Image Caption Generator with Novel Object Injection

Conference Paper · December 2018

DOI: 10.1109/DICTA.2018.8615810

Muhammad Ali Baig
NED University of Engineering and Technology, Karachi
7 PUBLICATIONS **34** CITATIONS
SEE PROFILE

Nauman Zafar
National University of Sciences and Technology
4 PUBLICATIONS **33** CITATIONS
SEE PROFILE

Omar Arif
National University of Sciences and Technology
32 PUBLICATIONS **268** CITATIONS
SEE PROFILE

Image Caption Generator with Novel Object Injection

Mirza Muhammad Ali Baig, Mian Ihtisham Shah, Muhammad Abdullah Wajahat, Nauman Zafar and Omar Arif
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology, Islamabad, Pakistan
Email: {14bscsmbaig, 14bscsmshah, 14bscsmwajahat, 14mscsnzafar, omar.arif}@seecs.edu.pk

Abstract—**Image captioning is a field within artificial intelligence that is progressing rapidly and it has a lot of potentials. A major problem when working in this field is the limited amount of data that is available to us as is. The only dataset considered suitable enough for the task is the Microsoft: Common Objects in Context (MSCOCO) dataset, which contains about 120,000 training images. This covers about 80 object classes, which is an insufficient amount if we want to create robust solutions that aren't limited to the constraints of the data at hand. In order to overcome this problem, we propose a solution that incorporates Zero-Shot Learning concepts in order to identify unknown objects and classes by using semantic word embeddings and existing state-of-the-art object identification algorithms. Our proposed model, Image Captioning using Novel Word Injection, uses a pre-trained caption generator and works on the output of the generator to inject objects that are not present in the dataset into the caption. We evaluate the model on standardized metrics, namely, BLEU, CIDEr and ROUGE-L. The results, qualitatively and quantitatively, outperform the underlying model.**

Index Terms—**Image Caption, Microsoft Common Objects in Context (MSCOCO), Convolutional Neural Network, Recurrent Neural Network**

I. Introduction

Generating natural language descriptions of images automatically is called image captioning and is a very challenging task. Considerable research has been done over the past few years for solving the image captioning problem. Most contemporary solutions involve the use of an encoder-decoder mechanism in the form of a Convolutional Neural Network, for acquiring the feature map of the image, followed by a Recurrent Neural Network for machine translation tasks and generating natural language sentences. However, what most solutions fail to include is a way to identify objects or classes that are not present in the training data set. Currently, the only data set available that is tailored to the Image Captioning task is the MSCOCO (Microsoft Common Objects in Context) [1] data set which contains images along with their natural language descriptions. However, the dataset only covers a limited number of object categories (specifically 80 classes), which when compared to the 1000 object classes in ILSVRC Challenge 2012, is comparatively lacking. The most obvious way to tackle this problem is to increase the number of objects covered in the training data set. However it is very time consuming and laborious to add more training examples with ground truth captions. State-of-the-art deep learning models require significant number of training examples to meaningfully affect the algorithms accuracy.

Another way of solving this problem is to use Zero-Shot Learning to inject novel objects into the caption generating mechanism. Zero-Shot Learning is a supervised learning technique that allows us to predict objects or classes that are not part of our training set and are hence unseen or unknown to the algorithm. The way we use this concept in our methods is by exposing the algorithm to more object classes that are currently present within the training dataset that we are using. Our solution aims to leverages state of the art solutions for existing subproblems and bridge this gap in order to improve the accuracy of the generated captions.

The solution we present uses Long-Short Term Memory that uses a word embedding trained on external corpora as well as training data captions. An object identifier, which in our case will be YOLO9000 model (jointly trained on MSCOCO [1] and Imagenet [2]), will allow us to accurately detect the objects in the image and, once the caption has been generated, we can compare and selectively inject novel objects that were not identified in the caption. Our main contribution is proposing a modular architecture that uses existing state-of-the-art techniques that can detect and insert new objects that are not present in the dataset that the image caption generator was trained on. The experiments that we have conducted give us positive results and show that this approach successfully inserts the correct objects in the caption, and hence leads to a more accurate result than that generated by the underlying image caption generator.

II. Related Work

We briefly review existing algorithms and techniques in the context of image captioning, zero-shot learning and novel object injection.

A. Image Captioning

Recent works in image captioning focus on an encoder-decoder framework [3]–[5]. This framework allows us to generate an encoding for the image using a Convolutional Neural Network, and generate a caption with that encoding using a Recurrent Neural Network. Both CNNs and RNNs excel at their respective tasks, those being object identification and detection, and NLP tasks like machine translation respectively, and have thus established themselves as the norm when it comes to the task of image captioning. Karpathy and Fei-Fei [4] expand on the basic encoder-decoder model by using a bidirectional RNN. This differs from a regular unidirectional RNN by using two hidden states, one running in the positive time direction (forward) and one in the negative time direction (backward). This allows the network to preserve information from both the past and the future, and therefore they can understand context better. Donahue et al [5] adds the to basic CNN-RNN approach in their proposed Long-term Recurrent Convolutional Network (LRCN) by feeding in image features along with words at each timestep.

