IMAGE CAPTIONING WITH ATTENTION

Group 21: Saurabh Mirani[⋆]*, Eunji Song*[⋆]*, and Shiladitya Biswas*[⋆]*,*

[⋆]University of California San Diego, La Jolla, CA 92093-0238

ABSTRACT

Image captioning is the fundamental task of scene recognition and understanding, it is achieved by combining two vast fields of Machine Learning namely Computer Vision and Natural Language Processing. In this project we are interested in studying the performance of several image captioning encoder-decoder architectures by using different encoder networks and comparing the BLEU scores of the captions generated by the architecture. Furthermore, we have also used an attention based approach to focus on particular sections of the image and accordingly describe it. The MSCOCO dataset was used to train and test our network.

1. INTRODUCTION

A generic image captioning architecture consists of an encoder-decoder pair where the encoder network encodes the input image into an encoding vector which is then used by the decoder network to generate meaningful captions related to the image. Textual Image descriptions helps in compressing huge amount of image data into simple sentences. This enables us to use Image captioning for various important applications. Firstly, we can use it to help the visually impaired, by combining Image captioning with text to speech technology. Secondly, we can use the generated scene descriptions to help self-driving vehicles to properly understand the environment, and plan the optimal and safe path accordingly. Another important application is in smart surveillance with CCTV cameras. Scene descriptions can be used to raise alarms in case malicious activities are detected.

The input to our system is an RGB image of any size. The image is pre-processed by resizing it to 224x224x3 and then normalized about a mean and variance for each colour channel. Upon passing the pre-processed image through the encoder-decoder network we get a caption describing the contents of the image.

2. RELATED WORK

Before Deep Learning image captioning was done by two methods i.e. Retrieval based and Template based method. Retrieval based method was used by Ordenez et al.([1]), Hodosh et al.([2]), Kuznetsova et al.([3], [4]) where image comparison was done to find the right caption. The closest image portions among the training images with captions were searched and rearranged to get meaningful captions. In template based approach([5], [6], [7], [8], [9]), detectors are used to detect objects, then traditional natural language modeling tools like Conditional Random Field ([10]) was used to predict the best caption for the image.

Although, these methods gave reasonable results,they wern't very accurate, since the retrieval-based model used the pre-existing caption and the template based models were generally bound by the templates among the training data.

The first use of deep learning for image captioning was proposed by Kiros et al.([11]) at 2014 where the encoder-decoder based architecture was introduced. Researchers mostly used pre-trained CNN to encode the image into a feature vector, which is then used with a decoder to model the caption. The decoder network part kept changing among researchers. Kiros et al.([11]) used a log-bilinear language model, Mao et al.([12]) replaced the feed-forward neural language model with an RNN model. Then, Vinyals et al.([13]) used LSTM for the decoder, and proposed the idea in the "Show and tell" paper. This paper is used as a baseline by Xu et al.([14]), in "Show, Attend and Tell" paper, that applied visual attention to captioning. In this project, we will mainly focus on Xu et al.([14]) work, since this research has contributed to the breakthrough in this field. Recently, Transformer([15]) network are used as decoder([16]) instead of RNN. Transformer model basically uses Attention for the main scheme, and it has been widely used in NLP area since it was introduced. Moreover, there is a very recent work this year that only uses Transformer even replacing the CNN encoder network ([17]) for image captioning.

3. DATASET AND FEATURES

We used the MS Common Objects in Context(COCO) [18]. We have specifically worked with the 2014 Train, Validation, Test images with the corresponding annotations file containing the image caption data. This dataset contains 83K train images, 41K validation images, 41K test images. We chose this dataset because it is a well-known, reliable, and has enough data to train, validate and test.

4. METHODS

Our model uses Encoder-Decoder model. Encoder-decoder has been used a lot in language modeling. The most popular encoder-decoder model is a sequence to sequence, or seq2seq([19]), which is widely used for machine translation or text summarization. Image captioning can also be described as a machine translation from an input image as a source to a text as a target. The only difference is that the source is not a language but an image. Our model is vastly based on [14], and we implemented using the references of [20], [21], [22], [23], [24] and [25].

