Combining NLP and Computer Vision to Help Blind People

Stanford CS224N Custom Project

Volha Leusha
Department of Computer Science
Stanford University
leusha@stanford.edu

March 17, 2020

Abstract

This paper is about an attempt to help visually impaired population by solving image captioning task for VizWiz dataset [12]. This dataset consists of images taken by blind people in real-world situations and is the best choice to address their needs. It has been overlooked by researches comparably to other commonly used visual-linguistic datasets, and consequently requires further investigation. In this project several models based on CNN-LSTM architecture [2], [11] and Transformer [14] are constructed. All models show solid performance with the best result achieved by CNN-LSTM model with attention in combination with beam-search inference. Additionally, the dataset limitations are explored, and it can be concluded that algorithms pre-trained on artificially created datasets like MSCOCO perform poorly when deployed on VizWiz.

Key Information

Mentor: Sarah Najmark Grading policy: 2

1. Introduction

Image captioning is a process of generating textual description from images. It requires not only understanding of objects interactions in the scene, but also translating them into natural language [5].
There are plenty of user cases for image captioning and other visual-linguistic applications, namely retrieving an image from internet or helping objects to orient in space.
Another important example is related to improving life of blind people who daily rely of image captioning services to help them to learn about the world around. All such services are based on human assistance that is slow and costly process. Moreover, users have to share sensitive for them information with random volunteers, and thus, often their privacy is violated.
While automating image captioning task has been an important topic in vision community, most of this research is done on artificially constructed datasets like MSCOCO and Flickr30k. This datasets are not applicable to address needs of users in the real word situations. Specifically, images taken by blind people and collected in VizWiz-Caption [12] exhibit different conditions than observed in contrived environment of artificially created datasets. Examples are presented on Figure 4.
As was shown in [1] these differences result in a poor performance of pre-trained models on VizWiz-Caption dataset. This leads to necessity to search for the algorithms and parameters that are better suitable for the task.
In this work several different approaches are explored. The baseline model is based on CNN-LSTM architecture described in [2] with multiple modifications to boost performance. In addition, two different architectures are explored: CNN-LSTM with attention based on [11] and CNN-transformer [14].

2. Related Work

Image Captioning

In earlier days image description generation was based on templates [10]. This methods detected different objects and interactions and used this information to fill blank spaces in template slots [9]. In recent decade image captioning with deep learning is gaining popularity. It proved to handle challenges and complexities

of image description better than traditional machine learning based techniques [3].
Most of deep learning techniques are based on encoder-decoder architecture inspired by neural machine translation [5]. The image encoder is typically represented by a convolutional neural network (CNN). The decoder can take different forms. For example, Long-Short Term Memory (LSTM) net was proposed by Vinyalset et al. in [2]. This model is used as a baseline model for this work, and to my knowledge was never applied to VizWiz-Caption dataset.
Another popular and successful approach in modern image captioning utilizes the attention mechanism [16]. It is based on human intuition to pay attention to certain places on the image while interpreting the scene. There are different types of attention: soft attention and hard attention [11], spacial and channel-wise attention [18], multi-head and self-attention [21]. In [21] Vaswani et al. were able to achieve state-of-art performance for machine translation using only self-attention mechanism. Instead of LSTM Transformer model is used in both encoder and decoder. In [14] authors use the same self-attention (Transformer) method to solve image captioning task for MSCOCO dataset. They substitute encoder transformer model with CNN to produce image features instead of word embeddings. This architecture allows to boost performance by more that 20% on most of the metrics comparison to [2] without significant increase in computation time.
The most recent papers in this field utilize more complex transformer architectures. In [5] attention-on-attention (AoA) approach is used which adds another attention on top via element-wise multiplication.
Such models as VLBERT[4] and VILBERT[26] adopt powerful transformer model BERT architecture to take both visual and linguistic embedded features as input. Currently they achieve state-of-art performance on popular visual-linguistic datasets.
Performance results for AoA on VizWiz-caption dataset are available in [1]. However, for Soft-Attention model [11] and Transformer [14] no attempts are made so far. Also, for all discussed models performance results are available for other visual-linguistic datasets which provides benchmarks and makes them promising to implement in this work.

VizWiz

VizWiz-Caption is a new dataset released just recently. To the best of my knowledge there is only one paper published up to date [1]. In the paper the dataset is introduced to researches and results on test split for three models (Attention-on Attention [5], Top-down attention [22], SGAE [23]) are presented. This paper contains important for this project data analysis. Moreover, several other papers for related VizWiz-VQA dataset are available [19], [20]. 80% of images overlap with VizWiz-Caption, but the task is different - visual question answering.