Other models include additions on top of this core framework in order to get better results. Xu et al [6] feeds in a feature projection of the image along with word embeddings at every timestep to allow the algorithm to focus on and give attention to specific parts of the image over time. This allows for the algorithm to give attention to different parts of the image over time. A similar approach is taken in [7], but in this case, the effectiveness of introducing the features at different places in the algorithm (as a joint input, as every timestep etc) is discussed. In [8], high-level concepts and attributes are injecting in the RNN model while performing the image captioning task. Such features are also utilized in [9] as semantic attention. A more recent development using the same concept is found in [10], where attention to the image is not given at every timestep, rather where and when to give attention is decided by the model itself. A multimodal framework in which both language and image features are embedded into the same space is adopted in [11], [12].

B. Zero-Shot Learning

Zero-shot learning allows us to predict unseen categories or classes using classes in our existing dataset. Because of this, it has gained significant traction over the past few years as it allows us to effectively expand our dataset without having to go through the process of collecting and independently labeling each class and hence, it saves us time and effort in favor of something more elegant. In [13], we find that unseen classes are predicted by finding a connection between them and the seen classes that are present. This connection can be represent by mapping these classes to the same semantic space. [14], [15] do the same task by establishing a mapping function to link visual features to semantic representations.

In the context of image captioning specifically, [16] used transfer learning to transfer the knowledge of objects in the same semantic space to generate natural language captions about unseen objects and classes in their proposed Deep Compositional Captioner (DCC). Our focus will be to utilize the semantic relationship found in external text corpora as well as ground truth captions, and using this relationship in tandem with state-of-the-art object identification algorithms to inject novel objects into the generated captions in the right context.

III. Methodology

This solution is aimed towards adding novel words and unseen object classes into the caption at the decoding stage. In order to do this, we first train a Word2Vec [17] word embedding on both training captions and external corpora (namely Wikipedia) and use those embeddings on the outputs of an image caption generator.

Figure 1 outlines the architecture of the proposed solution. It comprises of 3 modules. The first is the image caption generator, which is responsible for generating of a caption of the image being fed into it. For our solution, we are using a model of the caption generator described in [3], as our main goal is to show that novel objects can be correctly embedded into the generated captions. We are using a version of the model that has been pre-trained on MSCOCO. The second module is an object detection algorithm that is responsible for accurately generating a list of objects present in the image. The third module is a semantic word embedding. The embedding space is used to determine how close two objects are, contextually and semantically, within that space. The objects present in the caption, and those extracted from the image are then compared and are probabilistically replaced if they are close enough to each other, with preference being given to the objects extracted from the image as their results would be more accurate than those identified in the caption.

A. Semantic Word Embedding

The Word2Vec [17] embedding maps language features into the same semantic space. These features lie both in our training set (seen classes) and external unseen corpora (unseen classes). The importance of this is that it allows us to compare two classes effectively as classes that are semantically similar, such as man and king, are mapped closer together in the same vector space as compared to a pair of words like man and dog. Apart from the input, the word embedding space is also used when running comparisons between generated words and objects identified within the image as this allows us to make sensible word substitutions. The word embedding is trained using the Continuous-Bag-Of-Words (CBOW) Word2Vec model with negative sampling. CBOW is preferred to the Continuous Skip-Gram model because of the nature of the task at hand, which is to predict the next word if a context is given, which is what CBOW is programmed to achieve. The raw text used to train the embedding comprises of the captions present in the training dataset alongside the latest Wikipedia article dump.

B. Object Detection

In order to inject novel words into the generated caption, they first need to be accurately identified in the image. For this purpose, a pre-trained detection model, Yolo9000 [18], is used. Yolo9000 is a state-of-the-art, real-time detection system able to detect over 9000 object categories. It can outperform other detection systems like Faster RCNN. It is jointly trained on object detection and classification, COCO detection dataset and the ImageNet classification dataset respectively. The algorithm will return a list of objects that are present within the image, which we can then use after the caption generation process.

