Training and Evaluating Language Model for Image Captioning: The Quirks and What Works

Shahla Farzana
University of Illinois chicago
Chicago, Illinois
sfarz3@uic.edu

ABSTRACT

Recurrent neural network language model has achieved state-of-the-art results in image captioning system. The image information can be fed to the neural network either by directly incorporating it in the RNN - conditioning the language model by 'injecting' image features - or in a layer following the RNN - conditioning the language model by 'merging' image features. In this paper we first implemented a simple merge model architecture of image captioning system and experimented with the word embedding layer of this model to see how it influences the language model of the system. We have chosen the merge architecture as it allows the RNN's hidden state vector to shrink in size by up to four times. Our evaluation shows promising results as manipulating the embedding layer of the language model makes difference in generating captions. Within this neural models, there are different architecture in terms of integrating the image and textual features which subsequently influence the model size, scalability and caption generation performance.

1 INTRODUCTION

Image captioning is an interesting problem, involving both computer vision techniques and natural language processing techniques. Recent progress in automatic image captioning has shown that an image-conditioned language model can be very effective at generating captions.The generated captions are conceptual: it can be either specific captioning like identifying people and locations or generic like related to scene understanding (which is our main goal). One way to do this is to use a neural language model, typically in the form of a recurrent neural network (RNN), accepting textual and image features and map those features in common space to predict the next token in sequence.

Recent works on image captioning have explored different architecture of the fusion model of image and text feature like init-inject, pre-inject, par-inject and merge model [2, 3, 11, 15]. In this paper, we have picked the late fusion specially the merge model to implement the image captioning system as this kind of delayed integration leads to create less memory intensive model allowing faster training. Within this architecture, natural language processing model like RNN has involved learning word vector representations [1] through its embeding layer. In this paper, we have experimented with different methods of leaning the word vector representation using various word embedding approach. We then compare the merits of these different usage off embedding layers in language modeling part for the first time by using the same state-of-the- art CNN as input to generate the image features. We examine issues in the different approaches, including linguistic irregularities, caption repetition, incomplete captions and detailed captioning. By using pretrained word embedding layer and fine tuning the layer to the task specific system, we achieve slightly better performance over previously published results on the benchmark Flickr8k dataset for merge model architecture [12] . However, the gains we see in BLEU do not translate to human judgments. In sum, the contributions of this paper are as follows:

- We compare the use of different word vector representations as the embedding layer of language models trained to generate image captions.
- We demonstrate that a state-of-the-art word vector representation [10] and its fine tuned variation works better than the randomly initialized word vectors.
- We advance the state-of-the-art BLEU scores on the Flickr8k dataset.
- we introduced a qualitative error analysis on the generated captions to look beyond the automated evaluation metric BLEU

2 RELATED WORK

Image captioning problem has been investigated for a long time in computer vision literature and it has come a long way since its inception. While some previous models [5, 7] have been proposed to address the problem of image captioning, they rely on either use sentence templates, or treat it as retrieval task through ranking the best matching sentence in database as caption. With the recent surge in neural network based model, it has taken over the template based models in image caption problem. Recent work [13, 14] has indicated that embedding visual and language to common semantic space with relatively shallow recurrent neural network (RNN) can yield promising results.

In this section we discuss a number of recent image caption generation models with emphasis on how the image conditions the neural language model, based on the distinction between inject and merge architectures illustrated in Figure 1. We will first outline four broad sub-categories of architectures of the language generation model below:

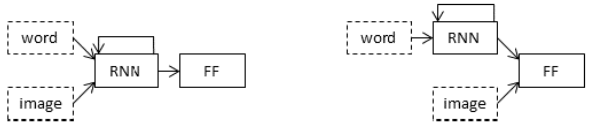

Figure 1: The inject and merge architectures for caption generation. Legend: RNN - recurrent neural network; FF - feed forward layer.

2.1 Types of System Architecture

There are different subcategories of system architecture based on the integration of linguistic and image features in a multimodal layer, and those that inject image features directly into the caption encoding process. We can in fact distinguish four theoretical possibilities arising from these, which are described below:

- Init-inject: This type of model treats the image vector as the initial hidden state vector of an RNN [2]. It requires the image vector to have the same size as the RNN hidden state vector. This is an early binding architecture and allows the image representation to be modified by the RNN.
- Pre-inject: Pre-inject models treat the image as though it were the first word in the prefix [11]. The word vectors of the caption prefix come later. The image vector is thus treated as a first word in the prefix. It requires the image vector to have the same size as the word vectors. This too is an early binding architecture
- Par-inject: Par-injection inputs the image features into the RNN jointly with each word in the caption. It is by far the most common architecture used and has the largest variety of implementation. Par-injection inputs the image features into the RNN jointly with each word in the caption. It is by far the most common architecture used. The image vector (or a vector derived from the image vector) serves as input to the RNN in parallel with the word vectors of the caption pre
  x, such that either (a) the RNN takes two separate inputs; or (b) the word vectors are combined with the image vector into a single input before being passed to the RNN.
- Merge: Rather than combining image features together with linguistic features from within the RNN, merge architectures delay their combination until after the caption prefix has been vectorised [9]. This is a late binding architecture and it does not modify the image representation with every time step.

