International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

Effective Image Captioning With YOLO and LSTM

Pooja R.Negi, Sanjay H.Buch

[1]*Pooja R. Negi, College Of Computer Applications, BhagwanMahavir University, Gujarat, India*
[2]*Sanjay H.Buch , College Of Computer Applications, BhagwanMahavir University, Gujarat, India*

Date of Submission: 01-02-2023 Date of Acceptance: 10-02-2023

ABSTRACT: Some of the enduring problems in artificial intelligence are computer vision and natural language processing. In this study, we investigate a model for generative automatic image annotation that makes use of recent developments on both counts. In our method, picture regions are detected using a deep convolution neural network and afterwards fed into a recurrent neural network that has been trained to increase the likelihood that the given image will be described by the target sentence. During our testing, we discovered that when our detection model's image representation is combined with the input word embedding, better accuracy and training are obtained. We also discovered that the majority of the information from the detection model's final layer disappears when it is used as a thought vector for our LSTM.

KEYWORDS: Computer vision, Natural language processing, Image captioning, LSTM.

I. INTRODUCTION

Automatic picture annotation, automatic image tagging, or captioning are terms used to describe the process by which a computer system automatically assigns a caption metadata. A typical image annotation system considers two crucial elements: a natural language processing unit to translate the semantic information included in the image into output that is intelligible by humans; and a semantic understanding of digital images. An automated image annotation system is one of the systems that significantly contributes to image retrieval systems by automatically generating image metadata that is afterwards indexed for searching. When compared to CBIR systems, one of the main benefits of automatic image annotation systems is that users have greater freedom and can more naturally specify queries.

There has been several alternative automatic picture annotation system architectures proposed thus far, using a variety of methods. Statistical models [1], Text similarity [2], support vector machines [3, 4],maximum entropy [5], and linguistic indexing models [6, 7] and other techniques are a few of the techniques employed. However, a significant performance improvement in computer vision and language processing wasn't made until the introduction of deep learning.

The automatic image annotations system has advanced further as a result of significant advancements in both computer vision and natural language processing. At the time this paper was being written, Google's NIC (neural image caption generator) model was the state of the art [7]. This model employs a deep convolutional neural network (InseptionV3) for image classification and an encoderdecoder recurrent neural network (RNN) to produce a description for the supplied image. A bi-directional recurrent neural network object detection model is used in yet another outstanding Stanford study [8]. Our model is based on a thorough knowledge of both models. For a more successful outcome, we also took into account the strengths and weaknesses of various automatic picture captioning models and their sub models, including other potential sub models [9–11].The goal of the study is to better understand how humans can describe images in a variety of ways. One option is to completely describe a situation, while the othera part of it. An important characteristic for deep semantic comprehension is the ability to recognize an object and its relationship to an adjacent object and produce a description out of it. Since multiple detections and sub regions can be applied to the language model to produce localized descriptions out of the entire, utilizing an object detection model as an initial module for the automatic annotation makes more sense to achieve this goal. In order to create a quick captioning system, we therefore intend to use a faster detection model that can recognize objects in real-time and an encoder-decoder language model.

Understanding and learning the inter-object relationships of the many objects in the

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

given image presents the biggest challenge for our approach. Working with deep learning, however, presents a variety of difficulties, including a steep learning curve, data preprocessing, an architecture flow that requires fine tweaking, and a high computing requirement.

The ability to generate an image description and display the items discovered is made possible by using a single pipeline for bounding box prediction and classification. We can create a strong model that comprehends the latent association between identified objects, their spatial location, and the relationship between these objects by including this final result into our language producing network.

II. MODEL DEVELOPMENT AND IMPLEMENTATION

The proposed model is built to automatically create image annotation for the provided photos by combining a detection model with a natural language model (encoder and decoder).In order to build this innovative architecture, which combines the most advanced rapid detection model and the most advanced Language Translation Architecture, we conducted a thorough analysis of other object recognition models and automatic picture captioning systems.

This architecture is based on the reliable YOLO detection model, which approaches detection as a straightforward regression problem as opposed to classification over a broad variety of locations and scales like the earlier models did. Apart from the tremendous performance advantage it brought about, this architectural modification in object detection also paved the path for further future advancements in several other domains.