3. Approach

In this work three different models are benchmarked to gauge the difficulty of VizWiz-Caption dataset for modern algorithms. Publicly-shared code is used and adopted to meet the needs of the project. Details are explained below. Moreover, both default parameters provided by authors, as well as fine-tuned parameters are tested. The results are compared with each other and with results of this methods on MSCOCO.

3.1. Base CNN-LSTM Model

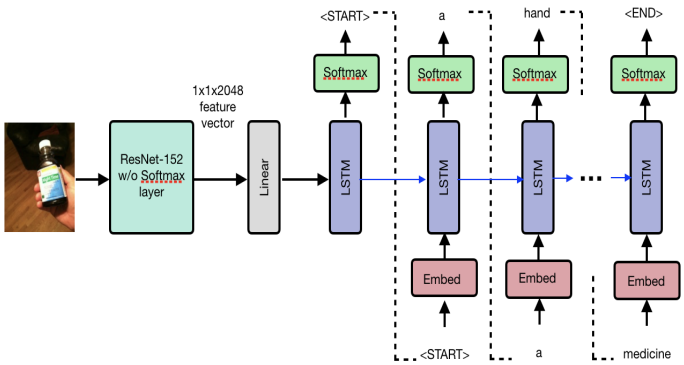

Figure 1 CNN-LSTM Architecture

The base model presented in this work is built on encoder-decoder architecture. More precisely the algorithm is adopted from CNN-LSTM structure described in [2] and is shown on the Figure 1. The encoder is a deep convolutional neural network (CNN) with deleted softmax layer. It produces embeddings of fixed-length vectors from the input images.

Being one of the best performance methods in image classification, ResNet-152 [7] (pre-trained on ImageNet dataset) is used as encoder in this project. However, the model can be easily changed to any other CNN architecture.
The decoder is LSTM model that can be described as:

$$x_{-1} = CNN(image)$$

$$x_t = W_e S_t, t \in \{0...N-1\}$$

$$p_{t+1} = LSTM(x_t), t \in \{1...N-1\}$$

where x_{-1} are output CNN features, p_{t+1} is a word predicted by LSTM in t^{th} step, and $S = (S_0, ..., S_N)$ is a true sentence describing the image.
The first LSTM cell takes features produced from images and special start word S_0 and predicts next word in the caption - S_1. The second cell takes S_1 and predicts second word in the caption S_2. This step is repeated until the special stop word S_N is produced. Each LSTM cell shares the same parameters W_e. The training objective function is Cross Entropy loss for all the methods.
Limitations of base model: This baseline model has multiple limitations, e.g. it is relatively small, lacks attention to important features, and has high computation complexity due to sequential nature. To improve the model following hypothesis were tested:

- To improve performance computational time - fine-tune parameters
- To improve computational time - freeze first half of CNN layers (no gradient propagated through)
- To improve performance - increase amount of LSTM layers from 1 to 2
- To improve performance - add attention mechanism
- To improve performance - add beam-sampling
- To improve performance and computational time - implement transformer model

Code: The code for the algorithm is based on Pytorch tutorial [6]. The model allows to select any number of LSTM layers and size of hidden states. Tested model configurations are described in details in Experiments section. The code from tutorial is adapted to work on VizWiz dataset structure and collect necessary statistics, e.g. loss and model parameters for each epoch. Inference step for the whole validation set and model evaluation step, calculating metrics like BLEU, CIDER, etc. [8] is added. Additionally, the model required learning rate fine tuning to avoid over-fitting.

3.2. CNN-LSTM Attention Model

The attention model is built on top of the base model using approach presented in [11] and [24] and is shown on Figure 2.
The lower layers of encoder consist of ResNet-152 with deleted softmax layer. The last layer of encoder (Adaptive Pooling Layer) is changed to output 9x9 attention regions (instead of 1x1 in base model) to allow the model to attend to certain areas of an image.
The decoder is LSTM model with attention. In contrast to the base model it looks at image features not only at the first step of decoding process, but at every step. This allows to learn to attend to specific locations in the image for each generated word. The decoder can be described as:

1. From feature vectors create initial cell and hidden state of LSTM h_0, c_0:

$$c_0 = f_{init,c}(\frac{1}{L}\sum_i^L a_i), h_0 = f_{init,h}(\frac{1}{L}\sum_i^L a_i)$$

   , where a_i, $i = 1, ..., L$ correspond to the image features extracted at different image locations.

2. At each decode step t:

   - Use encoded image a and hidden state h_{t-1} to generate attentions weights α_{ti} for each pixel in a. For this purpose use the soft attention mechanism:

$$\alpha_{ti} = softmax(f_{att}(h_{t-1}, a_i))$$

$$f_{att}(h_{t-1}, a_i) = tanh(W_a a_i + W_h h_{t-1})$$

- Feed the previously generated word $word_{t-1}$ ($w_0 =< start >$) and α_{ti} to the decoder to generate the next word $word_t$

The model extracts information from images and passes this information to every step of decoder. This helps to boost performance in comparison to the base model. However, due to LSTM sequential nature the model is still slow.

