Google AI Open Images – Visual Relationship Track using Deep Learning for Vision and NLP

Ashraf Neisari *Ruturaj R. Raval* *Zarreen Naowal Reza* *Zeeshan Mansoor*
{neisari, raval115, rezaz, mansoorz}@uwindsor.ca[i]
University of Windsor
401 Sunset Ave. N9B 3P4 Windsor, ON

Abstract

In this work, we have tried to solve the Visual Relationship Detection Track competition launched by Kaggle. The aim of the competition is to check if computers can detect the relationship between objects presented in images. Not only it is a very state-of-the-art research area, it is a very challenging task to accomplish compared to existing computer vision tasks. It is a combination of two prominent tasks – object detection and image caption generation. Although, deep learning models are able to produce highly accurate results in these individual tasks, it still struggles to perform with acceptable accuracy in the visual relationship detection task. In this paper, we have attempted a different approach to solve this problem and compared the result with the state-of-the-art baseline result. We have explored two attention-based caption-generation models from Bengio et.al. (2016) [1] and Fei-Fei et.al. (2015) [2] and modified them to solve the visual relation detection task.

1. Introduction

Generation of captions for an image automatically is a very challenging task and inline with scene understanding, an important goal of computer vision. [5] It can also be very important in many applications including helping visually impaired individuals partly understand what other people can see. The task is considerably harder than image classification and object recognition since it needs to capture and express the relationships in a natural language. Objects attributes and the activities they are involved with are then used for DCI, as well as language models. *Describing the content of an image* (DCI) is an important task that can be mechanized using *Machine Learning* (ML), and so has been the focus of some research for the past years. It combines the two fundamental aspects of ML, namely *Computer Vision* (CV) and *Natural Language Processing* (NLP), especially machine translation, in order to achieve its goals. Recent advances in CV and NLP have enabled researchers to come up with new methods to dramatically improve the performance of the DCI and making it a viable automated task. Deep learning has offered great and powerful models to work with large volumes of labelled data to learn the features from caption generation models which helps in learning novel task-specific image representation using a state-of-the-art captioning system. The improvements of performance are measured and compared between different models like Show and Tell [2] and the DenseCap [3] models based on commonly used BLEU and Flickr scores. We are using an attention-based generative model using deep *Recurrent Neural Network* (RNN) and providing sentences for DCI. During the training, we maximized the likelihood of the resulting sentences for any provided image. This is done using standard backpropagation technique and maximizing a variational lower bound. The models work with Open Image Dataset [4] offered by Google AI to test 99K+ images to generate captions showing relationships between the objects in the image and the caption(s).

Majority of the previous attempts proposed the concept of stitching the solution for the above-mentioned problems together to come up with a solution for DCI. [6] [7] More recent research articles have proposed a more holistic approach of a single joint model of likelihood

maximization. In the deep learning approach where data plays an important role since its revolution in the area of computer vision. Data is a critical aspect of deep learning to enable the computer to a scope to interpret images as compositions of objects. We are helping the machine to learn effortlessly about image classification, object detection, and their interaction as visual relationship detection, etc.

The quality of caption generation has been significantly improved using a combination of *Convolutional Neural Network* (CNN) and RNN. The second part of the task, NLP, is inspired by machine translation which was traditionally done through a series of underlying tasks, namely translation of individual words, alignment, reordering and so on. This was drastically changed by the advent of RNN with huge performance boost as it transforms a source sentence into a vector representation. [8] [9] [10] This representation is then used as the state of an RNN to generate the target sentence. We use the same approach of providing The RNN with this representation coming from a deep CNN. Analyzing the source image. This approach is called Neural Image Captioning (NIC). Hard and soft attention models that can attend to salient part of an image, have also been used in previous research to generate caption.

In this work, we first propose our system detail for NIC. The training of the system is then described. We finally evaluate our system using measuring scores of BLEU on Kaggle dataset.