We have picked the merge model for our implementation with variations in the embedding layer. The merge architecture does have practical advantages, as conditioning by merging allows the RNN's hidden state vector to shrink in size by up to four times. With these architectures in mind, We will talk about augmenting the word embedding layer o the language model in the next subsection.

2.2 Word Vector Representation

Deep learning language models have achieved remarkable results in computer vision [4]. Within natural language processing, much of the work with deep learning methods has involved learning word vector representations through neural language models. Word vectors, wherein words are projected from a sparse, 1-of-V encoding (here V is the vocabulary size) onto a lower dimensional vector space via a hidden layer, are essentially feature extractors that encode semantic features of words in their dimensions. In such dense representations, semantically close words are likewise closeâĂŤin euclidean or cosine distanceâĂŤin the lower dimensional vector space. The pretrained embedding layer has been proved to improve the sentence (question and sentiment) classification task [6].
We investigate the word embedding layer in the language model of image caption generation task and compare those results with our baseline model. To the best of our knowledge, we have taken this approach for the first time to show the impact of varied word vector representation for image caption problem.

3 OUR MODEL

Overview. The ultimate goal of our model is to generate generic caption (related to scene understanding) of image. During training, the input to our model is a set of images and their corresponding sentence caption Figure 2. We first present a model that first extract the image feature with deep neural model and process the input caption sequence with the RNN model. Then the model integrates the image and text features and learn the next token in sequence. After the training phase, multi modal Recurrent Neural Network model learns to generate the snippets. The merge model architecture of the system is depicted in Figure 3

Figure 2: Overview of our approach. A dataset of images and their sentence captions is the input to our model (left). The model then learns to generate novel caption (right)

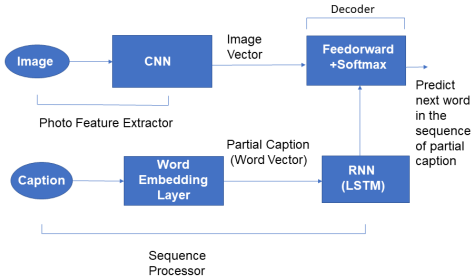

Figure 3: The merge model architecture of image caption system

3.1 The Flick8k Dataset

Flilckr8K contains 8092 images that are each paired with five different captions which provide clear descriptions of the salient entities and events. The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations. **Preliminary analysis** There are total The number of 8092 unique file names. And there are five captions per image. Figure 4 shows some saple image caption pairs from the dataset

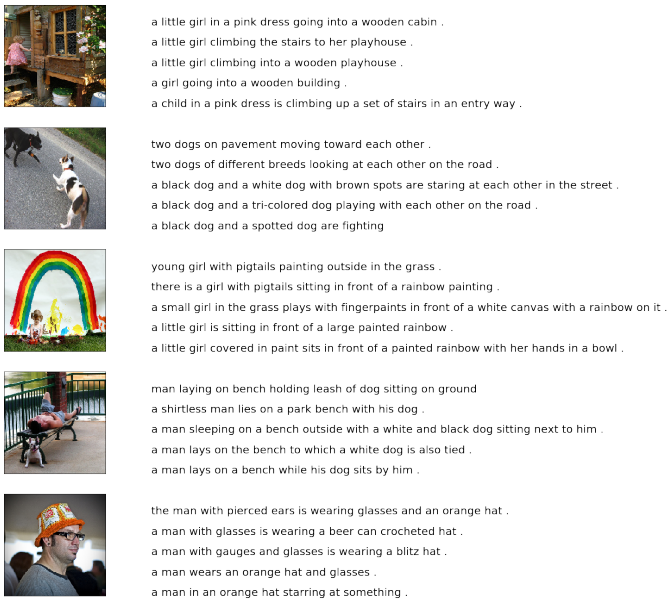

Figure 4: Sample images with 5 captions in Flickr8k dataset

3.2 Text preparation

We create a new dataframe dfword to visualize distribution of the words. It contains each word and its frequency in the entire tokens in decreasing order. The vocabulary size is 8918. The most common words are articles such as "a", or "the", or punctuations. These words do not have much infomation about the data. Figure 5 shows the preliminary distribution of words and caption length In order to clean the caption, I will create three functions that: remove punctuation, remove single character and remove numeric characters. After cleaning, the vocabulary size get reduced by about 200. Then we add start and end sequence tokens.