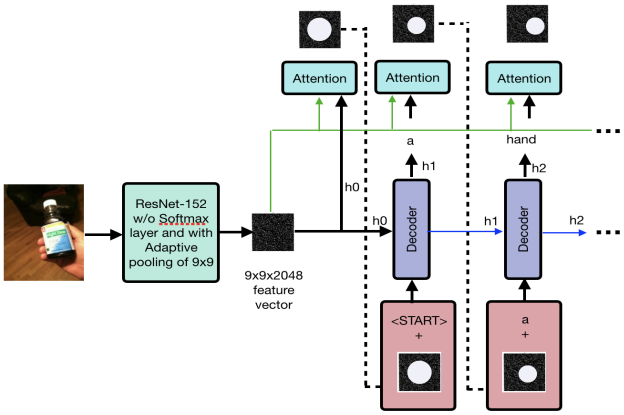

Figure 2 CNN-LSTM Architecture with Attention

Code: The code for the algorithm is built on top of the base model, and is partially adapted from Pytorch tutorial [24]. The model required learning rate fine tuning to avoid over-fitting as well as adaptation of batch and layer size parameters to fit on single GPU.

3.3. Transformer Model

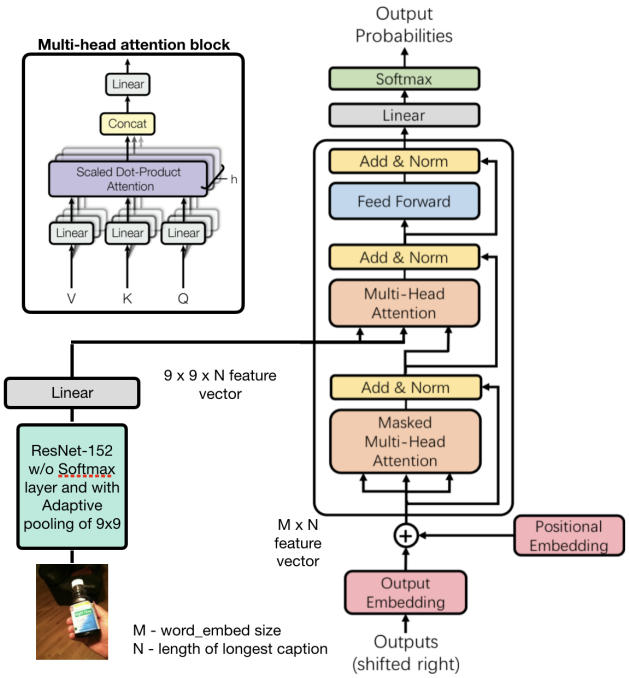

Figure 3 Transformer Architecture [21]

Transformer model created in this project is based on [14] that adapts the machine translation model described in [21].

The encoder is similar to encoder used in CNN-LSTM Attention Model with the only difference that CNN output is adapted to match word embedding size. It is necessary for multiplication in Multi-head attention layer.

The decoder has more complicated structure in comparison to models above. However, as it processes the whole sentence rather than one word at a time, it is more computationally efficient.

Transformer is not sequential, therefore to fully explain captions it is necessary to input not only words

embeddings, but also positions of words in sentences. This is done by addition of positional embedding. The model uses the mechanism of multi-head attention. This layer is composed of n scaled dot-product attentions that can be described as [21]:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

where Q and K queries and keys of dimension d_k, and V is a values.
The feed-forward layer helps to deepen the network. It uses linear modules to analyse dependencies in the attention outputs, and consists of two linear transformations with a ReLU activation in between:

$$FFN(x) = max(0, xW_1 + b_1)W_1 + b_1$$

Normalization is a common method in deep neural networks that allows to speed up training and ability to generalize. For more details about the layers refer to [21].
Code: The code for the algorithm is adapted from github repository [13]. It was originally created for machine translation task and therefore only decoder part could be reused. Encoder was substituted with CNN. Additionally, the model required learning rate fine tuning and adaptation of batch and layer size parameters to fit on single GPU.

3.4. Inference

For the inference, two methods are used. First is sampling, e.g. sample the first word according to p_1, then provide the corresponding embedding as input and sample p_2, repeat until end-of-sentence token or maximum length is reached. Second is a beam search. In the beam search a set of k best sentences up to time t is considered to generate sentences of size $t + 1$. Only the best k of them are kept iteratively. This approach is slower than sampling if $k > 1$, but provides better results [2].

