Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2017"
Moscow, May 31—June 3, 2017

NEURAL LANGUAGE MODEL FOR IMAGE CAPTIONING

НЕЙРОСЕТЕВЫЕ ЯЗЫКОВЫЕ МОДЕЛИ ДЛЯ АННОТАЦИИ ИЗОБРАЖЕНИЙ

Alexey Mastov (alexei.mastov@skolkovotech.ru)
Viktoriya Malyasova (viktoriya.malyasova@skolkovotech.ru)

Skoltech, Moscow, Russia

Annotation Automatically generating descriptions of the content of an image is a fundamental problem in artificial intelligence that connects computer vision and natural language processing. We experiment with a language model based on a LSTM neural network that combines recent advances in computer vision and machine translation. We compare several word embeddings in our model. We also examine functions of individual LSTM cells and find some interesting patterns.

Keywords: image captioning, deep learning, text generation, recurrent neural network, LSTM.

Introduction

Image captioning is the problem of automatically describing the content of an image in natural language. Its solution would have many applications: it can boost search engines by allowing them to analyze picture's content as well as text content, and improve web experience for visually impaired people. This task is hard because we need to not only identify the objects on the picture, but also their relationships to each other.

Most earlier attempts Kulkarni *et al.* (2011); Farhadi *et al.* (2010) glued together existing solutions for object recognition and natural language modeling. In contrast, we present a joint model that is trained to maximize likelihood p(caption|image), like in the work of Vinyals *et al.* (2014).

A usual neural language model uses a neural network to predict probability of the next word given previous words. The loss function usually used for language models is perplexity:

$$Perplexity(w_1, \ldots, w_n) = \sqrt[n]{\prod_{i=1}^{n} \frac{1}{P(w_i|w_1, \ldots, w_{i-1})}}$$

Perplexity is an estimate of categorical crossentropy between the probability distribution learned from the training set and the probability distribution of the test set.

In image captioning pipeline, predictions are based both on previous words and on the image, so probability of the next word is defined as follows:

$$P(w_i|w_1, \ldots, w_{i-1}, image)$$

1 Our model

The general architecture of our model is as in the paper by Vinyals *et al.* (2014): a convolutional neural network extracting image features, followed by a recurrent neural network generating sentences. The architecture is illustrated on Figure 1.

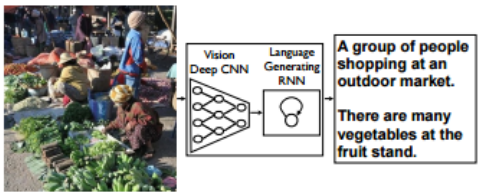

Figure 1: Overall architecture of our model. CNN is used for extraction image features, which describe context of the scene. Then its output is used by RNN-based sentence generator. Illustration from Vinyals *et al.* (2014).

For learning language model we use LSTM neural network, which has shown state-of-the-art performance on language modeling tasks. In our implementation we have 512 LSTM units. The LSTM architecture is the standard one, suggested by Graves (2013).

For extracting features from images we use GoogleNet Szegedy *et al.* (2014), which showed state-of-the-arta performance for classification and detection in the ImageNet Large-Scale Visual Recognition Challenge 2014. It extracts 1000 features from each image.

The difference between our model and that of Vinyals *et al.* (2014) is that for representation of words, instead of one-hot encodings, we use word embeddings. In our experiments we tested a word2vec model, GloVe model and Lasagne EmbeddingLayer.

Word2vec (Mikolov *et al.* (2013)) uses a fully-connected neural network with one hidden layer to learn an embedding of words in a low-dimensional space. We use the continuous bag of words model.

GloVe is another word embedding model suggested by pennington2014glove.

Embedding Layer is a layer that maps one-hot-encoded word vectors in a lower dimensional space. The mapping is learned during the training, so the word embedding is tuned to our specific task of image captioning.

2 Experiments

In our experiments we used COCO (Common Objects in Context) 2014 dataset, which contains around 128,000 images annotated with 5 captions each.

During the study, we wanted to check how different embeddings influence performance. We tried two pre-trained models: a pretrained model glove.6B (trained on 6 billion tokens corpus from Wikipedia) and a word2vec model trained on our data. We used 200 and 300-dimensional embeddings for both. Also we tried Lasagne EmbeddingLayer.

For evaluation, we used traditional metrics for evaluating translation quality: BLUE, METEOR, ROUGE and CIDEr. They are also used for benchmarking image captions.

3 Results

We summarise our results in Table 1. Surprisingly, pre-trained embeddings showed worse result, compared to EmbeddingLayer. In official COCO leader board our implementation takes 44th place.

Some of the captions generated by our model are shown on Figure 2.

4 Visualization

It turns out that individual LSTM cells have interpretable functions in caption generation. For example, one cell in our model is responsible for generating gerunds. On Figure 3 we can see that it triggers on *holding, standing, riding* etc.

Table 1: Final results of our experiments. First line is given for reference as one of the closest results in o□cial leader board, where our method is on 44th position. In bold are best results among all our implementations.

Models	Metrics						
	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE	CIDEr
Karpathy et Fei-Fei (2015)	0.65	0.464	0.321	0.224	0.21	0.475	0.674
w2v-coco-200	0.633	0.442	0.296	0.198	0.194	0.458	0.587
w2v-coco-300	0.634	0.444	0.299	0.201	0.197	0.461	0.603
w2v-glove-200	0.632	0.444	0.298	0.199	0.195	0.461	0.583
w2v-glove-300	0.638	0.451	0.306	0.205	0.197	0.464	0.602
emb-layer-200	0.647	0.461	0.316	0.216	0.206	0.473	0.657
emb-layer-300	**0.649**	**0.463**	**0.318**	**0.219**	**0.207**	**0.474**	**0.666**

Figure 2: Captions generated by our model. Image in red shows failure case.