In contrast to earlier models, the YOLO model views detection as regression, making the bounding box a learnable parameter as well. The YOLO's final output consists of the consequently, a single tensor will be encoding both the class probability and bounding box information for each grid cell. This makes it simpler to pass the class and their relative positions into another network for additional processing because they are represented as a single tensor output.

S * S* (B*5+C) is the final output Tensor YOLO model. Where the image is partitioned into a S*S grid, B–numbers of bounding boxes are predicted for each grid cell, and C–numbers represent class probabilities.Here, a grid cell is in charge of detecting an object if its centre does so within that grid cell. This extensive unsorted image information on the locations of detected objects with their corresponding confidence level is contained in this dense tensor.

YOLO consists of two fully connected layers. Our model uses the fully connected layer's final output from the YOLO network as the initial state of our LSTM network.

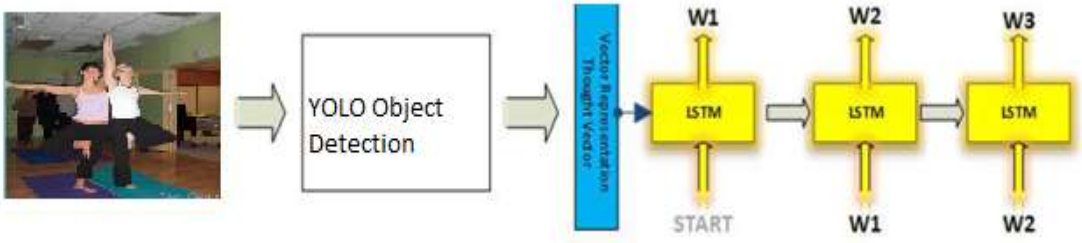

Fig.1 General Flow of System.

Object Detection Model

A CNN can be thought of as a feed-forward neural network that can extract an image's topological information. Only now have convolution neural network become widely used in computer vision applications. In the recent years, CNNs have demonstrated cutting-edge performance in a wide range of challenging computer vision tasks, including, among others, image classification [12], object recognition [11, 13], fine-grained categorization, image segmentation, posture estimation, and OCR in real-world images. The CNNs are often trained from beginning to end in these works and perform noticeably better than systems that rely on precisely designed representations, like SIFT or HOG features. This accomplishment can be partly ascribed to CNNs' inherent invariance to local picture alterations, which supports their capacity to learn hierarchical data abstractions.We apply a 24-layer convolutional model with two completely linked layers called the YOLO small model. We are interested in the output of the final FC layer. The form of the final layer tensor is 7*7*30. Our LSTM Network uses this flattened tensor as its initial state, making it a 1470-long vector.

The bounding box regression network's output, this Tensor, provides the class probabilities

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

of the observed items and their spatial positions. This network will provide an accurate, granular classification of the image's areas and their geographical locations. This intern deduces the internal connections between the identified objects. Our main finding is the use of the class probability and bounding box information vector, which informs the language model of all the important objects found in the model and their relative placements. This information can give the language model clear, high-level information.

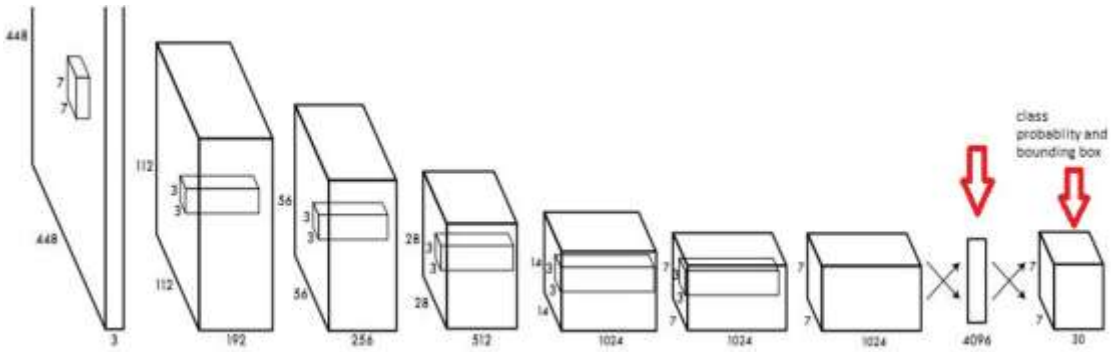

Fig.2 Fuly connected layers of model.