4.1. Encoder

The model takes an image as input. In order to get the encoder output, i.e. the annotation vector a, we use Convolutional Neural Network (CNN). There are m vectors in a, each of dimension l: $a = \{a_1, a_2, a_3, ..., a_m\}$ where, $a_i \in \mathbb{R}^l$.

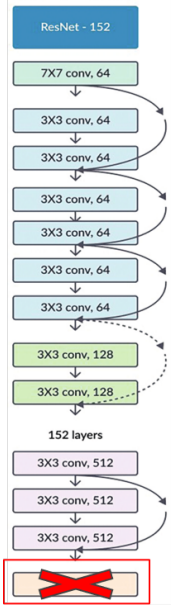
We used transfer learning technique for encoder, i.e. used pretrained models. In particular, we used pretrained ResNet-152([26]), WideResNet([27]), and ResNeXt([28]) provided by PyTorch([29], [30], [31], [32]). Since we need a 2d-feature vector to calculate the attention, we have removed the last fully connected layer of the the network as in **Fig.1a**, then feed the output to decoder with attention.

4.2. Attention

We calculated Attention based on the annotation vector from the encoder. First, for the annotation vector a, we calculate the α (weight) using Attention f_{attn} (muilti-layer perceptron): $e_{ti} = f_{att}(a_i, h_{t-1})$, $\alpha_{ti} = softmax(e_{ti})$. Here, the previous hidden state h_{t-1} is used, which means where to focus on in the step t is also related to what words have been generated so far.

Based on the calculated α_{ti}, the Attention calculates the the context vector z, which will be the input for the LSTM. In our project we used Bahdanau Attention([33])[1], which has been widely used recently, and is calculated as: $z_t = \sum_i^L \alpha_i a_i$.

Since this attention model was introduced, it has been widely used in encoder-decoder model. Without attention, the output of the encoder is just one single vector, and it is the only information that is fed to the first part of decoder. Using attention, we can also focus on certain part of the input when we generate the certain word. In our work, we focus on certain portion of the input image when generating each word. Where to focus on will be learned by the neural network using the previous word and the annotation vector. [2]



(a) Encoder([34])

Layer (type:depth-idx)	Output Shape	Param #
Decoder	--	--
├─Linear: 1-1	[-1, 512]	1,049,088
├─Tanh: 1-2	[-1, 512]	--
├─Linear: 1-3	[-1, 512]	1,049,088
├─Tanh: 1-4	[-1, 512]	--
├─Embedding: 1-5	[-1, 1, 512]	5,307,392
├─Attention: 1-6	[-1, 2048]	--
│└─Linear: 2-1	[-1, 512]	262,656
│└─Linear: 2-2	[-1, 49, 512]	1,049,088
│└─Tanh: 2-3	[-1, 49, 512]	--
│└─Linear: 2-4	[-1, 49, 1]	513
│└─Softmax: 2-5	[-1, 49]	--
├─Linear: 1-7	[-1, 2048]	1,050,624
├─Sigmoid: 1-8	[-1, 2048]	--
├─LSTMCell: 1-9	[-1, 512]	6,295,552
├─Dropout: 1-10	[-1, 512]	--
├─Linear: 1-11	[-1, 10366]	5,317,758

Total params: 21,381,759
Trainable params: 21,381,759
Non-trainable params: 0
Total mult-adds (T): 2.59
Input size (MB): 6.43
Forward/backward pass size (MB): 5.13
Params size (MB): 85.53
Estimated Total Size (MB): 97.09