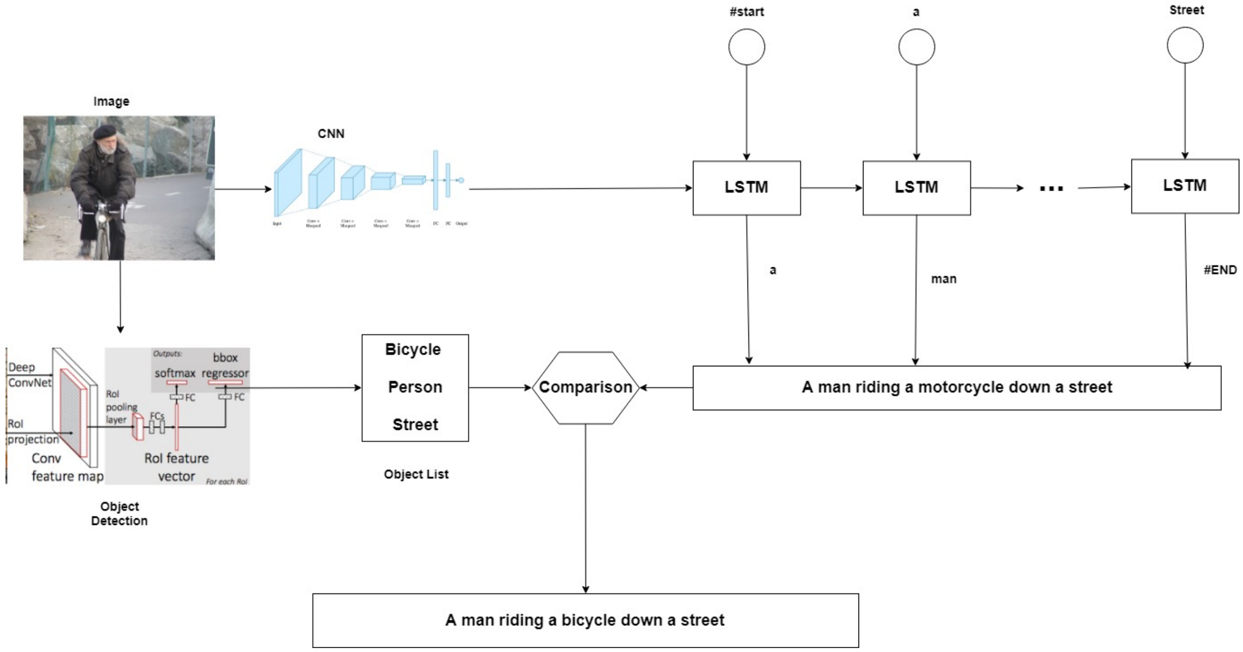

Fig. 1. Overall architecture of the proposed solution. The design is meant to be modular so that the solution can function on top of any image caption generator

C. Novel Object Injection

The two modules described above are used in conjunction in order to fulfill the desired task, i.e allowing our caption to include unseen object classes. This is done by combining the accuracy and effectiveness of existing solutions to our sub-problems and leveraging them in order to reach our goal.

The object detection algorithm will give us a list of objects that are present in the input image. Along with their names, we will also receive the likelihood of them actually being in the image. The caption can be generated using any modern framework, as our proposal is meant to function on top of the caption generator by working with the output. In our examples, we used the pioneering CNN-LSTM framework outlined in [3] to test the effectiveness of our algorithm. Once a caption is generated, the nouns present in it are extracted and compared to the objects identified in the image using the following equations:

$$s(a,b) = \frac{\boldsymbol{a}.\boldsymbol{b}}{||\boldsymbol{a}||||\boldsymbol{b}||} \tag{1}$$

$$f(a,b) = \left\{ \begin{array}{ll} s \times p(b), & s \times p(b) \geq \lambda \\ 0, & s \times p(b) < \lambda \end{array} \right\} \tag{2}$$

where a represents the object that is present in the caption generated and b represents an object that is present in the image and has been identified by the object detection model. s(a, b) is the cosine similarity between the two words and is calculated using their respective vectors that are present in the semantic word embedding space. p(b) represents the probability that the object b is in the image. The object is substituted if the join factor calculated using s(a, b) and p(b) is greater than or equal to the threshold value λ.

TABLE I
TESTING ON FLICKR8K. BLEU-1,2,3,4/CIDER/ROUGE-L METRICS COMPARED TO THE ORIGINAL MODEL

Model	B-1	B-2	B-3	B-4	CIDEr	ROUGE-L
Google NIC	57.6	36.4	17.7	9.0	19.0	43.1
Novel Object Injection	**59.2**	**38.2**	**18.7**	**10.5**	**21.1**	**44.0**

TABLE II
TESTING ON FLICKR30K. BLEU-1,2,3,4/CIDER/ROUGE-L METRICS COMPARED TO THE ORIGINAL MODEL

Model	B-1	B-2	B-3	B-4	CIDEr	ROUGE-L
Google NIC	62.6	41.6	24.0	14.2	23.9	42.0
Novel Object Injection	**63.0**	**42.1**	**24.2**	**14.6**	**25.0**	**42.3**

IV. EXPERIMENTS

We performed a set of experiments to assess the effectiveness of our model using several metrics and data sets in order to compare to the original model.

A. General Image Caption Correction

1) Datasets: The evaluation is performed on two datasets: Flickr8k [19] and Flickr30k [20]. The Flick8k dataset consists of 8000 images and the Flickr30k dataset consists of 30000 images. The reasoning for choosing these specific sets is because the model we are using is pre-trained on the MSCOCO [1] dataset, which means that there would be objects in the chosen datasets that the caption generator would not be able to predict, and so we will leverage that in empirically evaluating our algorithm.