Language Model

Recurrent neural networks (RNNs) are used by our language model to create caption sentences for the inputted images. It is a "recurrent neural network"a style of artificial neural network in which directed cycles are formed by connections between units. These networks develop internal states that enable them to develop dynamic temporal memories.

The fundamental RNN models experience gradients that vanish or explode. It is a phenomenon that the gradient tends to get smaller as we go back in time or via sequences. Exploding gradient problems, on the other hand, have the exact opposite effect on the gradient. In the latter scenario, the gradient will increase exponentially as time passes. The primary issue, it should be noted, is not a disappearing or exploding gradient, but rather an unstable gradient that affects neural networks. LSTM (Long-Short Term Memory) and GRU are two models that have been suggested as solutions to this issue (Gated Recurrent Unit). In compared to the conventional technique, both LSTM and GRU have demonstrated to produce greater performance [14, 15].

The following formula can be used to express the LSTM Cell Block.

$$i_l = \sigma(W_{ix}X_l + W_{im}M_{l-1} + b_i)$$
$$f_l = \sigma(W_{fx}X_l + W_{fm}M_{l-1} + b_f)$$
$$O_l = \sigma(W_{ox}X_l + W_{om}M_{l-1} + b_o)$$
$$C_l = f_l \odot C_{l-1} + i_l \odot h(W_{cx}X_l + W_{cm}M_{l-1} + b_c)$$
$$M_l = O_l \odot h(C_l)$$

where C_1 denotes the cell state of the LSTM network and i_1, f_1, O_1, respectively, indicate the input, forget, and output gets of LSTM cell l. The symbols for a sigmoid and tanh function areσ(.) and h(.). The procedure is an element-by-element multiplication. The LSTM cell's weights, W_{ix}, W_x, and W_{ox}, are x for the input gate, forget gate, and output gate, respectively. The equivalent weights for the hidden vector and biases of the LSTM cell, however, are W_{im};,W_{fm}, W_{om} and b_i, b_f; b_o. The weights of the input x and the prior state M_{i-1} are W_{ex} and W_{em}, respectively. Similar to that, b_c represents the bias attached to these weights.

Our work's language model is based on the sequence-to-sequence model, a recent achievement in the field of language translation. This sequence-to-sequence model is followed by the language model we utilized, which is comparable to the NIC model from Google [7]. However, the model uses image representations rather than language sequences.

Despite being effective models, deep neural networks are challenging to train on sequential data because they require a fixed input dimension. However, a simple application of

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

LSTM Networks can effectively handle the sequence-to-sequence problem. According to research on language translation from French to English, this model is sensitive to sequence order and performs better when the intended sequence is reversed [16]. In situations when the integration between input and output is known in advance, RNNs may readily map sequence-to-sequence inputs. However, it is unknown territory for RNN networks to be used in complex, non-monotonic connections. One RNN can be used to match the input to a fix sized vector, which can then be used to map the vector into the target sequence.

We made numerous architectural variations to the right end of the sequence to sequence model that we utilize in our image annotation model. In order to send an ideal representation of the image through our LSTM model, we made a number of architectural changes.

Our architecture contains the localization of the detected object and its classification in a single vector, which we believe allows it to better characterize an image for generating a thorough picture description when compared to preliminary image caption models.

However, employing this representation in its entirety will unnecessarily increase the amount of our vocabulary, pushing the size of our final FC layer above the limits of our processing power. In order to avoid this, we created a 10000 vocabulary size using the training and validation annotation dataset in conjunction with additional data corpora. From the Glove representation, we used the word to vectorize these words.

III. ARCHITECTURES

To generate visual descriptions, our models use high-level image representation. We experimented with various architectures based on this fundamental concept to attain a better and faster performance.

Class-Probabilities Bounding Box Instruction

For the right side of the Sequence to Sequence model, we used the class probabilities and bounding box vector as a thought vector in this instance. The line of reasoning leading up to Sequence model, the YOLO model's final output, will be a representational vector of our input image. The thinking vector in this instance has a dimension of 1470. The information about the given image's box regression and class probability are both contained in this vector.