The challenge offered on Kaggle platform consists of two different tracks: object detection to predict a tight bounding box around all instances of 500+ classes; and visual relationship detection to predict pairs of objects in particular relations. [11] The visual relationship detection track requires detecting relationships connecting two objects including **human-object relationships** (an e.g. *woman playing guitar*, *man holding microphone*) and **object-object relationships** (e.g. *glass on the table*, *dog inside the car*). Each relationship connects different pairs of objects. This track also considers object-attribute relationships (e.g. handbag is made of leather, the bench is wooden). Identifying the categorized objects is an important problem on its own but identification between the objects is critical for the much real-world use case. [12]

1.1 Google AI Open Image Dataset

The dataset which we have used from the open image dataset offered by Google AI [4] contains images and ground-truth annotations for image classification, object detection and visual relationship detection. The images are collected from Flicker (www.flicker.com) without any predefined list of classes names or tags, leading to natural class statistics without being biased. All the information about the dataset acquisition and annotations, image acquisition, classes, image-level labels, candidate labels for test and validation, candidate labels for train, human verification of candidate labels, bounding boxes, extreme clicking, training annotators, annotation time, box verification series, historical process, hierarchical de-duplication, attributes, visual relationships, selecting relationships triplets, annotation process, statistics, human-verified image-level labels, bounding boxes, etc. can be found on this websites [12] [11] which is the core of execution of the dataset using mentioned models.

1.2 Visual relationship detection

In the image captioning environment, the objects in the images are the core building blocks which helps in representing the holistic interpretation of the relationship among the objects. [13] There have been few attempts in

the process of learning the spatial relationships between the objects which helps in improving the segmentation. Learning the spatial relation is very important as well as the inclusion of non-spatial relationships matters a lot. The efforts made in the visualizing and detecting the human-object interaction and action recognition which helps in processing discriminative models to find and observe the relationships between the objects, considered to be subject and the object. In the case of a subject, generally we consider human (which doesn't need to be constrained all the time) and the predicate represents the verb (doesn't apply all the times). The concept of **visual phrase** has opened many new possibilities, where the relationship is predicted in the form of the visual phrase. This method helps in detecting unseen relationships which were not detected using previous methods due to its infrequency.

1.3 Show and Tell model

The state of the art approach of **Show and Tell** uses a single joint model that takes an image I like input, and is trained to maximize the likelihood p(S|I) of producing a target sequence of words S = {S1, S2,. .}. Each word of S comes from a dictionary describing the image adequately. Over the last few years, it has been convincingly shown that CNNs can produce a rich representation of the input image by embedding it to a fixed-length vector. The representation can be used for a variety of vision tasks [14] and so it is natural to use a CNN as an image encoder. Show and Tell model has also employed a CNN by first pre-training it for an image classification task and using the last hidden layer as an input to the RNN decoder that generates sentences.

They have also used a statistical approach based on the recent evidence in Machine translation. It was shown the possibility of achieving state-of-the-art results by directly maximizing the probability of the correct translation given an input sentence in an "end-to-end" fashion – both for training and inference. A similar approach was used here for the only description of a scene.

1.4 DenseCap Model

The **DenseCap** model has made some advancements in terms of two orthogonal directions: rapid progress in object detection; recent advances in image captioning. DenseCap model is an effective model in the field of computer vision to understand the images with dense description. DenseCap helps in detecting visual concepts from images, language models from the description, etc. This approach helps in localizing and describing salient regions in images in natural language. To solve the localization and description-based task *Fully Convolutional Localization Network* (FCLN) architecture is used to process the image using a single forward pass without having external region contribution. The architecture consists of CNN and a novel dense localization layer and RNN based language model to generate a sequence of labels. DenseCap outperforms the baseline models based on the current state of the art approaches in terms of speed and accuracy. This model describes the image regions as a better interpretation of the visual content known as **dense captioning**. [15] The human annotators are required to label the bounding boxes thoroughly over ambiguous visual concepts. To solve the complexity running in the label density and label complexity axes, the framework being used to unify two inter-connected tasks are approached as: dense captioning task, using several prediction of a set of descriptive captions across the regions of image; then a FCLN to address dense captioning composition; dense localization layer to differentiate and to be inserted into