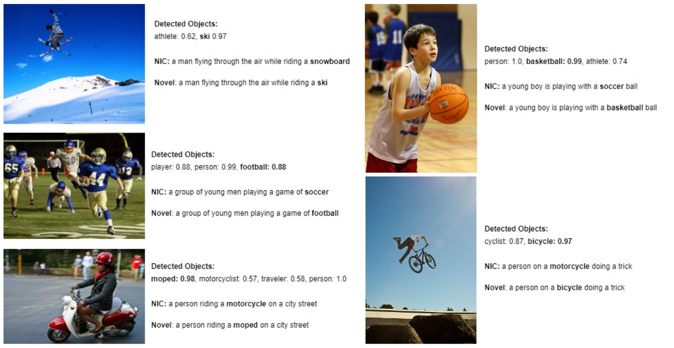
Detected Objects:
athlete: 0.62, **ski** 0.97
NIC: a man flying through the air while riding a **snowboard**
Novel: a man flying through the air while riding a **ski**
Detected Objects:
person: 1.0, **basketball: 0.99**, athlete: 0.74
NIC: a young boy is playing with a **soccer** ball
Novel: a young boy is playing with a **basketball** ball
Detected Objects:
player: 0.88, person: 0.99, **football: 0.88**
NIC: a group of young men playing a game of **soccer**
Novel: a group of young men playing a game of **football**
Detected Objects:
cyclist: 0.87, **bicycle: 0.97**
NIC: a person on a **motorcycle** doing a trick
Novel: a person on a **bicycle** doing a trick
Detected Objects:
moped: 0.98, motorcyclist: 0.57, traveler: 0.58, person: 1.0
NIC: a person riding a **motorcycle** on a city street
Novel: a person riding a **moped** on a city street

Fig. 2. Sample positive results from the Flickr30k dataset. The detected objects are identified by the YOLO9000 model, and the sentences are generated by 1) Google NIC and 2) Novel Object Injection

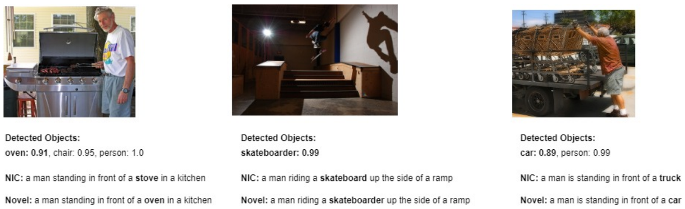
Detected Objects:
oven: 0.91, chair: 0.95, person: 1.0
NIC: a man standing in front of a **stove** in a kitchen
Novel: a man standing in front of a **oven** in a kitchen
Detected Objects:
skateboarder: 0.99
NIC: a man riding a **skateboard** up the side of a ramp
Novel: a man riding a **skateboarder** up the side of a ramp
Detected Objects:
car: 0.89, person: 0.99
NIC: a man is standing in front of a **truck**
Novel: a man is standing in front of a **car**

Fig. 3. Sample negative results from the Flickr30k dataset. These examples outline the constraints of the model, that being the constraints of the underlying algorithms. The YOLO9000 algorithm detects the wrong objects in the pictures which result in incorrect captions

The Word2Vec embedding is trained using both external test corpora (namely Wikipedia) as well as captions present in the training dataset. The Wikipedia article dump [1] is openly available and contains 4.4 million articles. These articles are preprocessed by removing any punctuation and spaces. Each article is then saved as one line, and is tokenized, keeping all words that appear at least 5 times. The Word2Vec model is trained using the Continuous-Bag-Of-Words (CBOW) strategy with negative sampling. The dimensionality of the word vectors is empirically set to 512. The threshold λ is set to 0.6 empirically after running evaluations several times.

2) Results: For quantitative evaluation of our algorithm, we adopt the metrics used in the MSCOCO caption challenge, namely BLEU [21], CIDer [22], and ROUGE-L [23]. Each method evaluates a candidate sentence by measuring how well it matches a set of five reference sentences written by humans. The BLEU score is is a form of precision of word n-grams between generated and reference sentences. Because our model is effectively changing the underlying caption that is being generated, the best way to test it's effectiveness is to run these metrics on the subset of captions that are changed by Novel Object Injection. In doing so, we effectively create test splits for the evaluation sets.

We conducted experiments of our model on the Flickr8k and Flickr30k. These datasets contain objects that are outside the training dataset, which made them ideal for verification. In order to ensure fairness, both the original results and the results using Novel Object Injection are calculated using the MSCOCO evaluation toolkit [2].

3) Evaluation on Flickr8k and Flickr30k: Table I shows the performance of our model as compared to Google NIC [3] across all six metrics on the Flickr8k dataset, while Table II shows the same information for the Flickr30k dataset. Overall, the results show that Novel Object Injection consistently performs better than the underlying image caption generator in both cases. This shows that the idea of inserting objects seperately into captions after they have been generated does improve the accuracy of the model, which means that the word injection being performed is accurate. This is more important given the fact that the model we are using in our algorithm is not trained on the two sets we are using for evaluation. This means that it is possible for the algorithm to scale and cover lots of novel objects.

4) Qualitative Analysis: In order to analyze the performance of our model, we take instances from the evaluation sets as well as external images that are not present in either the training or evaluation sets. Figure 2 provides examples where we can see how novel words are injected into captions. We see that the objects that are highlighted are successfully replaced with the wrongly identified object in the generated caption, without changing the context or meaning of the sentence.

We also see, however, that the model is prone to making mistakes as well, as showcased in Figure 3, and this is mostly down to the ability of the object detection module to correctly identify objects in the image. In these cases, the object detection algorithm identified objects in the image inaccurately, which result in the wrong objects being inserted into the algorithm. This happens because we are assuming that the results from the object detection algorithm will be more accurate than those in the caption.