(b) Decoder

Fig. 1: Overall Model

4.3. Decoder

We used Recurrent Neural Network for decoder to generate the result sentence, specifically Long Short Term Memory(LSTM) network([35]). We pass the values from Attention layer to the LSTM cell to generate each word(**Fig.1b**). RNN has been vastly used in natural language processing for text generation. It repeats the neural network word by word, so basically remembers the whole preceding words. However, as the resulting sentence grows, the model forgets the first part of the sentence. That is why the LSTM was introduced. In addition to the memory cell that RNN has, the LSTM network has an input gate(or sometimes called update gate), an output gate, and a forget gate. So that the model can learn how much it needs to forget the words. In our model, LSTM uses the z value calculated through the Attention layer previously as input, and the memory and hidden states are initialized as follows using the annotation vector from the Encoder: $c_0 = f_{init,c}(\frac{1}{L}\sum_i^L a_i)$, $h_0 = f_{init,h}(\frac{1}{L}\sum_i^L a_i)$. In addition, a method of calculating output y (the prediction) using values from LSTM is as $p(y_t|a, y_1^{t-1}) \propto exp(L_o(Ey_{t-1} + L_h h_t + L_z \hat{z}_t))$, where L's and E are learned parameters(initialized randomly).

The total trainable parameters were around 21 million and it took us about 18 hrs to train on 12GB Nvidia RTX 2080 Ti and with a batch size of 40. We used a local machine for training.

[1] our reference paper([14]) proposed two methods: stochastic("hard") attention and deterministic("soft") attention. We chose to use the soft one because it had better result on the paper

[2] respond to review: For example, in machine translation, it is natural to focus on the word "dog" when we produce the word "Hund(dog in German)". So the attention vector is set to high on the corresponding part. Then we use this attention vector in each step of the decoder together with the original input to the decoder.

5. EXPERIMENTS, RESULTS, AND DISCUSSION

5.1. Experiments

The hyper-parameters used are as follows:

- Learning rate: The initial learning rate was set to 0.0001. It is generally observed that high learning rate causes the model to converge too quickly to a sub-optimal solution, whereas a low learning rate can cause the process to get stuck. Hence, as mentioned in Xu's paper [14], we used an adaptive learning rate using the Learning_rate scheduler function readily available in PyTorch [29] library.
- Batch size: 35-40 : Since we just had 12GB of GPU memory and the train images were of high resolution, a large batch size would overflow the entire memory. Hence a small batch size in the range between 35-40 was used.
- Epochs: The whole network was trained for 10 epochs, as the BLEU scores would saturate thereafter.

5.2. Evaluation Metric

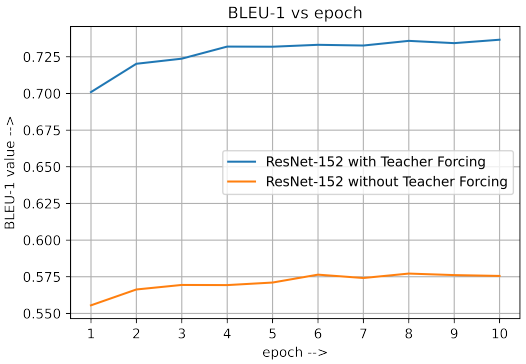
The performance of the networks were evaluated using the BiLingual Evaluation Understudy or BLEU score [36]. BLEU's output is always a number between 0 and 1. It indicates how similar the candidate text is to the reference texts, with values closer to 1 representing more similar texts. We used 4 types of BLEU scores namely, BLEU 1,2,3 & 4. The general definition of BLEU-n is a geometric average of precision over 1- to n-grams in a given reference and generated sentence. So, BLEU-1 considers single words similarity whereas BLEU-2 considers two consecutive words as well as individual words similarity. Similarly for BLEU-3 and BLEU-4.

For this project we used Python [37] as our programming language, Pycocotools library [38], COCO dataset's Python API, for handling our dataset, and PyTorch [29] to define the encoder and decoder networks and finally train them.