This model is lacking in additional specific details, making it simple to forget the picture representation. This causes a significant loss. This model is lacking in additional specific details, making it simple to forget the picture representation. This causes a significant loss.

Input Binding With Transformation Layer

Theoretically, a variety of architectures can produce a sentence description from supplied vectors of word and image representation. One option is to feed the convolutional neural network's output into the LSTM neural network's first hidden layer. On the other hand, supplying the initial input with the sentence's first token. The model will advance by feeding the most recent forecast to the incoming data. However, this model will quickly forget the image's content as it moves on to guess the next phrase.

In this strategy, we utilized a single neural network layer (transformation layer) that converts the network's inputs to uniform dimension, in this case a 200-d vector, in an effort to try and prevent our network from forgetting the image. This indicates that the convolutional neural network's output will be converted into a 200-d vector and afterwards linked with the appropriate word input. Be aware that in order to obtain an output of 200-d vector representation, the embedding vectors of the vocabulary must likewise pass through their appropriate transformation layer. The picture representation can then be supplemented with these vectors. Keep in mind that the bounding box vector and picture class probability will be included in the detection's final output.

Unified Model

Our integrated model makes use of both of the aforementioned systems. In this approach, the final layer of the convolution feature map is used to initialise the thought vector rather than random initialization, which provides no information to the network. The recurrent neural network will then have access to both high-level picture representation and low-level representation (detail) representation, to which it can repeatedly refer.

Other Components

In order to determine the optimum picture annotation model given the image representation and word sequence as inputs, we have tested a variety of other architectural modifications. Implementation of model using a modified input containing the concatenation of word sequence, 1472-D image representation, and 4097-D image representation has demonstrated encouraging results. Each unique input is first placed through a transformation layer, which creates an output 200-dimensional vector, to create this concatenation.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

We employed 600 LSTM units with a 64 batch size and 20,000 vocabulary size on this design.

IV. IMPLEMENTATION DETAILS

Because our model is implemented using strictly object-oriented programming, we may create a variety of architectural modifications as it is being trained. We attempt to determine the most effective variation in each. As mentioned in the sections above, our model consists of two sub-modules that operate as one system. We used the YOLO object detection system for the computer vision sub-system in the first, and the probabilistic language model in the second. The well-known sequence-to-sequence model is used as the basis for the language model.

When we look at the architectures, the majority of their implementation is comparable. The input of the language model, the input of the input transformation layers, and many hyper-parameters, however, varied significantly. Additionally, there might be variations in the regularisation implementation and optimizers employed. We will talk about how the model is generally put into practise in this section.

Implementation

Our methodology is quite easy to use and put into practise. From the training dataset, high-level representations of images are first produced. These representations will be processed using the YOLO object identification model in conjunction with the corresponding sentence caption.

A simple implementation strategy is used to speed up training of a computerized image-recognition system, where the data is stored on a disc rather than a single memory card.These epochs were produced using batches of 64 and 128. The downloaded and processed COCO picture dataset is used to create these batch scripts.Here, the target sequence is a one-hot encoding of the caption sequence with the index shifted by one, while the input is a vector of word sequence created from the image description. We have experimented with several Image annotation model topologies using these foundational foundations.We will examine our input and anticipated output parameters in more detail below. Four multi-dimensional parameters are required for the training; they are as follows: Image representation, input and output sequences, and input mask are all included.

Hyper Parameters And Over Fitting

Different hyper-parameters were defined during implementation. Each of these variables has some bearing on how our model behaves. The definitions of these hyper-parameters are as follows: Initial learning rate of 2, optimization (Adagrad), 1470 LSTM units, 0.6 dropout rate, 50-sequence batch size, 128-units per batch, and 100-embedding size. Take note that the result reported below is connected to this hyper-parameter configuration. In addition, we have used many configurations with each architecture in an effort to achieve the best accuracy. We observed that raising the LSTM units above 1500 would lead to a number of issues, such as over fitting, excessive time and space complexity, and, above all, resource limitations. The GPU pool limit was continuously exceeded by the model.