4. Experiments

4.1. Data

The VizWiz-Captions dataset consists of approximately 39 thousand images each paired with 5 captions (17.5 GB). The images are captures by people who are blind in real words situations. Therefore, it exhibits different conditions than observed in the contrived environments of artificially created datasets, like MSCOCO and ImageNet. The task is to generate textual description of a given image. See examples below. The dataset is divided into 3 splits: train, validation and test data 23431/7750/8000. Train and validation splits have five captions associated with each image, while the test split does not have any captions. For research purposes, test split is left out from the scope, and custom split is created. The validation split of the dataset is shuffled and divided in half. First half is used to validate and second half to test model performance.

Figure 4VizWiz-Caption vs MSCOCO

4.2 Evaluation method

For evaluation on image caption task BLEU is used in combination with CIDEr and ROUGE. The most commonly used metric in the image caption literature is BLEU score [17]. It is based on precision of word n-grams between predicted and target sentences. ROUGE is recall-based metric and is a good addition to the BLEU. CIDEr is a consensus-based image description evaluation. The metric shows high agreement with human based ground truth. Using sentence similarity, the notions of grammaticality, saliency, both precision and recall are captured well by CIDEr [25]. The benchmark data on these metrics is provided in [1] for several state-of-art models on test split of VizWiz-Cations. As results are only available for the test split of VizWiz-Caption it can only be compared with results presented in Table 2 under assumption that test and validation splits have similar distributions.

4.3 Experimental details

Details about model configurations explored in experiments, model complexity and training time per epoch on NVIDIA Tesla M60 GPU are presented in Table 1. All models are trained first on parameters suggested for MSCOCO dataset. Then, fine-tuning is performed, and parameters with best performance for each configuration are reported in Table 1. Moreover, only configurations that fit on single GPU are considered.

Table 1 : Model configurations explored in experiments

∗ Pre-trained(PM), CNN frozen-tuned (FT) and trained from scratch (TFS) configurations

Model Configurations		Parameters											
		Encoder CNN	Size of word embeddings	Number of layers	Size of hidden states	Attention size	Optimizer	Loss	epochs	Learning rate	Batch size	Training Time, min (per epoch)	Model Size, M
CNN-LSTM	PM	Resnet 152	256	1	512	n/a	Adam	Cross Entropy	5	0.001	128	40	66.4M
	TFS			1								45	68.6M
				2									
CNN-LSTM with Attention	PM	Resnet 152	512	1	512	14x14	Adam	Cross Entropy	10	0.0004	32	120	78.5M
	FT					9x9				0.0001		60	
	TFS											90	
Transformer	PM	Resnet 152	512	6	512	4x4	Adam	Cross Entropy	20	0.0004	16	-	92.7M
	FT		256			6x6				0.0001		50	
	TFS											70	

Pre-trained(PM), CNN frozen-tuned (FT) and trained from scratch (TFS) configurations have the same model structure. However, the first is pre-trained on MSCOCO; for the second lower layers of CNN are frozen to avoid gradient propagation; and the third is trained from scratch on VizWiz-Caption dataset. It can be seen that introduction of FT improved computation time by a factor of 1.5. Moreover, as anticipated the training time for Transformer is faster even though transformer model has more parameters (due to parallel computations).

For all CNN-LSTM models sampling strategy is used for the inference. The beam search with beam = 3 is tested for the attention model and results are presented in Table 2.

To optimize training Adam optimization technique is utilized for each model. To avoid over-fitting dropout of 0.1 and early stopping is used. To visualize over-fitting and choose epoch for early stopping, validation and train cross entropy loss and results on BLEU, CIDER and ROUGE are collected for each epoch. From Figure 4 it can be seen that loss decreases with increase in iterations. Consequently, the training objective is achieved. Validation results also achieve minimum for all configurations signalizing that models do not need to be trained further. For example, CNN-LSTM overfits after 5-th epoch and training can be stopped when validation minimum is achieved. From the figure it can also be seen that choosing the right learning rate is extremely important to achieve better performance. Fine-tuned learning rate of 0.0001 vs suggested for MSCOCO learning rate of 0.0004 allowed to significantly boost results of CNN-LSTM with attention on all the metrics. For additional learning curves check Appendix.

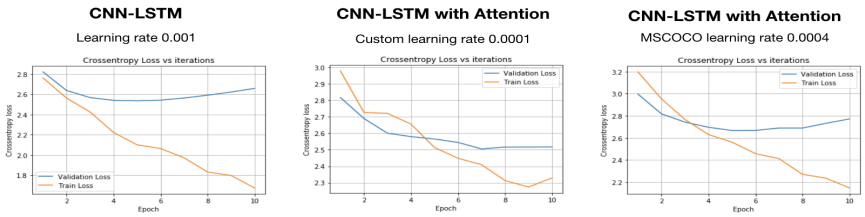

Figure 5 Loss versus iterations

4.4 Results

For each model configuration performance results on VizWiz-Caption (custom split) and MSCOCO datasets are presented in Table 2. Also in Appendix on Figures 9 and 10 two graphs are provided to help visualize the table.
From the table is can be concluded:

- Models trained on MSCOCO perform purely on VizWiz-Caption dataset and vice versa. Moreover, even TFS model configurations that adopt MSCOCO parameters perform worse than models with fine-tuned parameters. It proves that MSCOCO and VizWiz-Caption indeed have a very different data distributions, and models trained on artificially created data are not ideal for the real word situations.
- Introduction on beam-search with beam > 1 helps to boost model performance. This is anticipated especially for metrics like BLEU-3 and BLEU-4 that take into account multiple subsequent words, rather than a single word.
- In general performance for VizWiz dataset is lower than for MSCOCO dataset. This result is expected as images in VizWiz are created in the real word situations and are less perfect than images selected artificially.
- Increasing amount of layers in LSTM from one to two does not help to improve model performance. Probably, because captions have relatively simple structure and one layer LSTM is already enough for reconstruction.
- Unexpectedly transformer is not able to outperform CNN-LSTM with attention. Learning curve presented in Appendix suggests that optimal parameters for transformer architecture are not achieved and models just requires additional fine-tuning.

Table 2 : Performance results for VizWiz-Caption (custom split) and MSCOCO datatsets
∗ MSCOCO settings: model parameters adopted from papers and recommended by authors
∗ Custom settings: fine-tuned on VizWiz model parameters presented in Table 1
∗ Beam search: beam size = 3

Model Configurations		VizWiz Performance						Ms COCO Performance					
		BLEU1	BLEU2	BLEU3	BLEU4	ROUGE	CIDER	BLEU1	BLEU2	BLEU3	BLEU4	ROUGE	CIDER
CNN + LSTM	PM	37.4	20.6	10.0	5.0	24.2	6.1	54.2	36.8	24.1	15.5	36.6	41.2
	TFS with 1 LSTN Layer	50.9	34.8	24.1	18.1	35.5	26.9	35.4	18.4	8.4	4.1	23.8	7.7
	TFS with 2 LSTN Layers	49.3	33.1	22.0	15.9	33.9	24.3	data is not available					
CNN + LSTM with Attention (ATT)	PM [11]	trained model is not available						70.7	49.2	34.4	24.3	no data available	
	FT	**54.6**	**39.2**	28.5	21.9	**38.7**	34.7	46.2	20.1	11.4	7.2	24.8	10.2
	TFC with MSCOCO settings	40.6	28.0	19.4	14.6	32.7	25.2	27.1	9.8	3.4	2.2	18.9	1.8
	TFS with custom settings	53.9	38.6	27.5	20.8	38.2	33.5	26.1	10.6	3.8	1.5	17.6	1.7
	TFC + Beam search	48.4	39.0	**34.1**	**27.5**	35.4	**39.1**	data is not available					
Transformer (TR)	PM [13]	trained model is not available						71.8	55.4	41.8	31.5	54.7	105.8
	TFS with custom settings	51.4	35.9	25.7	19.7	37.1	28.9	32.6	16.3	6.8	2.9	23.2	4.6

5. Analysis

On Figure 6 captions produced by models are visualized. Results support findings from the metrics in Table 2:

- The bottle on the left-bottom picture is described as the cell phone by CNN-LSTM PM, but is correctly identified by CNN-LSTM TFS model. This result is expected as bottle with medicine is associated more with a real life of blind people rather than with artificially constructed data.
- The similar can be concluded about the bear on the right-top image. The caption produced by all TFS on VizWiz models does not make sense, because this kind of images are not likely to be present in a real data from blind people.
- In some cases model pre-trained on MSCOCO performs well on VizWiz dataset and vice versa. For example, for the right-middle picture from MSCOCO images TFS performs well. It is not surprising as there are a lot of images of living spaces in the VizWiz-Caption.
- The image on the right-bottom corner shows a situation were MSCOCO data partially overlaps with VizWiz data. The VizWiz dataset has plenty of pictures with people. However, the captions mostly explain how this people look and what they wear. Sport activities are not so common in VizWiz data,

in contrast to MSCOCO data. Consequently, it is not surprising that TFS models are able to recognize a person wearing blue and white shirt, but are not able to catch information about tennis.

- In general CNN-LSTM with attention and Transformer models produces better captions as can be seen on most images. This supports findings from the Table 2. However, as metrics are averaged for the whole dataset, there are exceptions to this statement. For example, on left-bottom picture the medicine bottle was better recognized by CNN-LSTM model without attention.
- Another interesting example is presented on the left-middle picture. It shows that images captured by blind people can be of a poor quality and this situation is handled in VizWiz dataset by assigning caption to 'low quality' category. This answer is beneficial as it is better to acknowledge that image quality is poor than to give a wrong description. This kind of situations are purely handled in models trained on MSCOCO and other artificial datasets.