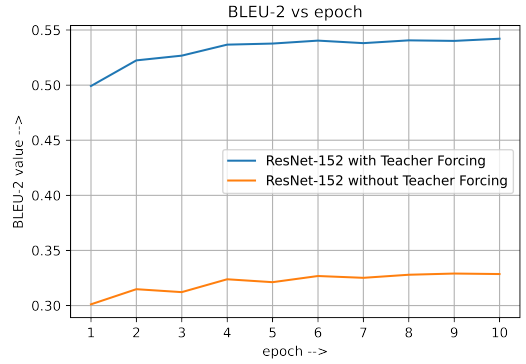
5.3. Results

Table 1 provides a comparison of BLEU-n scores between other methods and our three models. It can be seen from **Table 1** that our model with ResNet-152 as Encoder has outperformed all the other methods. Also, among our models, ResNet-152 as Encoder gives slightly better results compared to WideResNet and ResNeXt as Encoder.

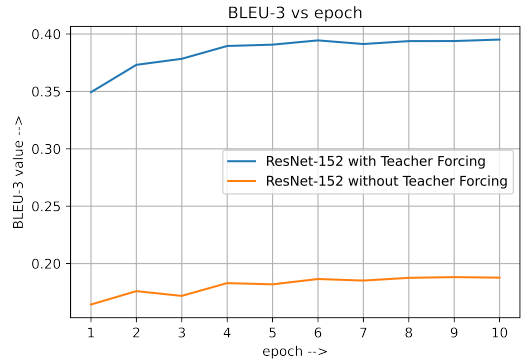
Fig. 2 provides a comparison between training using Teacher Forcing and without Teacher Forcing with ResNet-152 as Encoder. It can be seen that training using Teacher Forcing converges to a significantly higher BLEU-n score as compared to non-Teacher Forcing training.



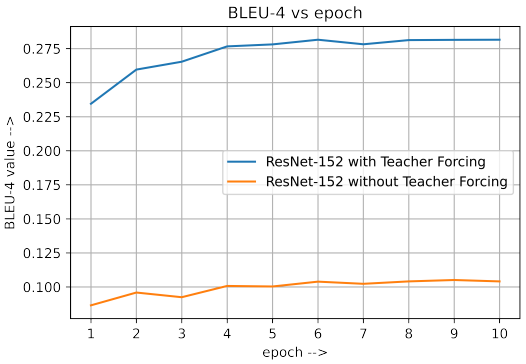
(a) BLEU-1



(b) BLEU-2

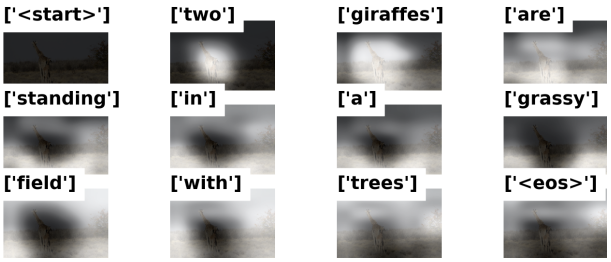


(c) BLEU-3



(d) BLEU-4

Fig. 2: Comparison of training using Teacher Forcing and without Teacher Forcing



(a) Attention to specific features

(b) Generated Caption using our model

Fig. 3: Effect of Attention

Model	BLEU-n			
	1	2	3	4
Show & Tell[13]	0.67	0.46	0.33	0.25
BRNN[39]	0.64	0.45	0.30	0.20
Show, Attend & Tell[14]	0.72	0.50	0.36	0.25
Our: WideResNet	0.73	0.53	0.38	0.27
Our: ResNeXt	0.73	**0.54**	0.39	**0.28**
Our: ResNet152	**0.74**	**0.54**	**0.40**	**0.28**

Table 1: BLEU-1,2,3 & 4 metrics compared to other methods

It can be seen from the **Figure 3** , the model focused on a specific part to generate each words on the result sentence. Attention is most useful when generating words like ”two”, “giraffe” “grassy”, ”field”, ”trees”, where the focus can be seen clearly in **Figure 3**. Instead of using the whole image at once, by focusing on the most relevant portion of the image, more accurate results are generated.