any neural network that processes images to enable region-level training and predictions. These model targets work based on object detection, image captioning, soft spatial attention, etc. [3] For the **object detection**: the core visual processing module is based on *Convolutional Neural Network* (CNN) which in itself very powerful model in the context of visual recognition task. The dense prediction is performed using *Region-Convolutional Neural Network* (R-CNN) where each *Region Of Interest* (ROI) is processed separately. The advancement can be discussed on the basis of ROI where focuses on processing all regions with a single forward pass of the CNN and it eliminates not-so-useful region proposals to predict the suitable bounding boxes either in the image coordinate system or in a fully convolutional. This approach is based on *Region Proposal Network* (RPN) which regresses from the anchors to regions of interest. For the next segment of **image captioning**: for the image captioning the *Recurrent Neural Network* (RNN) is adopted as the core architectural model to generate the captions as this model is an ideal for language modelling which offers long-term interactions, which is somewhat integrated with the language model combination with RNN on image information known as soft attention mechanism over regions of the input image with every generated word. The **soft spatial attention**: it is garnered generally in the network which processes arbitrary affine regions in the image instead of discrete grid positions. The major difference [15] the DenseCap has from the traditional approaches is, the DenseCap has an open set of targets, not limited to any valid objects and it includes parts of objects and multi-object interactions. And this difference support an issue, the target bounding boxes become much denser than object detection with limited categories; and having the huge number of visual concepts described making the target regions visually ambiguous without the information about the context. And these issues can be solved using the **joint inference** and **context fusion**.

1.5 Research Hypothesis

In this research, we have attempted to answer the following research questions.

1) Can the visual relationship between objects be captured using only feature vectors without explicitly providing object bounding boxes? (i.e. no annotations)

2) Can we use transfer learning to use CNN weights learned for one task and fine-tune them to solve a slightly different task?

3) How do the captions generated from our model compares to the baseline model?

4) How to deal with processing and training very large deep learning models in case of limited hardware resources?

1.6 Further outline

The following chapters represent the methodologies used in running the models on the open image datasets and which challenges were faced at our end and the result we came up with.

2. Explanation of model architectures of Show, Tell & DenseCap

2.1 Show, Tell

The overall architecture for this model is a CNN and LSTM RNN combined together. Specifics of the CNN used in this approach were not revealed in details. It was generally mentioned and referred to their choice of CNN with a novel batch normalization and best performer on the ILSVRC 2014 classification competition.

LSTM, a particular form of recurrent nets, was used in this model same as the ones which had been introduced and applied already with great success to translation and sequence generation. Using gates in this model not only provides memory to learn efficiently, but also deal well with the vanishing and exploding gradients.

2.1.1 Loss function

As the show and tell model aims at maximizing directly the probability of the correct description given the image by using the below formulation, its negative can be considered as the loss function.

$$\theta^* = arg\max_{\theta} \sum_{(I,S)} logp(S|I;\theta) \quad (1)$$

2.2 DenseCap

While deploying this model, we learnt that the challenge was to develop a model that supports end-to-end training with a single step of optimization and both efficient and effective inference. The DenseCap model uses the VGG-16 **CNN** architecture which performs on its state-of-the-art function. This model has 13 layers of 3 x 3 convolutions interspersed with 5 layers of 2 x 2 max pooling. Now, here's the catch, to increase the tensor of features of the shape of an input image, final pooling layer has been truncated using the shape 3 x W x H which gives C x W' x H' where C = 512, W' = $\left\lfloor \frac{W}{16} \right\rfloor$, and H' = $\left\lfloor \frac{H}{16} \right\rfloor$. The output we get is an image from this network encoded at the set of uniformly sampled images locations and forms the input to the localization layer. Now the next layer, which is the **localization layer** receives an input tensor of activations, identifies spatial regions and extracts features based on fixed-sized components from each region, which is based on Faster R-CNN. Now there is a need to use bilinear interpolation which replaces ROI pooling mechanism which helps gradients in back propagation over predicted regions. The localization layer accepts C x W' x H' sized tensor from activation and returns three different regions such as, region coordinates, region scores and region feature.