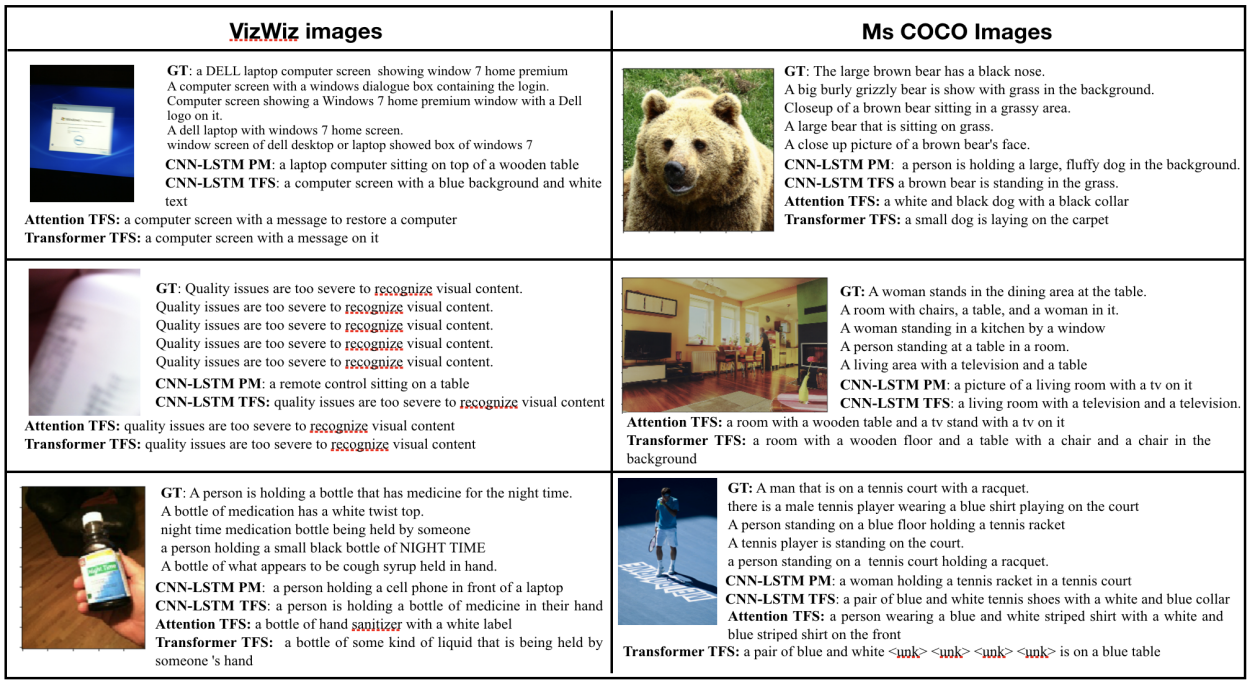

Figure 6 Images and Captions

Visualization of attention mechanism is shown on the Figure 7. Additional examples are presented in Appendix. From the figure it can be seen that the model indeed pays attention to pieces of images that are associated with predicted words, like 'bottle', 'on', etc.

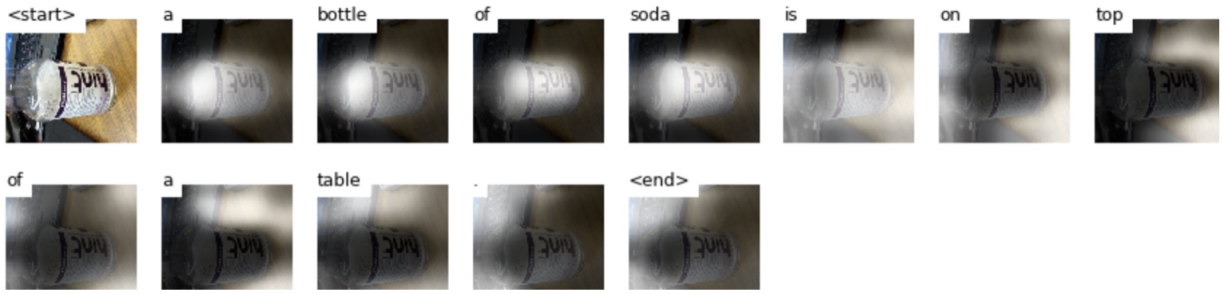

Figure 7 Attention visualization

6. Conclusion

This work shows that modern state-or-art algorithms pre-trained on other visual-linguistic dataset are not suitable to achieve top performance on VizWiz-Caption. This proves that dataset distribution is unique to the task and requires model fine-tuning or training from scratch on VizWiz data.

