RCNN [25] and YOLO [26]. However, both of them had some drawbacks. For example, F-RCNN was super slow even in our small flickr8k dataset, but the model performance was pretty good. In order to train F-RCNN better, we might need large

amount of time which was beyond our budgets of computational resources. YOLO was really fast but the model performance was not quite good. Moreover, most of the object detection methods returned a list of bounding boxes location along with their weight, height, confidence and classes information. They usually generated a lot of messy bounding boxes. In order to clean over the bounding boxes, we have to use non-max suppression method. However, this data processing will lead to inconsistent tensor shape that need further process such as repeat dimensions or cut dimensions.

In order to let our model learn more accurately on the low-level color features. We also tried to use a non-pretrained model and start to train from the start with only a couple of epochs instead of using our pretrained model that eliminated the deeper layers. We found that both of them would give us the pretty close results. As for high-level and low-level feature merging, we also tried to use weighted average, simply addition and concatenation on specific dimensions. We found that addition and concatenation will generate similar results. For weighted average, The high-level feature map needed higher weight in order to perform better.

For pretrained vision transformer, we tried not to use the special $< CLS >$ token as the image representation. Instead, we used the whole linear embedded sequence as the input to the transformer decoder. The result didn't have significant difference.

4.2.3 Decoder architecture and hyper-parameters

Table 4: Compare hyper-parameters in decoder

Src embed size	Trg embed size	forward expansion	LR	dropout	BLEU-1
512	256	2048	2e-4	0.1	55.57
512	512	2048	3e-4	0.1	59.07
512	256	2048	1e-4	0.5	59.86
512	256	1024	3e-4	0.5	61.04
512	256	2048	3e-4	0.5	61.10
256	**256**	**2048**	3e-4	0.5	61.55

For decoder part, we tried different combination of the hyper-parameters. The target and source embed size we used is [256, 512], forward expansion = [1024, 2048], dropout = [0.1, 0.5]. We found that small embedded size would gave us better model performance. We also replaced the dot-product attention with an additive attention since additive attention have learnable weights so that the model will learn faster with only a few epochs.

5 Conclusions

In this project, we have built 5 different models including CNN+RNN, CNN+Attention RNN, CNN + Transformer, ViT+Transformer and our new two-stage transformer model. Each of the models was built upon the previous one as we were adding more and more complex mechanisms to improve the evaluation metrics (BLEU Score). As the result shows, the two-stage model was outperformed than all other models because:

1. Used transformer on both encoder and decoder side as a parallel structure to process the text data efficiently instead of doing it sequentially like RNN

2. Used self-attention mechanism which increased the capacity of learning long-range spatial dependencies

3. Since the transformer lacks the inductive bias inherent from traditional Convolutional Neural Network such as locality and shift invariant, especially on small datasets. Therefore, we used a simple and small CNN with a few layers to add the inductive bias and focus more on the low-level features.

We notice that our model still has a lot of improvements, due to the limitation of computational resources (GPU) and total time. I will classify them into the following possible improvement areas for the future work:

1. So far, we only use Flickr8k dataset. It contains only 8k images. Each image have 5 ground truth captions. We can try a larger dataset like MS COCO since the transformer would have significant salient improvement on a large dataset.

2. We noticed that some of the generated captions contain repeat word tokens, which could be eliminated by decreasing the weight of the word tokens that have been generated before.

3. Currently we only did data augmentation on the image part. We think it is possible to increase data on the text part. For example, using back translation that first translates the captions into another language then translates them back.

4. We noticed that recently a lot of multi-modality models have been proposed such as CLIP, ViLT which have great achievements in the image-language domain. In the future, we can try some of those techniques.

5. Reinforcement Learning is another domain area we want to have a try in the future such as self-critical learning or use reinforcement learning for hyper-parameter tuning

6 Reference and acknowledgements

I would really appreciate professor Vishal Patel from Johns Hopkins University to support us this project. I also want to thanks professor Carey Priebe who approved my project statements and gave us a lot of great advice.

References

[1] Ilya Sutskever, Oriol Vinyals and Quoc V. Le. Sequence to Sequence Learning with Neural Networks, 2014; arXiv:1409.3215.

[2] 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, 2020; arXiv:2010.11929.

[3] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. Attention Is All You Need, 2017; arXiv:1706.03762.

[4] Oriol Vinyals, Alexander Toshev, Samy Bengio and Dumitru Erhan. Show and Tell: A Neural Image Caption Generator, 2014; arXiv:1411.4555.

[5] 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, 2015; arXiv:1502.03044.

[6] Dzmitry Bahdanau, Kyunghyun Cho and Yoshua Bengio. Neural Machine Translation by Jointly Learning to Align and Translate, 2014;

[7] 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, 2017;

[8] Lun Huang, Wenmin Wang, Jie Chen and Xiao-Yong Wei. Attention on Attention for Image Captioning, 2019; arXiv:1908.06954.

[9] Simao Herdade, Armin Kappeler, Kofi Boakye and Joao Soares. Image Captioning: Transforming Objects into Words, 2019; arXiv:1906.05963.