3.3 Image preparation

We create features for each image using VGG16's pre-trained networks [8]. This network takes input of size (224,224,3). The output layer contains 1,000 nodes. VGG16 is developed to classify images into 1,000 different classes. As we did not use VGG16 for the sake of the classification but I just need it for extracting features, I will remove the last layer from the network. For each image, 4096 features are created. As I cannot visualize the 4096 dimensional space,

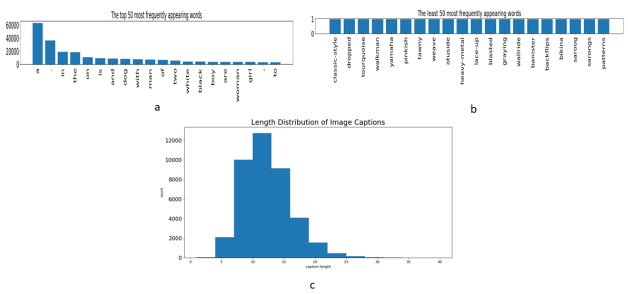

Figure 5: Preliminary analysis of text data of Flickr8k (a) 50 most frequent words, (b) 50 least frequent words and (c) length distribution of the captions

I will create 2 dimentional representation of the space using PCA and visualize the distribution of the sample images. To see whether the photo features extracted from VGG16 network makes sense, we manually selected similar images that are creating clusters. For instance, we created red: many people are in one image , green: dogs on green, yard or on bed, magenta: dogs in snow or with water splash, blue: guys doing sports with helmets, yellow: not sure what these are. From each cluster, we plotted the original images. Figure 6, images from the same clusters tend to be very similar. Figure 7 shows the clustered images.