For all the models presented in this work solid performance is achieved on custom dataset split. The best performer is CNN-LSTM model with attention and beam search inference with beam = 3. As addition of attention to the model shows to be beneficial to boost performance on VizWiz-Caption, exploring another

architectures with different types of attention is suggested for the next step. Moreover, further fine-tuning of Transformer model can be done.

References

[1] Danna Gurari, Yinan Zhao, Meng Zhang, Nilavra Bhattacharya *Captioning Images Taken by People Who Are Blind* arXiv:2002.08565v1

[2] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan *Show and tell: A neural image caption generator* CoRR, abs/1411.4555, 2014

[3] Md. Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, Hamid Laga *A Comprehensive Survey of Deep Learning for Image Captioning* arXiv:1810.04020v2

[4] Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, Jifeng Dai *VL-BERT: Pre-training of Generic Visual-Linguistic Perpesentations* arXiv:1908.08530, Nov 2019

[5] Lun Huang, Wenmin Wang, Jie Chen, Xiao-Yong Wei *Attention on Attention for Image Captioning* arXiv:1908.06954, Aug 2019

[6] *https://github.com/yunjey/pytorch-tutorial/tree/master/tutorials/03-advanced/*

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun *Deep Residual Learning for Image Recognition* arXiv:1512.03385, Dec 2015

[8] *https://github.com/ruotianluo/coco-caption/tree/ea20010419a955fed9882f9dcc53f2dc1ac65092/pycocoevalcap*

[9] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth *Every picture tells a story: Generating sentences from images* In European conference on computer vision. Springer, 15–29.

[10] Benjamin Z Yao, Xiong Yang, Liang Lin, Mun Wai Lee, and Song-Chun Zhu. I2t: *Image parsing to text description.* Proceedings of the IEEE, 98(8):1485– 1508, 2010.

[11] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, Yoshua Bengio *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention* arXiv:1502.03044

[12] *https://vizwiz.org/tasks-and-datasets/image-captioning/*

[13] *https://github.com/SamLynnEvans/Transformer*

[14] Xinxin Zhu, Lixiang Li, Jing Liu, Haipeng Peng and Xinxin Niu *Captioning Transformer with Stacked Attention Modules* Appl. Sci. 2018

[15] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. *Show, attend and tell: Neural image caption generation with visual attention.* In ICML, 2015.

[16] Ronald A Rensink. *The dynamic representation of scenes.* Visual Cognition, 7: 17–42, 2000.

[17] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk *Understanding and generating simple image descriptions.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(12):2891–2903, 2013.

[18] Long Chen, Hanwang Zhang, Jun Xiao, Liqiang Nie, Jian Shao, Wei Liu, and Tat-Seng Chua. Scacnn *Spatial and channel-wise attention in convolutional networks for image captioning.* In CVPR, 2017.

[19] Danna Gurari, Qing Li, Abigale J. Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P. Bigham. *VizWiz Grand Challenge: Answering Visual Questions from Blind People* IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[20] Danna Gurari, Qing Li, Chi Lin, Yinan Zhao, Anhong Guo, Abigale J. Stangl, and Jeffrey P. Bigham. *VizWiz-Priv: A Dataset for Recognizing the Presence and Purpose of Private Visual Information in Images Taken by Blind People* IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

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

[22] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. *Bottom-up and top-down attention for image captioning and visual question answering.* In CVPR, 2018

[23] X. Yang, K. Tang, H. Zhang, and J. Cai. *Auto-encoding scene graphs for image captioning.* In Proceedings of the IEEE Conference on Computer Vision and Pattern Recogni- tion, pages 10685–10694, 2019.

[24] *https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Image-Captioning*

[25] Ramakrishna Vedantam, C. Lawrence Zitnick, Devi Parikh *CIDEr: Consensus-based Image Description Evaluation* arXiv:1411.5726v2

[26] Jiasen Lu, Dhruv Batra, Devi Parikh, Stefan Lee *ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks* arXiv:1908.02265v1

Appendix (optional)