6. CONCLUSION

In this project, we implemented image captioning using encoder decoder model with attention. We found out from the experiment that the attention helped generating better caption for the images. We tried using three different types of pre-trained encoder, and model with ResNet-152 produced the best results in terms of BLEU score. This can be attributed to the fact that ResNet-152 is a deeper network compared to WideResNet and ResNext, suggesting that deeper is better in our case. In addition, using teacher forcing made the results much better. It is natural for RNNs to converge faster with teacher forcing and our results corroborates this theory.

(a) Sample 1

(b) Sample 2

Fig. 4: Sample Images([18])

[Results for Fig.3] - (a) ResNet-152, (b) ResNeXt (a) **two giraffes are standing in a grassy field with trees** (b) **two giraffe standing in a grassy field next to a tree**
an adult and baby giraffe walking through a field *both an adult and young giraffe walking in a field* *a big giraffe walks with a baby giraffe* *a tall giraffe is followed by a small giraffe in brown grass* *an adult and a baby giraffe stand gazing over a grassland*
[Results for Fig.4a] - (a) ResNet-152, (b) ResNeXt (a) **a man is eating a hot dog in his mouth** (b) **a young woman holding a hot dog in her hands**
A woman is taking a bite out of a hot dog. *A person with a red hat bites into a hot dog.* *The woman is enjoying her very large hot dog.* *A person holding a napkin and eating a hotdog.* *A young girl eats a large hot dog covered in sauerkraut, mustard, and relish.*
[Results for Fig.4b] - (a) ResNet-152, (b) ResNeXt (a) **a white plate topped with meat and vegetables** (b) **a white plate topped with meat and vegetables**
A bowl with rice, broccoli and a purple relish. *A plate of broccoli, rice, meat and other vegetables* *A bean and corn mixture, rice, and broccoli on a plate* *Looking down at a bowl of white rice, broccoli and a vegetable dish* *The meal in the bowl has rice and broccoli in it.*

Table 2: Sample Results: **Our Result** & *The Ground Truth*

7. CONTRIBUTIONS

Saurabh Mirani: Worked on the Encoder. Worked on the Decoder (added Teacher Forcing). Also worked on creating plots, and visualizations
Eunji Song: Worked on the Decoder, debugged and merged all our codes.
Shiladitya Biswas: Worked on the data-loader and dictionary building code. Trained the model on the local Machine and collected the relevent performance data.

8. REFERENCES

[1] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[2] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. *J. Artif. Intell. Res.*, 47:853–899, 2013.

[3] Polina Kuznetsova, Vicente Ordonez, Alexander Berg, Tamara Berg, and Yejin Choi. Collective generation of natural image descriptions. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 359–368, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

[4] Polina Kuznetsova, Vicente Ordonez, Tamara L. Berg, and Yejin Choi. TreeTalk: Composition and compression of trees for image descriptions. *Transactions of the Association for Computational Linguistics*, 2:351–362, 2014.

[5] Girish Kulkarni, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Baby talk: Understanding and generating simple image descriptions. In *CVPR 2011*, pages 1601–1608, 2011.

[6] Siming Li, Girish Kulkarni, Tamara L Berg, Alexander C Berg, and Yejin Choi. Composing simple image descriptions using web-scale n-grams. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 220–228, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[7] Yezhou Yang, Ching Teo, Hal Daumé III, and Yiannis Aloimonos. Corpus-guided sentence generation of natural images. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 444–454, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[8] Margaret Mitchell, Jesse Dodge, Amit Goyal, Kota Yamaguchi, Karl Stratos, Xufeng Han, Alyssa Mensch, Alex Berg, Tamara Berg, and Hal Daumé III. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 747–756, Avignon, France, April 2012. Association for Computational Linguistics.

[9] Desmond Elliott and Frank Keller. Image description using visual dependency representations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1292–1302, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[10] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, page 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[11] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal neural language models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 595–603, Bejing, China, 22–24 Jun 2014. PMLR.

[12] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn), 2015.

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

[14] 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.

[15] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[16] 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, 2018.