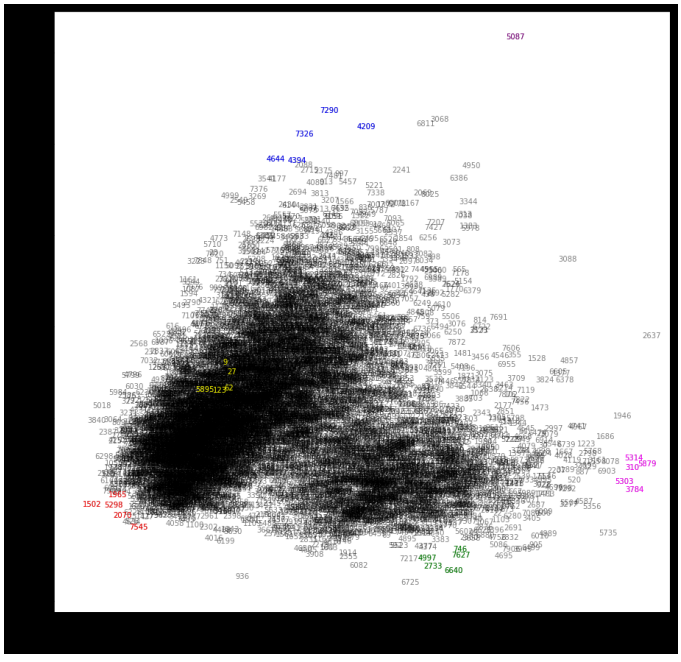

Figure 6: PCA embedding of the clustered images

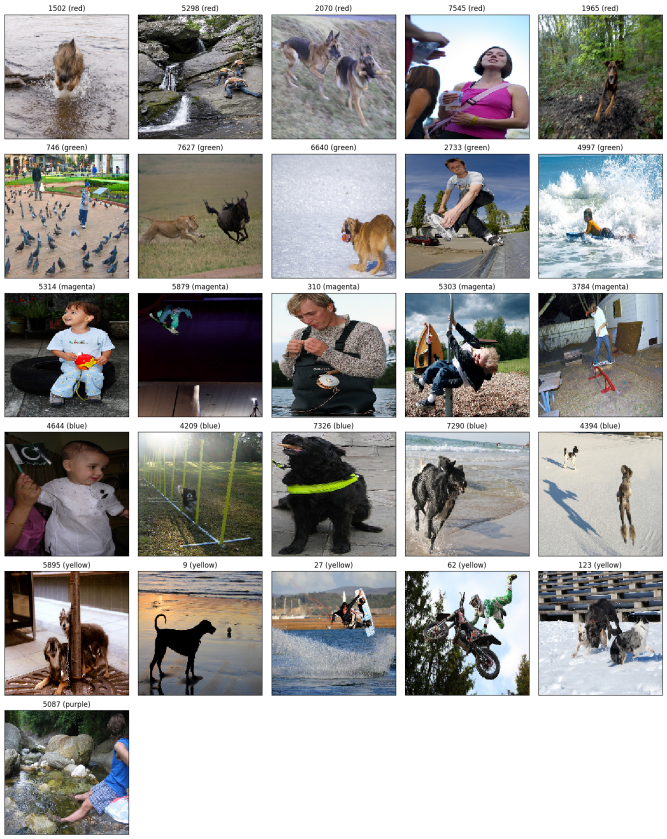

Figure 7: The sample images of red, green, magenta, blue, yellow clusters

In this dataset, a single image has 5 captions. We will only use one caption out of 5 for simplicity. Each row of the dtexts and dimages contain the same info. Remove captions (or images) that do not have corresponding images (or captions). The final vocabulary size is 4476. Then we change character vector to integer vector using tokenizer. We make a 60, 20, 20 split between training, validation and test set of the data.

3.4 Final Model:

This final model takes two inputs: a) 4096-dimensional image features from pre-trained VGG model, (b) tokenized captions up to *t* th word. The single output is tokenized *t+1* the word of caption.
Prediction: Given the caption prediction up to the *t* th word, the model can predict the t+1st word in the caption, and then the input caption can be augmented with the predicted word to contain the caption up to the *t+1* th word. The augmented caption up to the *t+1* st word can, in turn, be used as input to predict the *t+2* nd word in caption. The process is repeated until the "endseq" is predicted. The summary of the model is as following:

- There is 4096-dimensional image features from pre-trained VGG model. The image feature is passed to fully connected layer with 256 hidden units.

Table 1: Model Hyperparameter

Hyperparameter	value
Layer size	128
Image dropout	no
Embedding dropout	yes
RNN dropout	no
Minibatch size	128
Max. epochs	Earlystopping at 5

- The tokenized caption up to *t* th time point is passed to embedding layer where each word is represented with a dimension embedding dimensional vector. This means that a single caption is represented as many time series of length maximum length. The time series are passed to LSTM with 256 hidden states, and then a single output at the final time point is passed to the higher layer.
- The networks are merged by simply adding the two vectors of size 256.
- The vector of length 256 is passed to two dense layers and the final dense layer return probability that the *t+1* st word is kth word in the vocabulary (k=1,...,4476).

While training the model, we used the hyperparameters (Table 1) stated in the reference paper [12]. But while experimenting with the model, we found that over fit very quickly. This makes sense considering the small size of our data. So stopped the training at 5 epoch. Figure shows the training vs. validation loss over epochs.

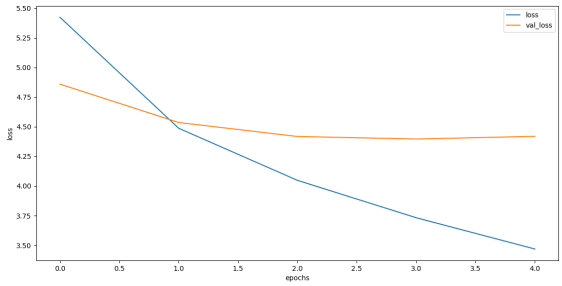

Figure 8: Validation loss and training loss over epochs

4 EXPERIMENT

4.1 Pre-trained Word Vectors

We experimented with word embedding layers of the model. Initializing word vectors with those obtained from an unsupervised neural language model is a popular method to improve performance in the absence of a large supervised training set. We use the publicly available glove 200d word vectors [10], unsupervised learning algorithm for obtaining vector representations for words. Words not present in the set of pre-trained words are initialized randomly.

4.2 Model Variations

We experiment with several variants of the model.

	RNN-rand	RNN-static	RNN-non-static
BLEU-4	0.226 (0.145)	0.246 (0.153)	0.235 (0.132)

- RNN-rand Our baseline model where all words are randomly initialized and then modified during training.
- RNN-static A model with pre-trained vectors from Glove. All words including the unknown ones that are randomly initializedâĂŤare kept static and only the other parameters of the model are learned.
- RNN-non-static Same as above but the pre-trained vectors are fine-tuned for each task.