B. Food Image Caption Correction

The algorithm is also evaluated on images containing various food items.

1) Dataset: Our goal was to make a data set that contains common food items, augmented with subcontinental dishes. We started by experimenting on the publicly available data set of food images, Food-101 [24]. It contains 101 classes of food items with 1000 images for each class. Food-101 is designed specifically for multi-class classification. This data set does not include food items or classes from the subcontinental cuisine which makes a huge portion of the food that people intake in the subcontinental region. Some subcontinental dishes exhibit low inter-class variation and are very similar to each other, so collecting high quality data for proper classification of different categories was a big challenge. The results returned by Google search engine against textual search queries for food images were quite relevant with very low noise content. Based on such results from Google search engine, our new

[1] https://dumps.wikimedia.org/enwiki/latest/enwiki-latest-pages-articles.xml.bz2

[2] https://github.com/tylin/coco-caption

data set was created by querying Google against each label of our data set. The newly formed data set had classes from Food-101, that are common and eaten everywhere. So, the final data set contained 100 common and subcontinental food classes, split into training and validation images. Each class contained around 800 training images and 200 validation images.

2) Food Recognition: Presented with the food image, the first task is to recognize the food. Owing to great success of CNNs, we shortlist top performing pre-trained models to train on our data set using transfer learning. As our data set contains 100 classes, so we surveyed accuracies on similar 100 class data sets (CIFAR-100 and Food-101). [25] achieved an accuracy of 75.72% on CIFAR-100 by using ELU as the activation layer in deep neural networks while [26] achieved an accuracy of 75.7% on the same by developing a CNN for processing spatially-sparse inputs. [27] used Inception-v3 to achieve a Top-1 accuracy of 88.28% on Food-101. [28] achieved 90.27% accuracy on Food-101 by using Wide-Slice Residual networks. Based on the analysis, we selected several pre-trained CNN models such as VGG-16, VGG-19 [29] Inception-v3 [30], Inception-v4 [31] and ResNet [32]. These models are pre-trained on Imagenet data set. Transfer learning was used to train these models on our data set. The last fully connected layer was removed and appended with dropout, ReLU activations and softmax layers. Fine tuning the model on data set took about 15 hours on a single Titan X GPU with a 12GB of memory. Models based on Inception-v3 and Inception-v4 gave better performances and were used as recognition engine in the rest of the paper.

a) Fine Tuning: After initial filtering we selecting Inception-v3 and Inception-v4 as the appropriate model, we proceeded on improving the validation accuracy of the model on our dataset. For this, we employed various techniques. First, we did intensive data augmentation so that the model is robust to affine variations as much as possible and the images are efficiently trained. In every epoch, various transformations, with random parameters specified by parametric range, were applied on each image of the data set to produce copies that are transformed from the original image. These included translations, rotations, shearing, zooming and flipping. The optimized results were obtained by setting parameters for data augmentation as mentioned in Table III. Images were also rescaled, considering the large values of RGB coefficients for the model to process. Rescaling involved multiplication of the RGB value by 1/255 factor, so target values are normalized and lie between 0 and 1. This makes the image robust to variations in illumination and makes processing data faster.

The Word2Vec embedding is trained using both external test corpora (namely Wikipedia), captions present in the training dataset and data crawled from the food and nutritional sites.

TABLE III
PARAMETERS USED AND THEIR VALUES FOR DATA AUGMENTATION

Parameter	Value
Width Shift	0.2
Height Shift	0.2
Rotation	90
Rescale	1/255
Shear	0.2
Zoom	0.2
Horizontal Flip	True
Vertical Flip	True