[10] Sen He, Wentong Liao, Hamed R. Tavakoli, Michael Yang, Bodo Rosenhahn and Nicolas Pugeault. Image Captioning through Image Transformer, 2020; arXiv:2004.14231.

[11] Wei Liu, Sihan Chen, Longteng Guo, Xinxin Zhu and Jing Liu. CPTR: Full Transformer Network for Image Captioning, 2021; arXiv:2101.10804.

[12] Yiyu Wang, Jungang Xu and Yingfei Sun. End-to-End Transformer Based Model for Image Captioning, 2022; arXiv:2203.15350.

[13] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, 2021; arXiv:2103.14030.

[14] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jarret Ross and Vaibhava Goel. Self-critical Sequence Training for Image Captioning, 2016; arXiv:1612.00563.

[15] Ramakrishna Vedantam, C. Lawrence Zitnick and Devi Parikh. CIDEr: Consensus-based Image Description Evaluation, 2014; arXiv:1411.5726.

[16] Shashank Bujimalla, Mahesh Subedar and Omesh Tickoo. Data augmentation to improve robustness of image captioning solutions, 2021; arXiv:2106.05437.

[17] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018; arXiv:1810.04805.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition, 2015; arXiv:1512.03385.

[19] Papineni, Kishore and Roukos, Salim and Ward, Todd and Zhu, Wei-Jing. a method for automatic evaluation of machine translation 2002

[20] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, 2014; arXiv:1412.6980.

[21] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho and Yoshua Bengio. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling, 2014; arXiv:1412.3555.

[22] Ron Mokady, Amir Hertz and Amit H. Bermano. ClipCap: CLIP Prefix for Image Captioning, 2021; arXiv:2111.09734.

[23] Wonjae Kim, Bokyung Son and Ildoo Kim. ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision, 2021; arXiv:2102.03334.

[24] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi and Jianfeng Gao. Oscar: Object-Semantics Aligned Pre-training for Vision-Language Tasks, 2020; arXiv:2004.06165.

[25] Ross Girshick. Fast R-CNN, 2015; arXiv:1504.08083.

[26] Xingkui Zhu, Shuchang Lyu, Xu Wang and Qi Zhao. TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios, 2021; arXiv:2108.11539.

[27] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin and Baining Guo. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows, 2021; arXiv:2103.14030.