In a different approach, we employed a transformation layer of 100 by 200 for word sequence embedding and a transformation layer of 1500 by 100 for the final FC-layer of the YOLO output.

V. DATASET AND EXPERIMENT

As a benchmark for our investigations, we used the coco dataset for image captioning. The dataset was preprocessed using the JSON annotation for picture caption coco dataset. As stated in the sections above, in order to increase training speed, we used a retrained YOLO model for object identification and trained our LSTM model for captioning. With a batch size, we created pairs of YOLO output tensor and caption input, and we dumped them on disc. This method significantly increased our training speed. YOLO can process 155 frames per second on average. However, loading the preprocessed output from file is much quicker; on the other hand, loading the batch from disc will also save having to process each image once again on each epoch. We first create batch files on the disc, then instantly create an input, targets, and mask based on the provided caption. This doesn't take up a lot of time, therefore it doesn't really affect how quickly you train. 50,000-dictionary size is what our model uses, as was already explained.

The top 40,000 words in the training caption data were used to create this lexicon. The next step is to extract 20,000 of these most common words from the intersection of the Glove word2vec embedding and the dictionary. After creating a dictionary out of this set, we used these 20,000 words and their vector form for training our model.

Tests Were Done

To reduce the loss of our model, we examined a variety of designs. We did not change the research's main tenet, which is to incorporate an

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

object detection system that incorporates the spatial location of the object into its learning parameters. Faster RCNN and YOLO are two major deep neural networks that can accomplish this. We chose YOLO since speed is our models' preferred important attribute, but this decision change wasn't made without trade-offs. In our situation, YOLO performs worse than other cutting-edge detection methods like Faster RCNN.

We applied the YOLO approach in our work. In order to implement our model, we experimented with various architectural variations on top of various deep neural network hyper-parameters. The first design incorporates the image-embedding vector as a thinking vector for the right side of the sequence-to-sequence model, following Google's work img2txt. The YOLO object detection system comprises two completely connected layers, it should be noted. These FC levels are 4095 and 1495 pixels wide, respectively. The box regression and class probabilities outcome of the object detection system are represented in the final FC layer, which is a 1495-d vector.

Outcomes

Although more than two architectural concepts were put forth in this study, only the outcomes of two structures are shown. The main causes of this are time and resource constraints.
The other architectures, however, are currently in development. We have specifically shown the mid progress of two of our architectures in this paper. We will also go over the key causes of our initial model's failure.