[17] Wei Liu, Sihan Chen, Longteng Guo, Xinxin Zhu, and Jing Liu. Cptr: Full transformer network for image captioning, 2021.

[18] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing.

[19] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014.

[20] Kelvin Xu. arctic-captions. *https://github.com/kelvinxu/arctic-captions*.

[21] AaronCCWong. Show, attend and tell: Neural image caption generation with visual attention. *https://github.com/AaronCCWong/Show-Attend-and-Tell/*.

[22] yunjey. Image captioning. *https://github.com/yunjey/pytorch-tutorial/tree/master/tutorials/03-advanced/image*$_c aptioning$.

[23] yunjey. Show, attend and tell. https://github.com/yunjey/show-attend-and-tell.

[24] sgrvinod. a-pytorch-tutorial-to-image-captioning. https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning.

[25] bentrevett. pytorch-seq2seq. https://github.com/bentrevett/pytorch-seq2seq.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[27] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks, 2017.

[28] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks, 2017.

[29] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.*

[30] Pytorch Team. Resnet. https://pytorch.org/hub/pytorch_vision_resnet/.

[31] Pytorch Team. Wide resnet. https://pytorch.org/hub/pytorch_vision_wide_resnet/.

[32] Pytorch Team. Resnext. https://pytorch.org/hub/pytorch_vision_resnext/.

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

[34] Stjepan Ložnjak, Tin Kramberger, Ivan Cesar, and Renata Kovačević. Automobile classification using transfer learning on resnet neural network architecture. 01 2020.

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation*, 9:1735–80, 12 1997.*

[36] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics.*

[37] Python software foundation. python language reference, version 2.7. available at http://www.python.org.

[38] pypi. pycocotools. https://pypi.org/project/pycocotools/.

[39] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3128–3137, 2015.*

9. REPLY TO REVIEW

9.1. Review from G2

Comments: Group 21 introduced an architecture where they used CNN as encoders to extract image features and LSTM and attention layer as decoders to generate sentence describing the image based on the features, which was first introduced by Xu in 2015. I personally like how they explain the model with details and images, which help me to understand the whole process of the project. The images from attention layer also explain how this attention works in this project. To sum up, they did a great job on image captioning. The slides and presentation are easy to understand.

Some improvements/unclear:

1. I noticed that the best accuracy is around 75% which I believe is not high enough to appropriately describe an image. From your results, the accurate description of the first image should be a knife sitting on top of a table next to a cup of coffee but the model predicts the knife as a pair of scissors. What can you do to improve the performance?

   > Please refer to **Section 5.2** for the evaluation metric. It's not the same as accuracy as accuracy doesn't make sense in sentences. The most important is the BLEU-4 score. To give an idea of the performance of our model, the BLEU-4 score of a human is around 22. Refer Show and Tell [13] where they have compared their model with human BLEU-4 score. However, this doesn't mean that our model is better than humans for any given image, since the model's dictionary is limited to the words that exist in training set.
   > Regarding the difference between knife and scissor, not that these two items actually look pretty similar when viewed from sideways, hence the model made an error. To improve we tried different Encoder networks and found that 'ResNeXt' gave us the best result. This signifies the importance of the encoder network. Previously, with ResNet152 the caption generated didn't make much sense, but with 'ResNeXt' the caption generated was **'a cup of coffee sitting on top of a table'**
   > We currently are not training the Encoder network but use pre-trained weights, hence another approach would be to fine-tune the Encoder network in our model which wouldn't have been possible in the given timeframe. Recently people have started using "Transformer([15])" and use Transformer even replacing the CNN encoder network ([17]). However, we are not fully familiar with this Transformer network and implementing this could be a new project to start with in the future.

2. The accuracy curves seem to saturate after around 5 epochs. I believe your model stops learning much after the first few epochs. Can you improve the model so that it will continue learning from the data, which may also help the performance?

   > As we mentioned earlier, we tried using more complicated encoder model. Because images have very complex information, we thought that the model will continue learning from the data when the model has more information about the image.