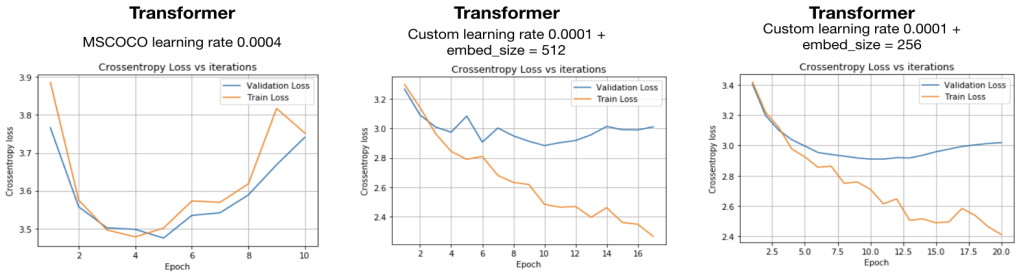

Figure 8 Learning curves for Transformer Model

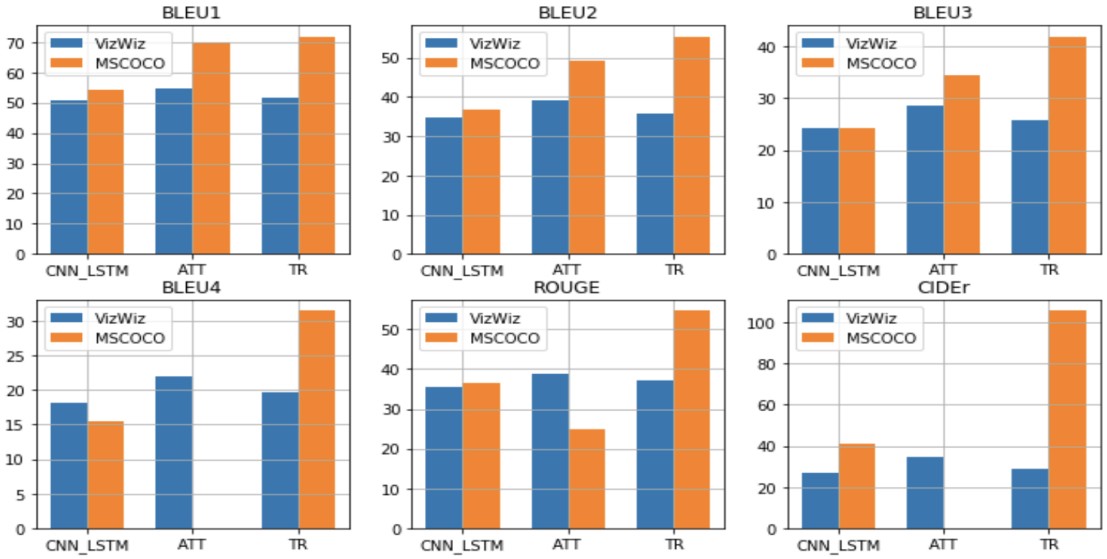

Figure 9 Metrics comparison: VizWiz vs MSCOCO

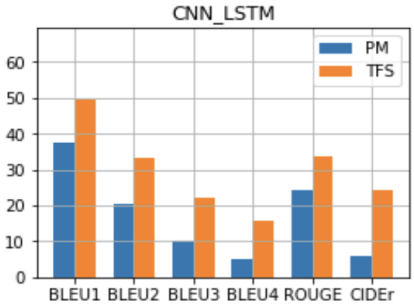

Figure 10 Metrics comparison: pre-trained on MSCOC vs trained from scratch models

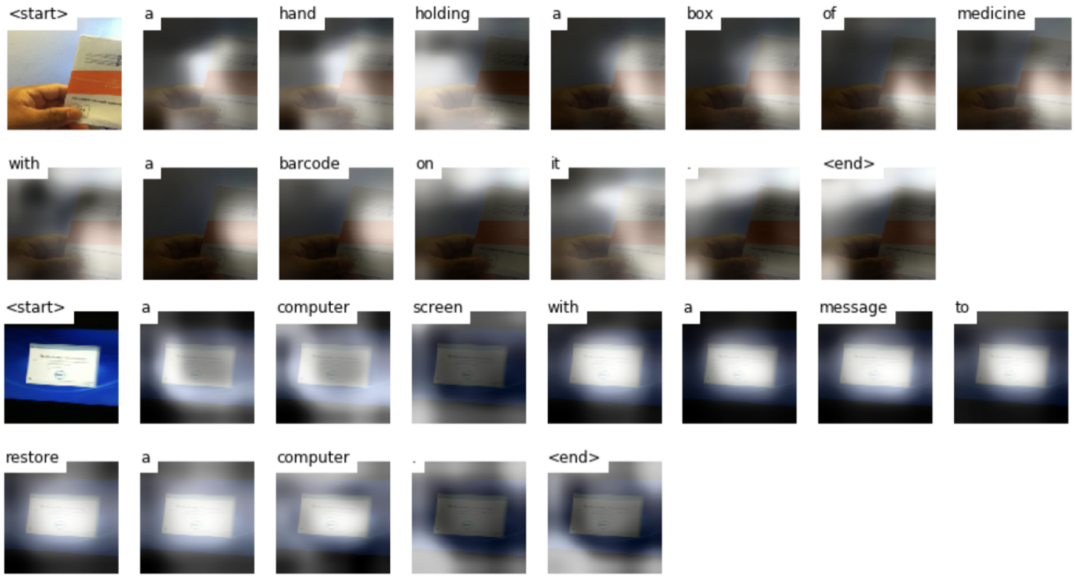

Figure 11 Attention visualization examples