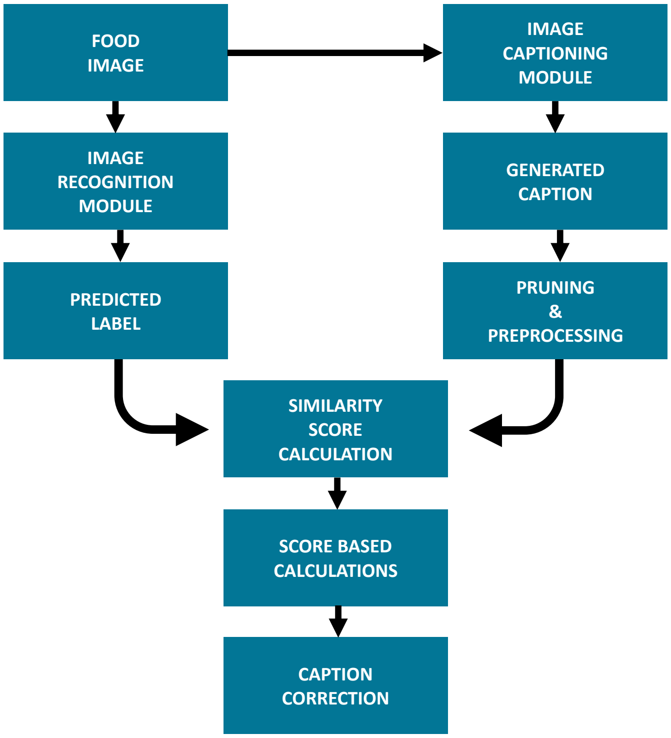

Fig. 4. Food Caption Correction Workflow

3) Workflow: As we can see in Figure 4, firstly the desired image is fed to recognition model to get its label from the food recognition module. Also, the caption is generated using pioneering CNN-LSTM framework outlined in [3]. Generated caption is then pruned and prepared to use for similarity calculation. Stemming is applied. Textual model is loaded for similarity score calculation. Similarity score is calculated for words (i.e. pruned from caption) given label. After finding score, a list containing key value pairs is prepared. Keys consist of individual words obtained from pruning caption along with similarity score as value. For finding appropriate word to replace, we assessed the calculated score. Higher the similarity, higher will be the score. All the calculated scores are arranged in descending order. Top most score will be the most suited candidate word for replacement. The new corrected caption will contain the label as replaced word.An example depicting input and output of all the phases is shown in Figure 5

4) Results: Total number of images reserved for testing purpose are 14,793. As explained in training section, different models are trained using different training parameters. Each model contains different set of vocabulary. All the models are tested and evaluated on fine-tuned inceptionV4 model