5 EVALUATION

To evaluate the different architectures, the test set captions (which are shared among all architectures) are used to measure the architectures' quality using metrics that fall into three classes, described below.

Generation metrics: These metrics quantify the quality of the generated captions by measuring the degree of overlap between generated those in the test set. We use BLEU-4 automated metric to evaluate all the models.

From the Table 2 above, we see that the RNN-static performs better than RNN-rand model. Also the RNN-non-static model performs slightly better than the RNN- random model. Though we expected that the RNN-non-static model outperform the other two by large margin, we think that as we used 50% dropuout rate at embedding layer, the model under fitted and the hence the result was affected.

5.1 Word Embedding Output

Table 3 shows the word embedding layer output of the three different models. It shows that the RNN-rand model output is quite different from the two other. The RNN-static and the RNN-non-static models are quite similar. But there are some variation like the top 5 words of RNN-static model has the word *Green* for the word *Girl* whereas the model RNN-non-static has the word *Dress* in that position. So the variations in word embedding layer is working.

6 QUALITATIVE ANALYSIS

Figure 9 shows some of the example output for three of the models. Figure 9 (a) shows that the RNN-rand creates caption with repetitive phrases whereas the other two models create captions with sentence where there is no problem like this. Again in 9 (b), RNN-rand creates simple caption whereas other two models included small details like the color, wearing of the subject. So those models is capable of integrating details and good at predicting appropriate action verbs than the random one. Finally in 9 (c), generates caption with which is incomplete and makes no sense. Whereas two other models create captions which atleast captures the essence of the main theme of the image. So the RNN-non-static model surely improves the caption quality in some aspects compared to other models.

7 CONCLUSION

We introduced models that manipulates the word embedding layer of the language model of the image caption system. The first variation uses a pre-trained word embedding layer and the second one fine tune the embedding to the task specific words. Our varied models show somewhat better performance than the RNN-rand model. Though The quantitative metric does not show that much improvement, the qualitative analysis captures improvements in several aspects. Human judgements on these results can give us better idea about the performance of these models. And also fine tuning of the models can improve the performance further.

Table 2: Top 5 neighboring wordsâĂŤbased on cosine similarity

Word	RNN-rand	RNN-static	RNN-non-static
Girl	Endseq	Two	Two
	People	Water	Water
	Jumping	Green	Dress
	green	Dress	Back
	children	area	area
Swimming	and	water	water
	two	red	red
	water	an	race
	group	race	area
	field	area	arms
White	on	water	water
	down	looks	small
	up	trick	looks
	yellow	dress	dress
	her	boat	boat

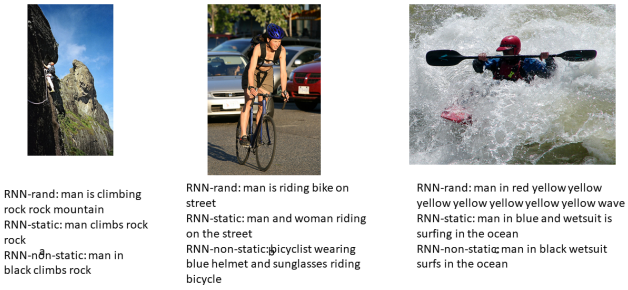

Figure 9: Example output

REFERENCES

[1] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, March 2003.

[2] Jacob Devlin, Hao Cheng, Hao Fang, Saurabh Gupta, Li Deng, Xiaodong He, Geoffrey Zweig, and Margaret Mitchell. Language models for image captioning: The quirks and what works. In *ACL*, 2015.

[3] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, abs/1411.4389, 2014.

[4] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012.

[5] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. In *J. Artif. Intell. Res.*, 2013.

[6] Yoon Kim. Convolutional neural networks for sentence classification. *CoRR*, abs/1408.5882, 2014.

[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. In *CVPR 2011*, pages 1601–1608, June 2011.

[8] S. Liu and W. Deng. Very deep convolutional neural network based image classification using small training sample size. In *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*, pages 730–734, Nov 2015.

[9] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan Yuille. Deep captioning with multimodal recurrent neural networks (m-rnn). 12 2014.

[10] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[11] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. *CoRR*, abs/1409.3215, 2014.

[12] Marc Tanti, Albert Gatt, and Kenneth P. Camilleri. Where to put the image in an image caption generator. *CoRR*, abs/1703.09137, 2017.

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

[14] Cheng Wang, Haojin Yang, Christian Bartz, and Christoph Meinel. Image captioning with deep bidirectional lstms. *CoRR*, abs/1604.00790, 2016.

[15] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. Boosting image captioning with attributes. *CoRR*, abs/1611.01646, 2016.