7 Appendix

7.1 Other Model Architecture

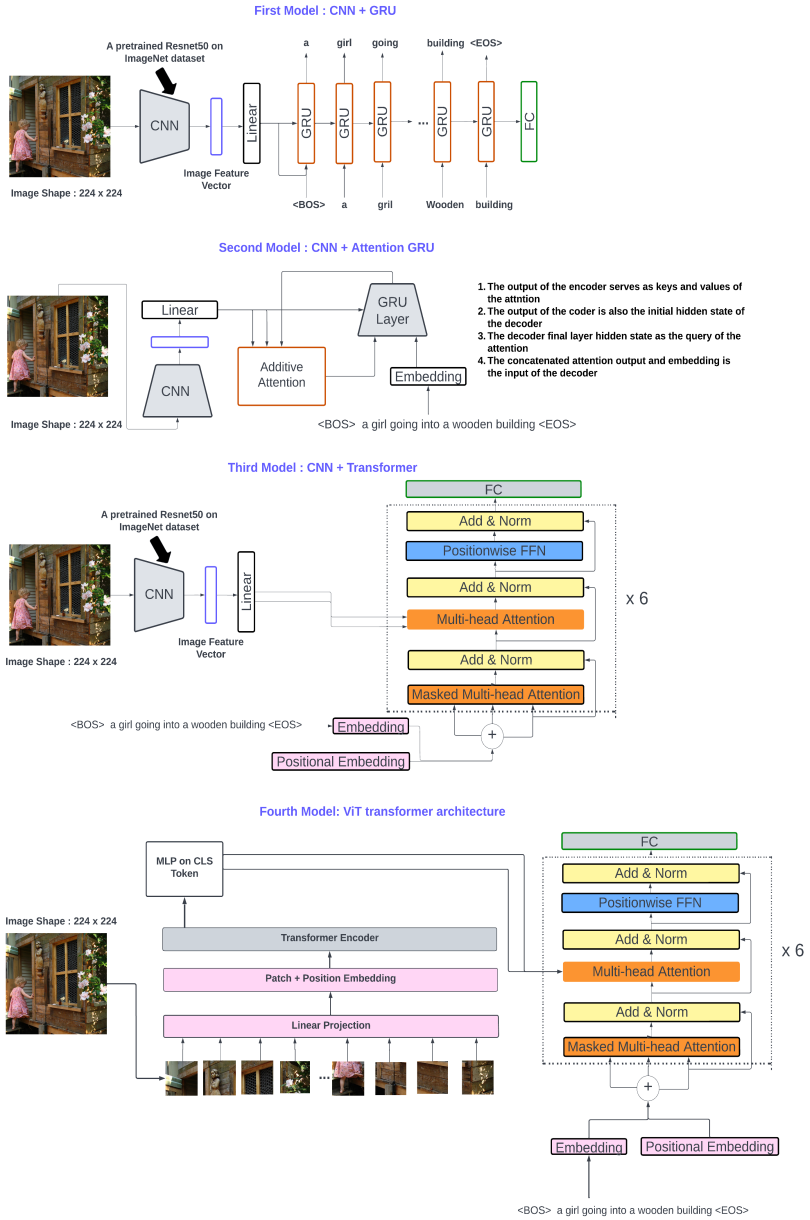

Figure 2: Four other models we created before: CNN + GRU, CNN + Attention GRU, CNN + Transformer, ViT + Transformer

7.2 Beam Search

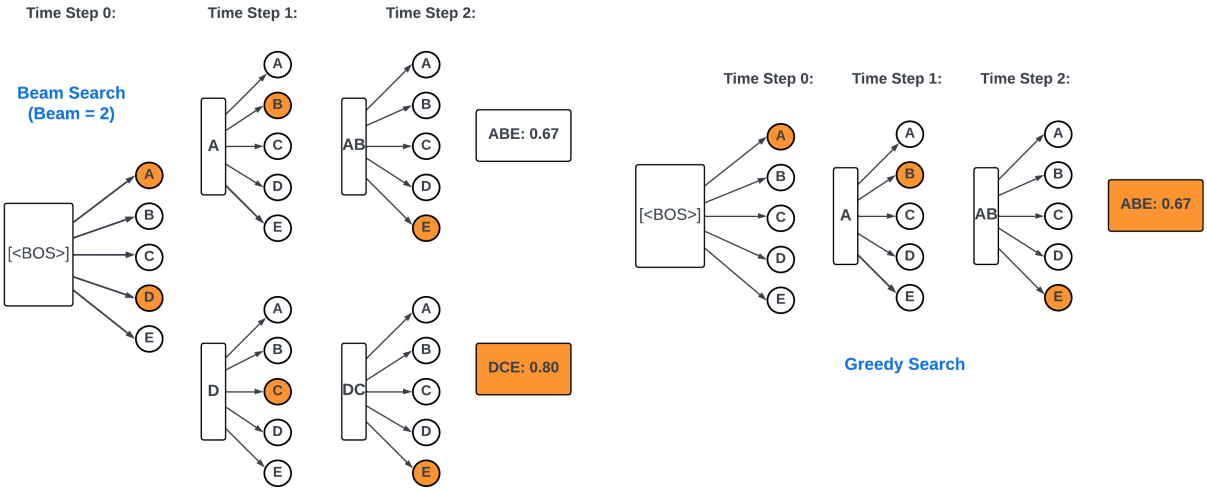

Figure 3: Beam Search example, where beam = 2

Greedy Search: In each timestep, select token with highest conditional probability
Beam Search: In each timestep, select k token with highest conditional probability

7.3 Result Examples

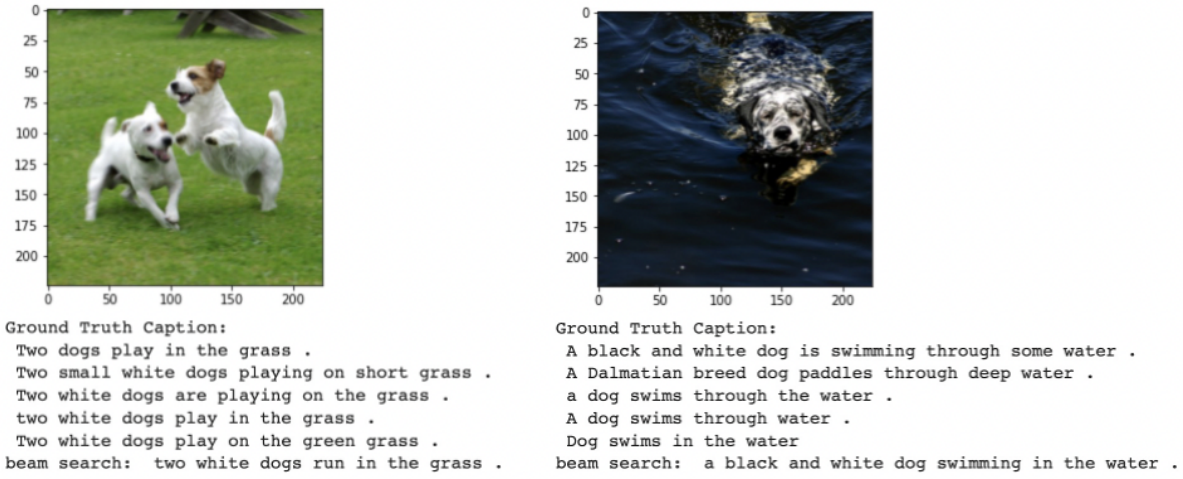

Figure 4: Example of Results