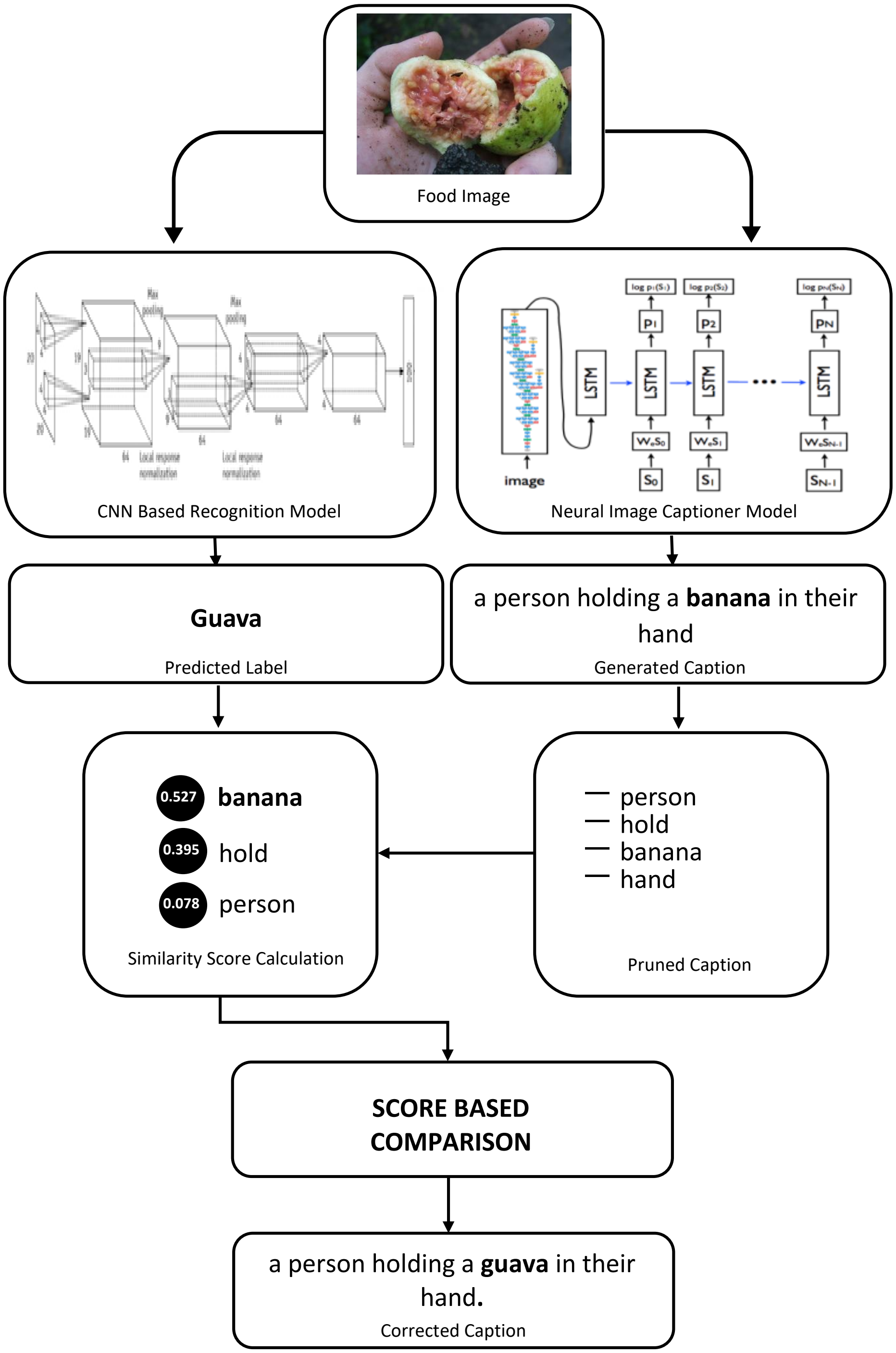

Fig. 5. Example depicting different phases of the Food Caption Correction

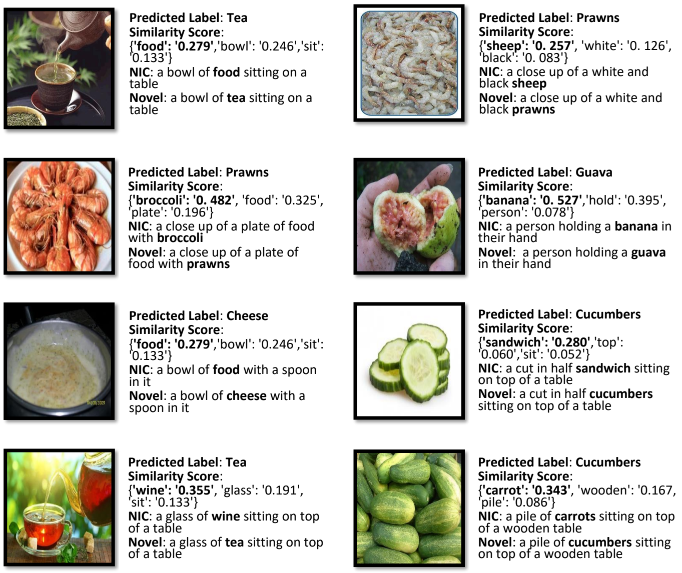
Predicted Label: Tea
Similarity Score:
{'food': '0.279','bowl': '0.246','sit': '0.133'}
NIC: a bowl of food sitting on a table
Novel: a bowl of tea sitting on a table
Predicted Label: Prawns
Similarity Score:
{'sheep': '0. 257', 'white': '0. 126', 'black': '0. 083'}
NIC: a close up of a white and black sheep
Novel: a close up of a white and black prawns
Predicted Label: Prawns
Similarity Score:
{'broccoli': '0. 482', 'food': '0.325', 'plate': '0.196'}
NIC: a close up of a plate of food with broccoli
Novel: a close up of a plate of food with prawns
Predicted Label: Guava
Similarity Score:
{'banana': '0. 527','hold': '0.395', 'person': '0.078'}
NIC: a person holding a banana in their hand
Novel: a person holding a guava in their hand
Predicted Label: Cheese
Similarity Score:
{'food': '0.279','bowl': '0.246','sit': '0.133'}
NIC: a bowl of food with a spoon in it
Novel: a bowl of cheese with a spoon in it
Predicted Label: Cucumbers
Similarity Score:
{'sandwich': '0.280','top': '0.060','sit': '0.052'}
NIC: a cut in half sandwich sitting on top of a table
Novel: a cut in half cucumbers sitting on top of a table
Predicted Label: Tea
Similarity Score:
{'wine': '0.355', 'glass': '0.191', 'sit': '0.133'}
NIC: a glass of wine sitting on top of a table
Novel: a glass of tea sitting on top of a table
Predicted Label: Cucumbers
Similarity Score:
{'carrot': '0.343', 'wooden': '0.167, 'pile': '0.086'}
NIC: a pile of carrots sitting on top of a wooden table
Novel: a pile of cucumbers sitting on top of a wooden table

Fig. 6. Sample positive results from our own prepared FOOD dataset. Labels are predicted by the fine-tuned inceptionV4 model, and the sentences are generated by 1) Google NIC and 2) Novel Object Injection

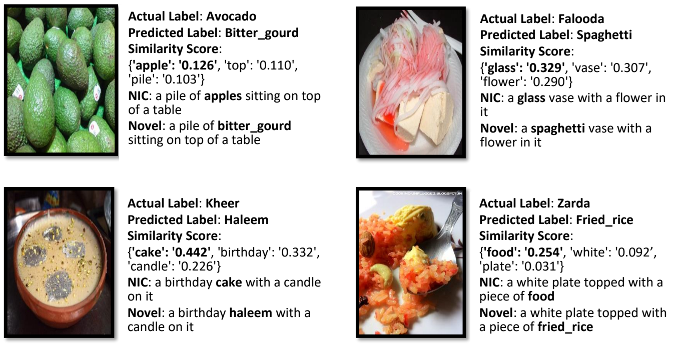
Actual Label: Avocado
Predicted Label: Bitter_gourd
Similarity Score:
{'apple': '0.126', 'top': '0.110', 'pile': '0.103'}
NIC: a pile of apples sitting on top of a table
Novel: a pile of bitter_gourd sitting on top of a table
Actual Label: Falooda
Predicted Label: Spaghetti
Similarity Score:
{'glass': '0.329', 'vase': '0.307', 'flower': '0.290'}
NIC: a glass vase with a flower in it
Novel: a spaghetti vase with a flower in it
Actual Label: Kheer
Predicted Label: Haleem
Similarity Score:
{'cake': '0.442', 'birthday': '0.332', 'candle': '0.226'}
NIC: a birthday cake with a candle on it
Novel: a birthday haleem with a candle on it
Actual Label: Zarda
Predicted Label: Fried_rice
Similarity Score:
{'food': '0.254', 'white': '0.092', 'plate': '0.031'}
NIC: a white plate topped with a piece of food
Novel: a white plate topped with a piece of fried_rice
COOKING-UNPLUGGED.BLOGSPOT.IN