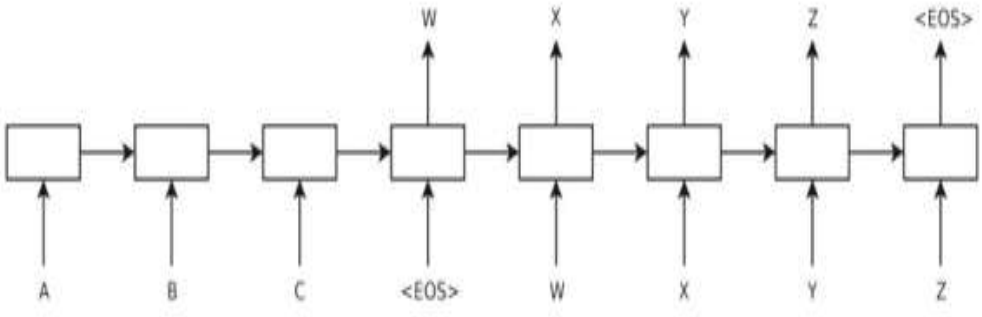

Fig.3 LSTM character sequence model

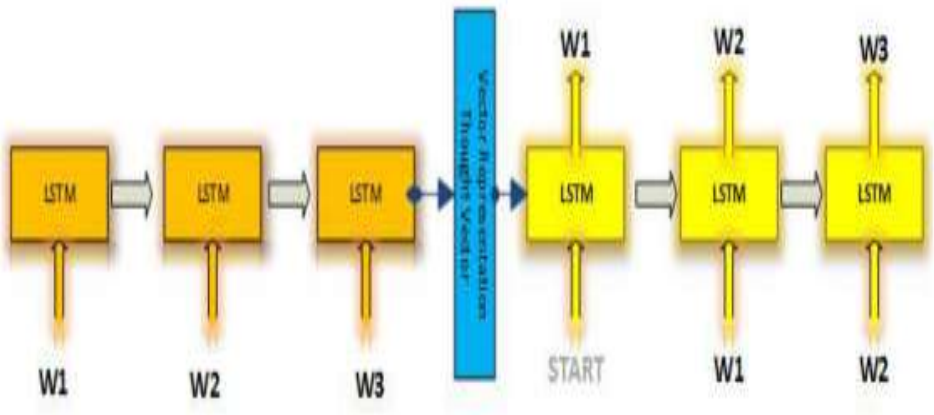

Fig.4 LSTM encoder decoder model

International Journal of Advances in Engineering and Management (IJAEM)
Volume 5, Issue 2 Feb. 2023, pp: 321-328 www.ijaem.net ISSN: 2395-5252

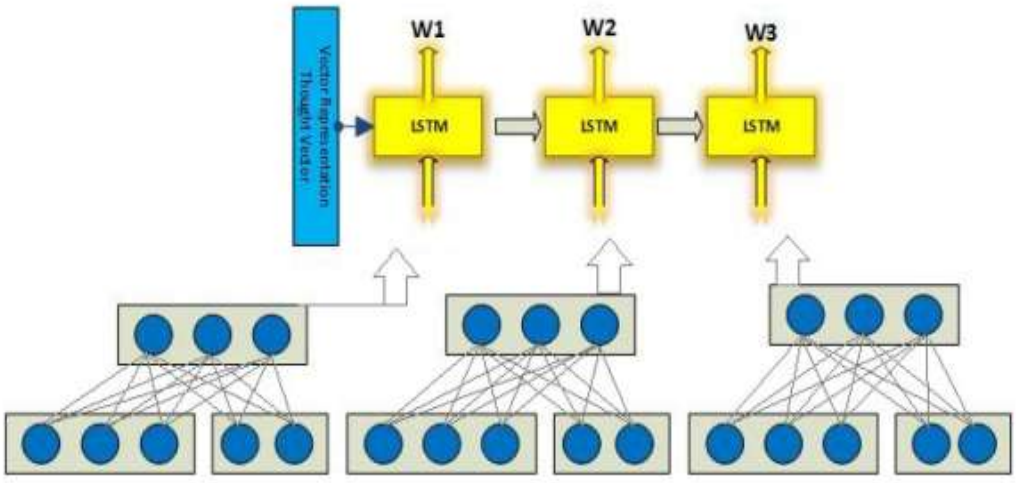

Fig.5 The time variation of the word imbedded in an image is fed to the LSTM time units by concatenating the image data with the word embedding. On this variation, we feed the last FC layer of the image detection model to each time step of our LSTM time units.

As a thought vector for our LSTM, our original model used the final fully connected layer of the YOLO-model (Figs-. 3, 4, 5). The LSTM in our model has 60 sequences, which provides a high level of input because our model's thought vector leads to high loss because the model forgets the picture reference as we move through the LSTM model's time steps. This had a highly negative accuracy as a result.

Table 1. Accuracy of different techniques on different datasets

Object detection model	Flickr8k	MSCOCO	People-Art-AP
YOLO-	0.81	0.75	45
R-CNN-	0.69	0.54	26