Having Faster R-CNN into account and similar to that **convolutional anchors** are retrieved based on the localization layers which predicts region proposed by the regression offset from a set of translation invariant anchors. The adoption of parameterization, regressed from the anchors to the region proposals, having the anchor box with the centre, which is known as the **box regression**. Processing the typical image of any size and anchor boxes results into several region proposals. And having it running the recognition network and the language model for all the proposals would end up expensive, therefore there is a need of subsampling them using **box sampling**. A region is positive if the *Intersection Over Union* (IOU) having at least 0.7 with some ground-truth region and if IOU < 0.3 throughout the regions considered negative region during the training time. After the sampling process is achieved, there are various region proposals of varying sizes and aspect ratios resulting in **bilinear interpolation**. And to solve this Faster R-CNN is used with ROI pooling layer. **ROI pooling layer** is a function of two inputs: convolutional features and region proposal coordinates. The sampling grid is a linear function of the proposal coordinates so gradients will be back propagated on the predicted region proposal coordinates using bilinear interpolation to extract features for all sampled regions to form the final output out of the localization layer known as **bilinear sampling**. Further, the recognition network is a fully-connected neural network to process the region features after the localization layer. The regions we get are flattened and passed

from two fully connected layers as vectors using *Rectified Linear Units* (ReLU) and they are regularized using **dropout**. This process also refines the confidence and position of each proposed regions known as **recognition network**. Then the regions are fed as region codes to the conditioned **RNN language model**.

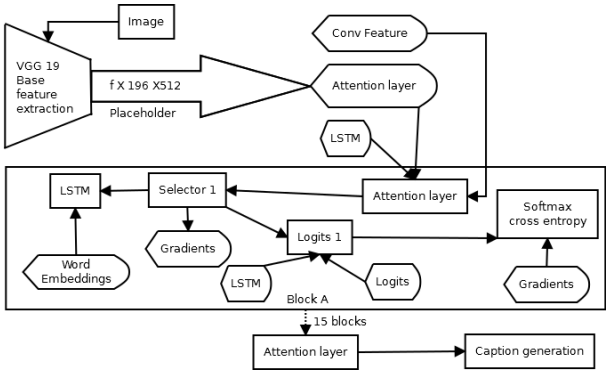
For the **Dense Captioning** segment, the model is fed with a single image with the produced output in terms of a set of regions, annotated with a confidence and a caption. The **evaluation metrics** produces well-localized predictions and accurate descriptions forming dense captions. As an evaluation metrics and to the measurement of the Mean *Average Precision* (AP) for localization and language accuracy over a whole range of object detection and image captions. For language modelling, METEOR score is used because it is highly correlated with the human judgements.

2.2.1 Loss function

For the DenseCap, the training duration consists of positive boxes and descriptions and the model predicts position and confidences of sampled regions. Then, the predictions made are conducted in the localization layer and again in the recognition network. The normalization is done based on all the loss functions (the smooth L1 loss for the box regression, cross entropy term at every time-step of the language model in the loss function, binary logistic losses for the trained confidences over positive and negative regions, etc.) by its batch size and the sequential length in the RNN stage.

3. Our models

For our experiment, we have used the model architecture proposed in [1]. The entire training process is described in detail below.



3.1 Base Feature Extractor

For the base feature extractor, we have used VGG19 CNN model architecture. This model is trained on MS COCO 2014 dataset consisting of 1 million training images. We have taken conv5_3-layer output which is passed to the next lstm layers as the feature vector.