Another cell seems to understand the concept of ownership and is active on such phrases as *with a, holding a*. We give its outputs on Figure 4

Some other cells are content-sensitive. For instance, a cell gave out values very close to 1 ($>$0.99) for every word of a caption for bathrooms, toilets and kitchens, and smaller values for other types of images.

5 Conclusion

In this project we successfully implemented image captioning pipeline using Theano and Lasagne. Our implementation shown competitive results to ones

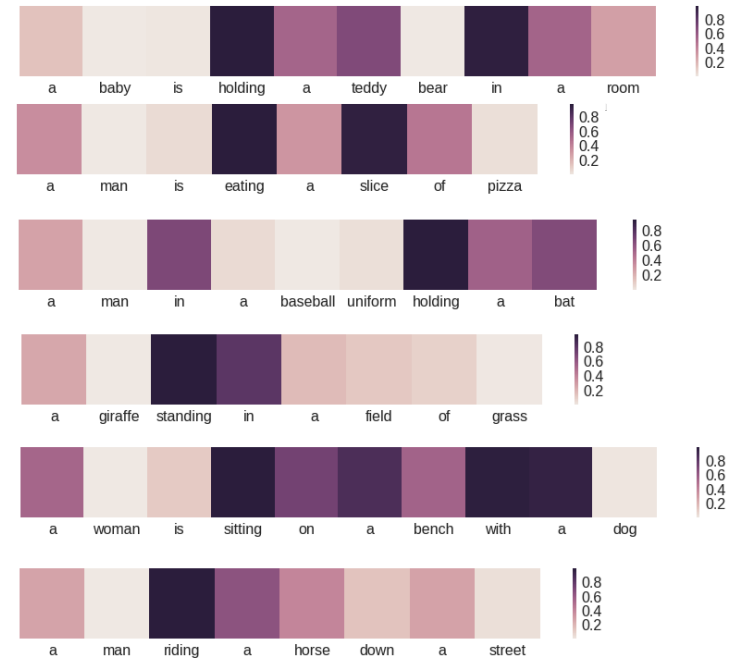

Figure 3: Color-encoded output of a cell, which turned out to be responsible for gerund generation. We can see that it triggers on *standing, holding, riding* etc.

with the similar architecture (for example Karpathy et Fei-Fei (2015)) and took 44th place in open COCO leader board.

Another our contribution lies in experiments. We compared three different embeddings and figured out that Lasagne Embedding Layer showed the best performance. Also we were able to study semantic meaning of some cells and had interesting and interpretable results.

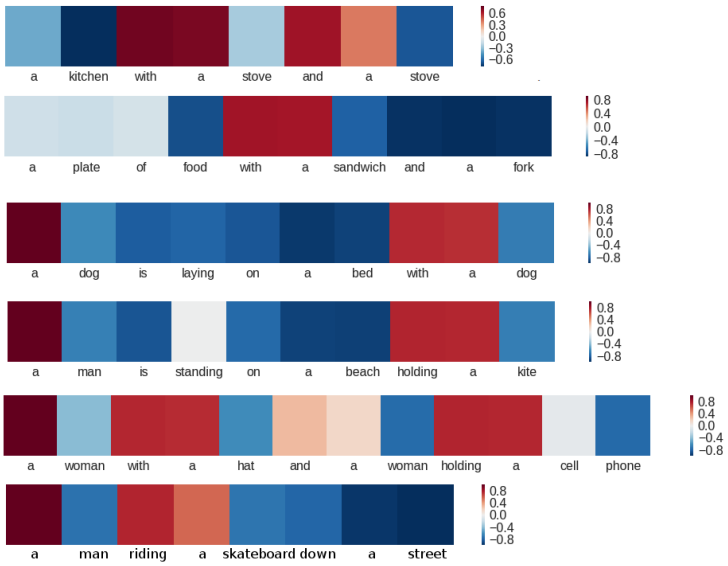

Figure 4: Color-encoded output of a cell, which turned out to be responsible for *with a, holding a* phrases.

Bibliography

Chen, X., Fang, H., Lin, T., Vedantam, R., Gupta, S., Dollár, P. et Zitnick, C. L. (2015). Microsoft coco captions: Data collection and evaluation server. *arXiv:1504.00325*.

Farhadi, A., Hejrati, M., Sadeghi, M. A., Young, P., Rashtchian, C., Hockenmaier, J. et Forsyth, D. (2010). *Every Picture Tells a Story: Generating Sentences from Images*, pages 15–29. Springer Berlin Heidelberg, Berlin, Heidelberg.

Graves, A. (2013). Generating sequences with recurrent neural networks. *CoRR*, abs/1308.0850.

Karpathy, A. et Fei-Fei, L. (2015). Deep visual-semantic alignments for generating image descriptions. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3128–3137.

Kulkarni, G., Premraj, V., Dhar, S., Li, S., Choi, Y., Berg, A. et Berg, T. (2011). *Baby talk: Understanding and generating simple image descriptions*, pages 1601–1608.

Mikolov, T., Chen, K., Corrado, G. et Dean, J. (2013). Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Pennington, J., Socher, R. et Manning, C. D. (2014). Glove: Global vectors for word representation. *In EMNLP*, volume 14, pages 1532–1543.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V. et Rabinovich, A. (2014). Going deeper with convolutions. *CoRR*, abs/1409.4842.

Vinyals, O., Toshev, A., Bengio, S. et Erhan, D. (2014). Show and tell: A neural image caption generator. *CoRR*, abs/1411.4555.