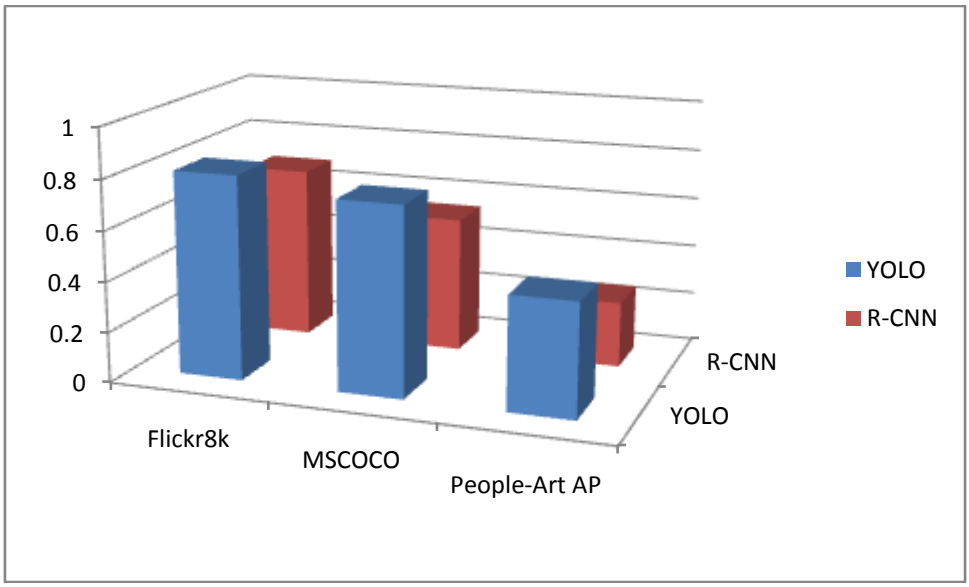

Fig.6 Analyzing data for YOLO and RNN

VI. CONCLUSION

As previously indicated, this work has not been fully trained, making it difficult to speculate on the model's overall correctness. Nonetheless, the current results appear promising, and this does not imply that we do not have a strong understanding of what functions and what does not.

Using class probability and bounding box as initial thought vectors without letting the LSTM model go back and review the image data results in subpar performance when generating text descriptions, according to our research. On the other hand, sending word embedding and image representation through a transformation neural network layer to the LSTM model and binding the results produces significantly better results, although this adds a training time lag.

We can fix this issue by lowering the many of hidden-layers in our LSTM to get results much more quickly. Anything over 500 layers is a good choice for this particular model. But we also need to be careful not to over-fit our model by add an extremeamount of hidden-layers.

REFERENCES

[1] Li, J., Wang, J.Z.: Real-time computerized annotation of pictures. IEEE Trans. Pattern Anal. Mach. Intell. 30(6), 985–1002 (2008)

[2] Picard, R.W., Minka, T.P.: Vision texture for annotation. Multimed. Syst. 3(1), 3–14 (1995)

[3] Cusano, C., Bicocca, M., Bicocca, V.: Image annotation using SVM. Proc. SPIE 1, 330–338 (2003)

[4] Tang, J., Lewis, P.H.: A study of quality issues for image autoannotation with the corel dataset. IEEE Trans. Circuits Syst. Video Technol. 17(3), 384–389 (2007)

[5] Jeon, J., Manmatha, R.: Using maximum entropy for automatic image annotation. Proc. CVIR Lect. Notes Comput. Sci. 3115, 24–32 (2004)

[6] Li, J., Wang, J.Z.: Automatic linguistic indexing of pictures by a statistical modeling approach. IEEE Trans. Pattern Anal. Mach. Intell. 25(9), 1075–1088 (2003)

[7] Vinyals, O., Toshev, A., Bengio, S., et al.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, 07–12 June, pp. 3156–3164 (2015)

[8] Karpathy, A., Li, F.F.: Deep visual-semantic alignments for generating image descriptions. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 07–12 June, pp. 3128–3137 (2015)

[9] Kulkarni, G., Premraj, V., Ordonez, V., et al.: Baby talk: understanding and generating simple image descriptions. IEEE Trans. Pattern Anal. Mach. Intell. 35(12), 2891–2903 (2013)

[10] Ren, S., He, K., Girshick, R., et al.: Faster R-CNN: towards realtime object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39(6), 1137–1149 (2017)

[11] Girshick, R., Donahue, .J, Darrell, T., et al.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)

[12] Szegedy, C., Ioffe, S., Vanhoucke, V.: Inception-v4, InceptionResNet and the impact of residual connections on learning. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, San Francisco, CA, 4–9 Feb, pp. 4278–4284 (2017)

[13] Redmon, J., Divvala, S., Girshick, R., et al.: You only look once: unified, real-time object detection. In: CVPR 2016, pp. 779–788 (2016)

[14] Karpathy, A., Johnson, J., Fei-Fei, L.: Visualizing and understanding recurrent networks. arXiv:1506.02078 (2015

[15] Chung, J., Gulcehre, C., Cho, K., et al.: Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv: 1412.3555 (2014)

[16] Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS, 2014, pp. 3104–3112 (2014)