Fig. 7. Sample negative results from our own prepared FOOD dataset.False prediction resulted in incorrect captions

as recognition module. Our proposed Novel Object Injection Method superseded Google NIC with corrected caption percentage of 77.41 percent. We have defined the threshold for replacement of words in caption to make it correct. if any of the word having similarity value equal to 1.0 then replacement will not occur. As a result, caption will remain unchanged. In our case, no change occurred in 1563 image captions. However, successfully corrected captions are 11,452 in number (i.e. 77.41%). Wrong predicted images resulted in wrong caption. Rest of the images (i.e. 1778) got false caption due to wrong label prediction. For the purpose of our models respectable evaluation, images are taken from both sources (i.e. from evaluation dataset and external sources). It is assured that images dont exist in training or testing sets. We can see the models performance in Figure 6 as words are replaced successfully to make the correct caption. Semantic meaning of the object and structure is not disturbed. Also, context is preserved for correctness of the sentence. However, downside of the recognition module also introduced the mistaken or wrong captions and make the model error prone. As we can see in Figure 7, wrong predicted label generated wrong caption because wrong words are being replaced in the original caption. This is also due to the assumption that our recognition module is more accurate than the one used in generating captions.

V. Conclusion

We have presented an Image Caption Generator with Novel Object Injection, which leverages external visual recognition and semantic word embeddings for image captioning. To verify this, we devised a model that can accommodate current image captioning methods and improve their accuracy by detecting novel objects and probabilistically injecting them into the result. Experiments conducted on Flickr datasets validate our proposal as performance is shown to improve when compared to the underlying image captioning model.

References

[1] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[3] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," in *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*. IEEE, 2015, pp. 3156–3164.

[4] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3128–3137.

[5] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[6] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015, pp. 2048–2057.

[7] T. Yao, Y. Pan, Y. Li, Z. Qiu, and T. Mei, "Boosting image captioning with attributes," *OpenReview*, vol. 2, no. 5, p. 8, 2016.

[8] Q. Wu, C. Shen, L. Liu, A. Dick, and A. van den Hengel, "What value do explicit high level concepts have in vision to language problems?" in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 203–212.

[9] Q. You, H. Jin, Z. Wang, C. Fang, and J. Luo, "Image captioning with semantic attention," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4651–4659.

[10] J. Lu, C. Xiong, D. Parikh, and R. Socher, "Knowing when to look: Adaptive attention via a visual sentinel for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 6, 2017.

[11] J. Mao, X. Wei, Y. Yang, J. Wang, Z. Huang, and A. L. Yuille, "Learning like a child: Fast novel visual concept learning from sentence descriptions of images," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2533–2541.

[12] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*. IEEE, 2016, pp. 2285–2294.

[13] Z. Zhang and V. Saligrama, "Zero-shot learning via semantic similarity embedding," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4166–4174.

[14] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean, "Zero-shot learning by convex combination of semantic embeddings," *arXiv preprint arXiv:1312.5650*, 2013.

[15] M. Bucher, S. Herbin, and F. Jurie, "Improving semantic embedding consistency by metric learning for zero-shot classiffication," in *European Conference on Computer Vision*. Springer, 2016, pp. 730–746.

[16] L. Anne Hendricks, S. Venugopalan, M. Rohrbach, R. Mooney, K. Saenko, T. Darrell, J. Mao, J. Huang, A. Toshev, O. Camburu *et al.*, "Deep compositional captioning: Describing novel object categories without paired training data," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[17] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[18] J. Redmon and A. Farhadi, "Yolo9000: better, faster, stronger," *arXiv preprint*, 2017.

[19] C. Rashtchian, P. Young, M. Hodosh, and J. Hockenmaier, "Collecting image annotations using amazon's mechanical turk," in *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*. Association for Computational Linguistics, 2010, pp. 139–147.

[20] P. Young, A. Lai, M. Hodosh, and J. Hockenmaier, "From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions," *Transactions of the Association for Computational Linguistics*, vol. 2, pp. 67–78, 2014.

[21] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, 2002, pp. 311–318.

[22] R. Vedantam, C. Lawrence Zitnick, and D. Parikh, "Cider: Consensus-based image description evaluation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4566–4575.

[23] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," *Text Summarization Branches Out*, 2004.

[24] L. Bossard, M. Guillaumin, and L. Van Gool, "Food-101–mining discriminative components with random forests," in *European Conference on Computer Vision*. Springer, 2014, pp. 446–461.

[25] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[26] B. Graham, "Spatially-sparse convolutional neural networks," *arXiv preprint arXiv:1409.6070*, 2014.

[27] H. Hassannejad, G. Matrella, P. Ciampolini, I. De Munari, M. Mordonini, and S. Cagnoni, "Food image recognition using very deep convolutional networks," in *Proceedings of the 2nd International Workshop on Multimedia Assisted Dietary Management*. ACM, 2016, pp. 41–49.

[28] N. Martinel, G. L. Foresti, and C. Micheloni, "Wide-slice residual networks for food recognition," *arXiv preprint arXiv:1612.06543*, 2016.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[30] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[31] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning." in *AAAI*, 2017, pp. 4278–4284.

[32] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.