3.2 LSTM and Attention Layer

After the feature extractor, we have a block consisting of an attention layer, a selector and a lstm layer. We have 15 consecutive layers containing this block.

3.3 Output layer

Finally, we have the final attention layer which assigns attentions to different regions of the images based on which the caption is generated. We have used Softmax Cross Entropy with logits to map probability to each word.

3.4 Data pre-processing

Preparing the dataset was the most challenging part of our work. In the Kaggle dataset, the training set in excel file contains samples in the following form,

Image_id, object1_class, object2_class, object1[xmin,ymin,xmax,ymax], object2[xmin,ymin,xmax,ymax], predicate

However, the Show and Tell requires JSON files containing samples in the following form,

Index, caption, image_id

Therefore, we have transformed the Kaggle dataset to the format readable by our model. To generate the *caption* field, we have used the following formula,

Object1_class + predicate + object2_class

Also, some of the images in the dataset were corrupted, and some samples had repetitive captions for many times. Thus, we had to clean the dataset to make it usable.

Apart from that, all the images were resized to 224x224 to obtain a fixed input size for our input layer.

3.5 Training and Testing samples

Although, the training set in Kaggle has 1M samples, due to hardware limitation, we could not use all of them to train our model. Thus, to overcome memory error, we used 8000 samples for training.

Similarly, out of 90 thousand images from the test dataset, we used only 1000 of them for validation and 1000 of them for testing.

3.6. Training Procedure

We arranged limited access to a windows computer with NVIDIA GTX 1070i GPU and 32GB RAM. Thus, we used the GPU only for training the model and another computer with core i5 CPU to evaluate the model. We tried out different configurations of hyperparameters. We trained the model from scratch as well as from fine-tuning on a pre-trained model. The pretrained model is publicly available in [github.com/KranthiGV/Pretrained-Show-and-Tell-model].

Due to hardware limitation we could not follow the above steps for DenseCap model.

3.7 Evaluation

We evaluated each model on the same test dataset. For each image we calculated the bleu score using the generated caption and the true caption. Finally, we took the average of all 1000 images and compared different models.

4. Hyperparameter tuning

In Show and tell model, we tried with different learning rates ranged from 0.0001 to 0.1 with interval of 0.1. The best bleu score we reported is obtained by learning rate of 0.01. Another hyperparameter we tried is the batch size. Initially we started with batch size of 128 and ended up with 12. With a larger batch size, the training time was faster however the model accuracy was degraded drastically. In addition to that, according to our observation, with larger batch size the model didn't generalize well and overfitted to the training set because there was a large gap between the training blue score and testing bleu score. On the other hand, smaller batch size increases the training time but the model accuracy is improved. As example, with batch size of 128, we got bleu score of only 52% whereas with size of 12 we got 60% with all other configurations being unchanged.

As for the loss function, we tried with both AdamOptimizer, AdaDelta and AdaGrad. According to our experiment, changing the loss function didn't have much effect on the model accuracy. The reported results are obtained with AdamOptimizer. Apart from that, we also tried different value for the beam size k randomly chosen from 1 to 20. The best result is obtained with beam size of 3.

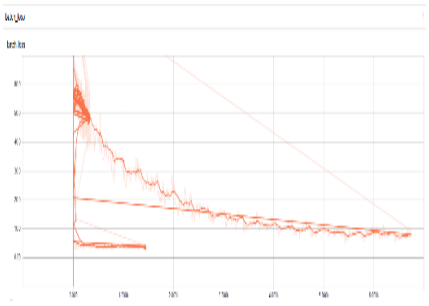
5. Adding dropout, batch norm, regularization

For Show and Tell model, we used dropout with keep_probability of 0.8 which means that 20% of the neurons in each layer are randomly nullified to zero. We have also used batch normalization. Without batch normalization, our average bleu score was 57% whereas after using batch normalization it has been increased to 60%. This empirical observation aligns with the result reported by the authors of Show and Tell paper, where they claimed that use of batch normalization increased their model bleu score by couple of points.

