Project pre-proposal

Project title: Real-time object detection in videos
Team number: 8 / Dig Vijay Kumar Yarlagadda (Class ID: 26)

Project goal and Objectives: -

Introduction:
Tensorflow provides many powerful libraries for object detection and segmentation in images. In this increment, I tried to implement one such advancement in deep learning, The Inception Architecture, for detection and segmentation of objects in images and generating captions to explain objects in images. For this increment, I tried to use image frames extracted from videos (for testing, training is done on large datasets).

Objectives:

- Design a real-time video analysis system using deep learning framework Tensorflow.
- Automatically detect objects in video and segment them. The system will be pre-trained on training datasets and a model is generated. Based on available computing resources the video is divided into 30/60 frames/second and objects will be classified/detected and segmented.
- Automatically generate captions for currently displayed frames. It can be helpful in assisting blind people.

Approach:

Data sources:

- Microsoft COCO image caption generation, object segmentation and dataset for training the inception architecture.
- Flickr30k image Dataset for training and testing the generated image captions.

Tools:
No APIs are used in designing the system.

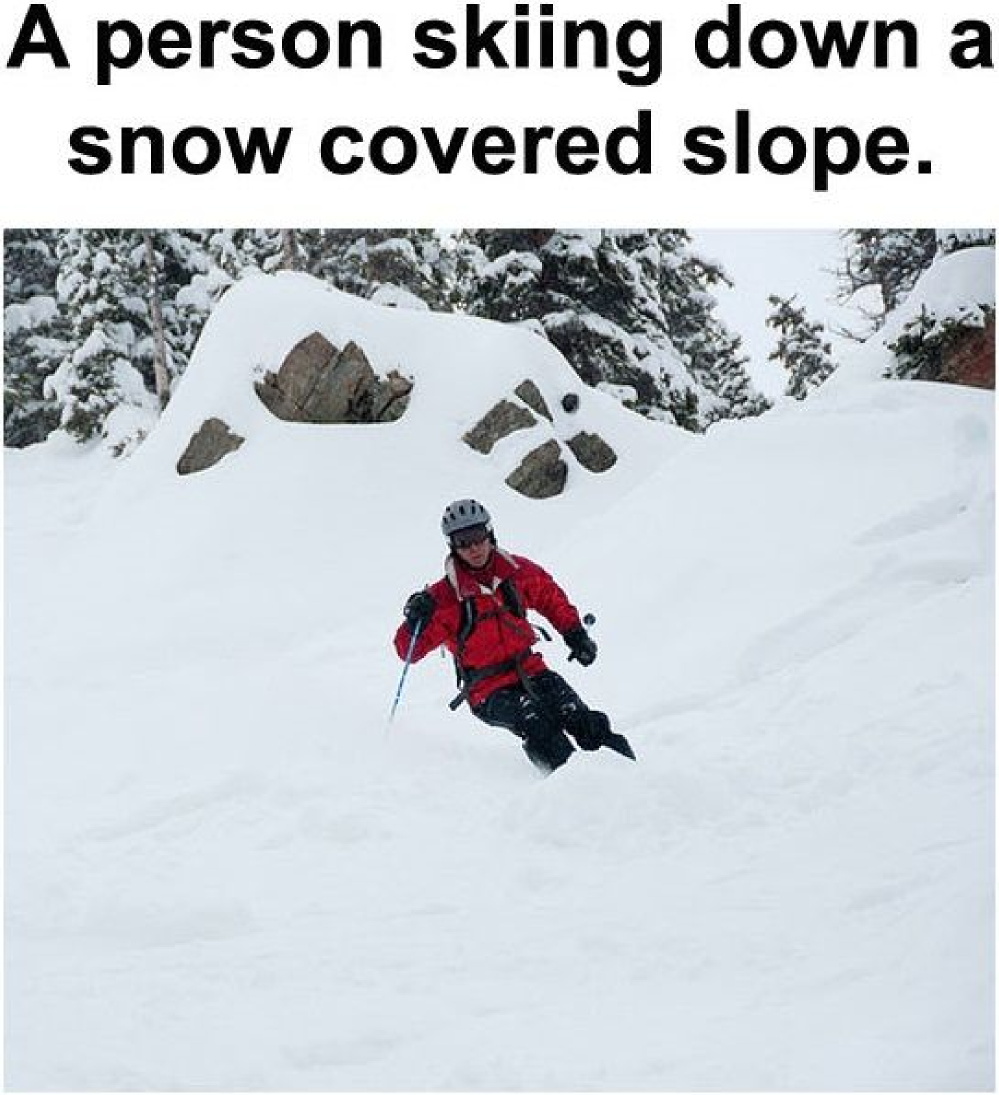
- Tensorflow
- Google Protobuf
- NumPy
- Keras
- OpenCV
- scikit-image