3. Since the decoders use an existing dictionary to generate sentences describing the images, maybe you can try a larger dictionary which may include more words so that it will generally help describing more images.

   > We used fairly large dataset for training, the MS COCO. It contains quite large vocabulary. Simply adding new words to our dictionary wouldn't make sense since the model wouldn't know how to use those words. So we would have to generate captions on our own in order to expand our dictionary which is beyond the scope of this project. In addition, our result did not show unknown tag ,i.e. $< unk >$ that much, so we do not think enlarging vocabulary is a good idea to get a result with good performance.

4. Did you try any other metric for the accuracy?

   > BLEU is a metric that is most commonly used in text generation or machine translation. Our work is basically generating new sentence from scratch so traditional metrics such as F1 would not work well. We did use the loss value while training to see if the model is learning. We can also try ROUGE, which is also widely used in text generation, however, ROUGE is more about recall whereas BLEU is more about precision. We wanted to focus more on precision.

In order to overcome using just one metric, we used four different kinds of BLEU. i.e. BLEU-1,2,3,4. These four metrics basically measures different characteristics of the results because BLEU-1 looks at only one word precision, but BLEU-4 also looks at precision of four consecutive words.

9.2. Review from G10

Comments: Team outlined the background and literature review well. It was clear as to why this problem needed to be addressed and how ML/DL could solve it. The model was explained well, and the team clearly stated which parts were done by scratch and which were pretrained. Visuals made the presentation easy to digest.

Some improvements/unclear:

1. Could have explained the choice of loss function and elaborated on the training process. Basically, what loss function did you use and why? What metrics are used for this space and why?

   We used Cross Entropy Loss because we aim to minimize the loss, and we used Adam optimizer because this is known to work well. We did not mention this on the presentation or on this report because we think this is too trivial to put in a limited space.

2. Many of the terms used in the presentation are specific to this problem - elaborate to make the presentation accessible for a wider audience. Such as, what are LSTM, RNNs, attention, etc. in detail

   Explained LSTM and Attention in more detail in this report. We did not explain RNN in detail because it is too trivial to explain in a limited space.

9.3. Review from G13

Comments: This project tries to use deep learning methods to resolve image captioning problem. They introduce the encoder decoder model and the details of the model. Their slide and video introduce each part of the model concretely.

Some improvements/unclear:

1. They can use some non DL methods to resolve this problem and compare the performance of non DL and DL.

   The sample results of the retrieval based method and template based method are already given in the presentation. As we explained earlier in the presentation and in this report, non-DL method is bounded by the pre existing caption or the given template, can not generate the new sentence that fits specifically to the input image. Instead, we compared with other methods that also use DL.

2. They can introduce other methods that can resolve the image captioning problem. T he accuracy seems not high enough.

   (Same answer to the first critique by G2 applies here too.) Please refer to **Section 5.2** for the evaluation metric. It's not the same as accuracy as accuracy doesn't make sense in sentences. The most important is the BLEU-4 score. To give an idea of the performance of our model, the BLEU-4 score of a human is around 22. Refer Show and Tell [13] where they have compared their model with human BLEU-4 score. However, this doesn't mean that our model is better than humans for any given image, since the model's dictionary is limited to the words that exist in training set.
   Regarding the difference between knife and scissor, not that these two items actually look pretty similar when viewed from sideways, hence the model made an error. To improve we tried different Encoder networks and found that 'ResNeXt' gave us the best result. This signifies the importance of the encoder network. Previously, with ResNet152 the caption generated didn't make much sense, but with 'ResNeXt' the caption generated was **'a cup of coffee sitting on top of a table'**
   We currently are not training the Encoder network but use pre-trained weights, hence another approach would be to fine-tune the Encoder network in our model which wouldn't have been possible in the given time-frame. Recently people have started using "Transformer([15])" and use Transformer even replacing the CNN encoder network ([17]). However, we are not fully familiar with this Transformer network and implementing this

could be a new project to start with in the future.