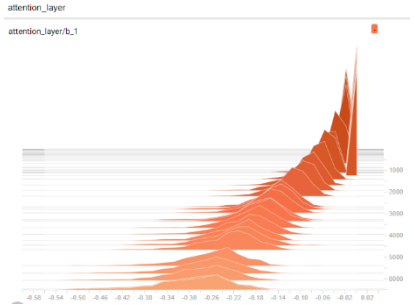
We used doubly stochastic regularization with alpha value of 1.0. This regularization is added in the attention layers.

6. Visualization of model weights in tensorboard

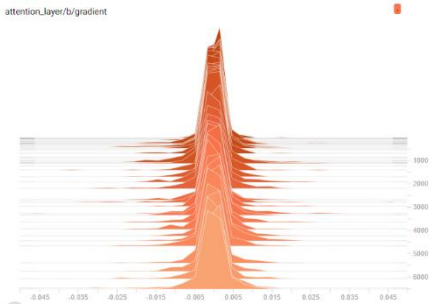
We have done real-time debugging of the training by visualizing the event logs in tensorboard. We have checked the batch loss curve, histogram of attention layer weights and lstm layer weights and biases. According to figure 1, the batch loss is gradually decreasing with no. of epochs.



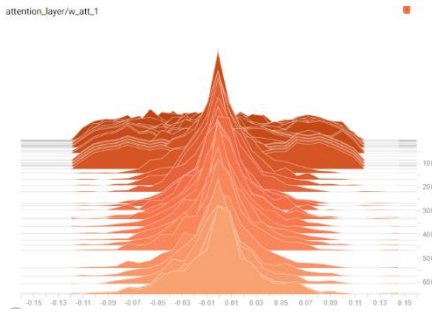
1



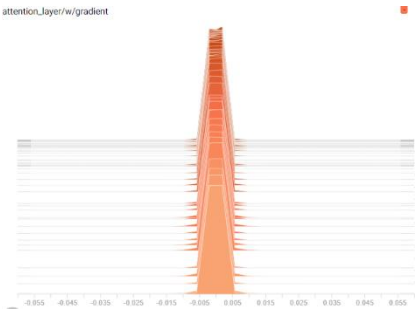
2A



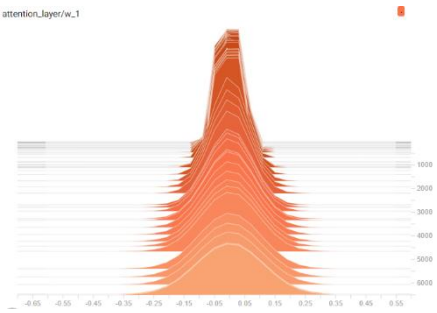
2B



2C



2D



2E

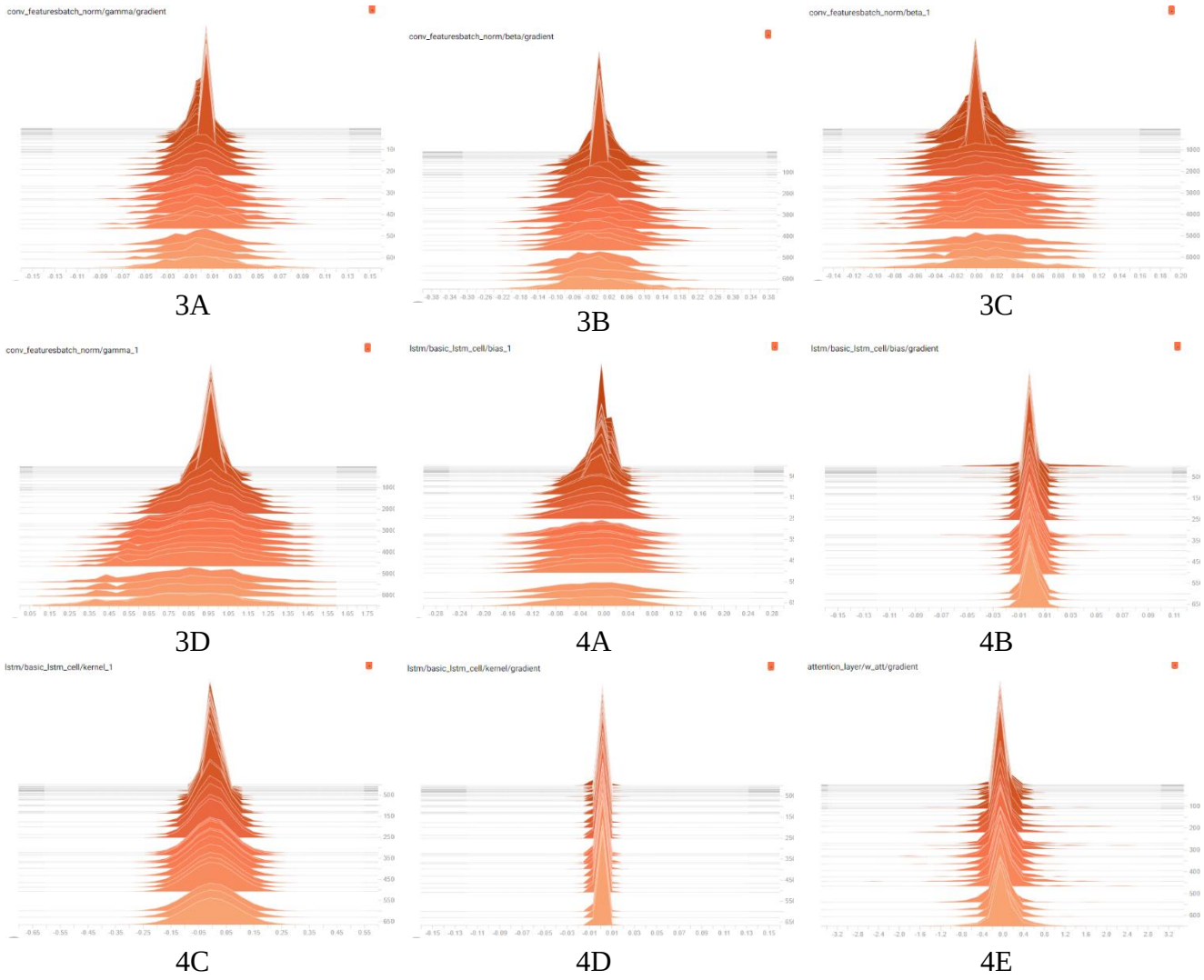


Figure 1 Image 1 is for loss curve. Figure 2A to 2E are attention layers weights. Figure 3A to 3D are Conv feature weights. Figure 4A to 4E are for LSTM weights

7. Transfer learning

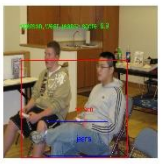
As mentioned earlier, we have used transfer learning in two stages. One is for the weights of the base feature extractor learned from MS COCO 2014 dataset and another one is the entire NIC model with 1 million iterations which is publicly available.

8. Comparison of our results with baseline models

We compared our result with the baseline model described in [13] and also with the true captions. As mentioned earlier, due to hardware limitation we could not train the DenseCap model. Thus, we used this model just for evaluation on the Kaggle dataset using their pre-trained model. Image 1 is an output generated by Visual Relationship Detection (VD) model.