Expected Inputs/Outputs:

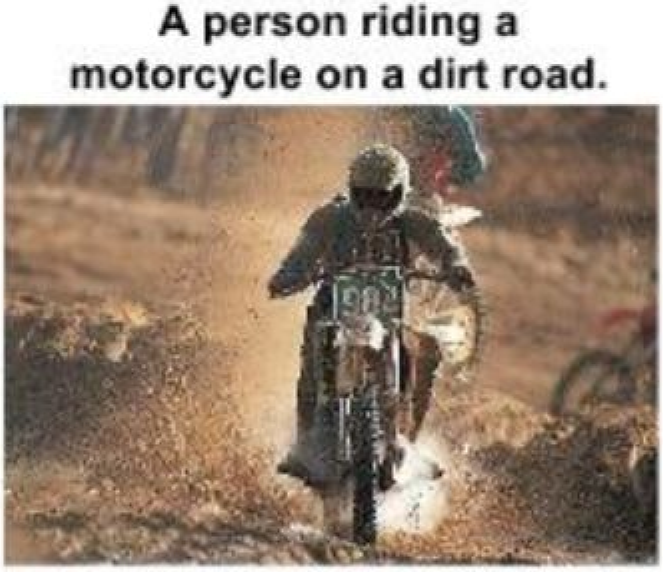
Inputs: Any image/video frame (resolution tested: 640 by 480)

(Image Credit: Google Inc., https://raw.githubusercontent.com/tensorflow/models/master/im2txt/g3doc/example_captions.jpg)

Output: Object in image is detected and an image caption is generated which explains what is present in the video.



(Image Credit: Google Inc., https://raw.githubusercontent.com/tensorflow/models/master/im2txt/g3doc/example_captions.jpg)



(image credit: http://arxiv.org/pdf/1609.06647v1.pdf)

Algorithms:

- Inception architecture uses multi-layers neural networks, my implementation in particular uses RNNs with LSTMs for achieving good accuracy.

Related Work:

Object detection and image captioning is a heavily researched topic in the field of deep vision. The closest work to Inception V4 is Microsoft RESNET which emphasizes the importance of building deeper networks for achieving state-of-art accuracy. Many approaches such as Imaginet, Show and Tell, Deep visual semantic approach and Attention-based Encoder–Decoder Networks are proposed but they are not adopted for video captioning and very limited in applications.

Application Specification:

System Architecture:

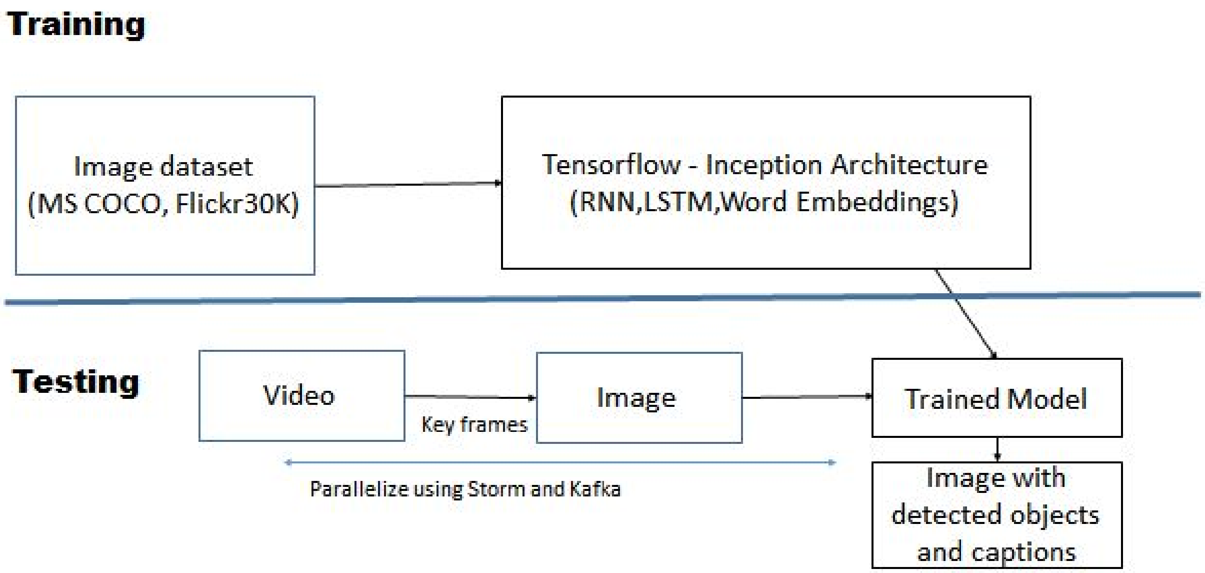

Training
Image dataset (MS COCO, Flickr30K)
Tensorflow - Inception Architecture (RNN,LSTM,Word Embeddings)
Testing
Video
Key frames
Image
Trained Model
Image with detected objects and captions
Parallelize using Storm and Kafka