Image id	True Caption	DenseCap	Our Model	VD
1	Guitar is Wood	green shirt on the man	Chair is wood	Man plays guitar

2	Bench is Wood	the man is wearing a black shirt	Desk is Wood	Man holds pipe
3	Man holds Microphone	man wearing black shirt holding a cell phone	Man with cell	Man holds microphone
4	Boy hits Football	boy is holding a frisbee	Boy with ball	Boy plays_ with football
5	Man holds Camera	man holding a camera	Man with camera	Man holds camera



1

2

3

4

5

9. Discussion

According to the result, we see that the baseline VD model gives the best performance in terms of finding the right relationship. However, our model also gives some competitive output. Training with larger number of samples should improve the model even more.

10. Limitation

There are few limitations which we have faced so far like, training the models, we need to have GPU, which is not available on the CS server so we contacted CS server administration, and they loaded a virtual server for our group to do the training (new server space: *ml.cs.uwindsor.ca*) and on this server we had *sudo* privileges and as a result we were able to install the libraries like *torch*, images, which requires 1 TB+ space, but we had space constraints on the ML server, so we only downloaded some 40K+ images. For training this model, we still required GPU, and one of our group members has an access in the Physics department, therefore we used that computer with GPU for training purpose.

11. Conclusion

Based on our experimentations, we have successfully answered to our initial research hypothesises and had hands-on experience with experimenting with large dataset and very large model. Our goal was to create a model which can easily capture visual relationship between objects based on the existing architecture for image captioning using the method of visualizing the relationship between the objects using the predicates. We did this with our model based on the Show and Tell.

12. Future work

We would like to train the DenseCap model with the Kaggle dataset and verify the model performance. Also, we would like to further modify the model architecture and change the base feature extractor to improve the model accuracy.

References

[1] K. Xu, J. L. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. S. Zemel and Y. Bengio, “Show, Attend and Tell: Neural Image Caption Generation with Visual Attention,” vol. 1502, no. 03044v3, 2016.

[2] J. Johnson, A. Karpathy and L. Fei-Fei, “DenseCap: Fully convolutional localization networks for dense captioning,” vol. 1511, no. 07571v1,

2015.

[3] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," 2014.

[4] O. Vinyals, A. Toshev, S. Bengio and D. Erham, "Show and Tell: Lessons learned from the 2015 MSCOCO image captioning challenge," in *IEEE transaction on pattern analysis and machine intelligence*, 2016.

[5] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, T. Duerig and V. Ferrari, "The open image dataset V4: Unified image classification, object detection, and visual relationship detection at scale," *IJCV*, vol. 1811, no. 00982v1, 2018.

[6] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier and D. Forsyth, "Every picture tells a story: Generating sentences from images," *ECCV*, 2010.

[7] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg and T. L. Berg, "Baby talk: Understanding and generating simple image descriptions," *CVPR*, 2011.

[8] D. Bahdanau, K. Cho and Y. Bengio, "Neural machine translation by jointly learning to align and translate," vol. 1409, no. 0473, 2014.

[9] K. Cho, B. V. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *EMNLP*, 2014.

[10] I. Sutskever, O. Vinyals and Q. V. Le, "Sequence to sequence learning with neural networks," *NIPS*, 2014.

[11] "Overview of the Open Images Challenge 2018," Google APIs, 2018. [Online]. Available: https://storage.googleapis.com/openimages/web/challenge.html. [Accessed 27 November 2018].

[12] "Google AI Open Images - Visual Relationship Track," Kaggle - Google AI, 2018. [Online]. Available: https://www.kaggle.com/c/google-ai-open-images-visual-relationship-track. [Accessed 27 November 2018].

[13] C. Lu, R. Krishna, M. Bernstein and L. Fei-Fei, "Visual relationship detection with language priors," vol. 1608, no. 00187v1, 31 July 2016.

[14] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," vol. 1312, no. 6229, 2013.

[15] L. Yang, K. Tang, J. Yang and L.-J. Li, "Dense captioning with joint inference and visual context," vol. 1611, no. 06949v2, 2017.

[i] Represents equal contribution by all MSc students in alphabetical order