Fig. Initial proposed Architecture

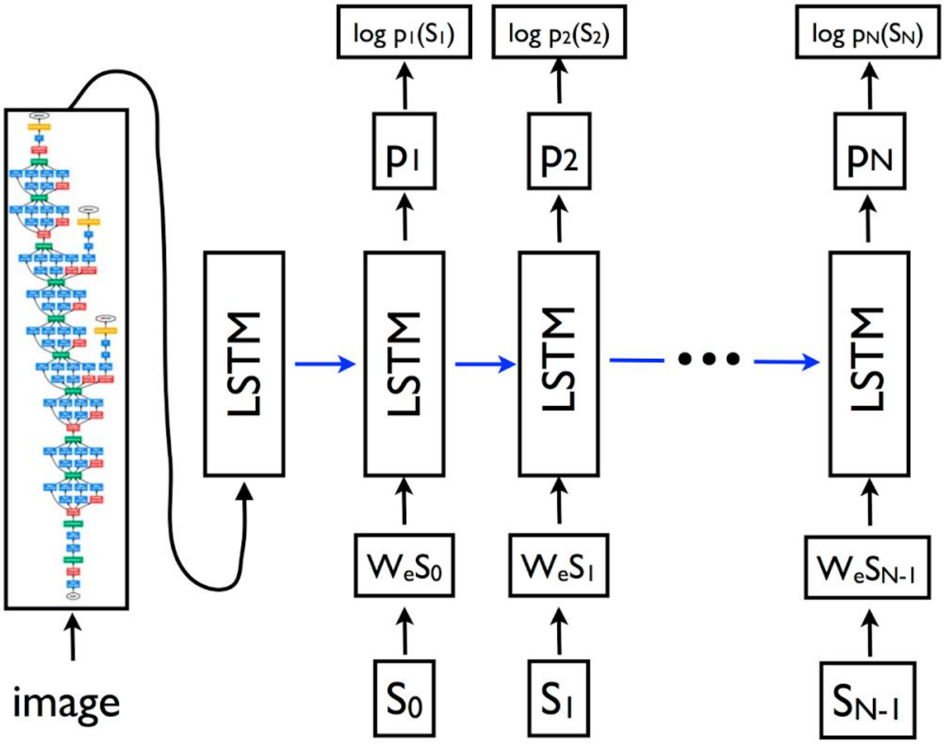


Fig. Internal system of Image captioning system based on Inception Architecture (credit: http://arxiv.org/pdf/1609.06647v1.pdf)

Project Management:

Work completed: Designing a system which can take images as input and generate 3 top captions which can precisely describe the image. The network is tuned with different hyperparameters,

Time Taken: >100 hours. Everything has to be configured again for this project, Tensorflow, GPU and all the libraries specifically required for the project, they took lot of time. Also, training even on top of small portion of Flicker30K dataset took significant GPU compute power and lot of effort is put into optimizing this system to work with training on small datasets.

Work to be completed: While image captioning is done after recognizing objects in images, my system in present state doesn't show any kind of outline/bounding box for detected objects. Also, the system in its present state only accepts single images, so parallelization aspect of the system has to be designed using Storm and Kafka.

Estimated time to take: >250 hours

Issues/concerns:

- Lack of enough GPU computation power. Google reported, to train on top of entire training set with a single NVIDIA Tesla K20m GPU took them 2 weeks. For my implementation, I trained only on a small portion of Flicker30K dataset, the results are acceptable, but to achieve state-of-art accuracy, more GPU power is required.

Project GitHub URL:

https://github.com/digvijayky/Tensorflow-real-time-video-analysis

Bibliography:

[1]. Vinyals, Oriol, et al. "Show and Tell: Lessons learned from the 2015 MSCOCO Image Captioning Challenge." (2016).

[2]. He, Kaiming, et al. "Deep residual learning for image recognition." *arXiv preprint arXiv:1512.03385* (2015).

[3]. https://github.com/tensorflow/tensorflow

[4]. Donahue, Jeffrey, et al. "Long-term recurrent convolutional networks for visual recognition and description." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

[5]. Cho, Kyunghyun, Aaron Courville, and Yoshua Bengio. "Describing multimedia content using attention-based encoder-decoder networks." *IEEE Transactions on Multimedia* 17.11 (2015): 1875-1886.

[6]. https://research.googleblog.com/

[7]. https://github.com/tensorflow/models/tree/master/im2txt#a-note-on-hardware-and-training-time

[8]. http://mscoco.org/

[10]. http://shannon.cs.illinois.edu/DenotationGraph/