






# Predicting Commentaries on a Financial Report with Recurrent Neural Networks

Karim El Mokhtari<sup>(✉)</sup> , John Maidens<sup></sup>, and Ayse Bener<sup></sup>

Data Science Laboratory, Ryerson University, Toronto, Canada

[elmkarim@ryerson.ca](mailto:elmkarim@ryerson.ca)

<https://www.ryerson.ca/dsl>

**Abstract.** Aim: The paper aims to automatically generate commentaries on financial reports. Background: Analysing and commenting financial reports is critical to evaluate the performance of a company so that management may change course to meet the targets. Generating commentaries is a task that relies on the expertise of analysts. Methodology: We propose an encoder-decoder architecture based on Recurrent Neural Networks (RNN) that are trained on both financial reports and commentaries. This architecture learns to generate those commentaries from the detected patterns on data. The proposed model is assessed on both synthetic and real data. We compare different neural network combinations on both encoder and decoder, namely GRU, LSTM and one layer neural networks. Results: The accuracy of the generated commentaries is evaluated using BLEU, ROUGE and METEOR scores and probability of commentary generation. The results show that a combination of one layer neural network and an LSTM as encoder and decoder respectively provides a higher accuracy. Conclusion: We observe that the LSTM highly depends on long term memory particularly in learning from real commentaries.

**Keywords:** NLP · Recurrent Neural Networks · LSTM · GRU

## 1 Introduction

Companies generate financial reports for both internal and external purposes. Financial results show the past, current and future performance of a company. Financial reports are generated from the daily sales transactions, inventories, cash flows, supplier transactions, etc. They are critical to evaluate the liquidity, profitability, and capital adequacy of the company. Companies make budgets to set targets for sales, revenues, expenses, assets and liabilities on an annual basis. Depending on the company policy these targets are compared against the actual numbers in different frequencies (i.e. weekly, monthly, quarterly, etc.) and if necessary adjustments are made to targets or some actions are taken in various departments to meet these targets.

Financial analysts within the company periodically study the variances between the actuals and the budget numbers at many levels and they provide

management with insights. This task often requires access to many datasets related to different areas (finance, logistics, planning, inventory, etc.) to explain the variance. The analyst creates a summary report that relates each variance with a short commentary that serves as a baseline for top management to take immediate actions.

In this paper we propose an approach for generating commentaries by learning from analysts reports. The proposed algorithm is trained on a dataset of variance and the corresponding commentaries created by analysts. The use of commentaries written in English has many advantages. First, it provides a clear understanding of the actual circumstances of the business in the form of a limited set of sentences and fosters immediate actions especially in a rapidly-changing market. Second, it accelerates the analysis by providing a quick feedback for the learned patterns. Finally, in the long-term, it reduces the cost and time of the analysis by involving the human analyst only in ambiguous cases.

The business considered in this paper is a consumer goods company making and selling hundreds of brands for customers worldwide. The products are grouped into brands, and the brands are then grouped into categories.

The monthly report generated by the company’s Enterprise Resource Planning (ERP) system includes the sales details in millions of dollars for each product and customer. Table 1 shows a typical report where the variance is the difference between the forecast and the actual results per product and customer.

**Table 1.** Sample from the monthly report generated by the company’s ERP. Numbers are in Millions of CAD

Customer	Product category	Product brand	Product name	Forecast	Actual	Variance
Customer 1	Category 1	Brand 1	Product 1	2.5	2.4	−0.1
Customer 1	Category 1	Brand 1	Product 2	0.5	0.7	+0.2
...	...	...	...	...	...	...
Customer 32	Category 35	Brand 52	Product 15	0.8	0.75	−0.05
...	...	...	...	...	...	...

The monthly report created by the analyst is shown on Table 2. The sales figures are summed up by brand and compared with the Forecast. The analyst writes his comments to explain the observed variance.

**Table 2.** Sample from the monthly commentaries report generated by the expert. Numbers are in Millions of CAD

Brand	Forecast	Actual	Variance	Commentary
Brand 1	4.75	5.25	+0.50	The variance is driven by Customer 1 and Customer 2, caused by over delivery
...	...	...	...	...

Generating commentaries is a complex process and requires an extensive experience and meticulous data analysis. This process is time consuming and entails high costs for the company. It needs multi-disciplinary experts and data sources of different natures: Point of Sale (POS) transactions, inventory, manufacturing and marketing to name a few. One downside of this process is its reliance on the expert that may not be consistent over time or might occasionally miss some critical information. Such errors could incur important losses to the company and miss significant opportunities for growth. Our aim is to build a learning based model that is able to understand patterns in data residing in different silos of the organisation and generate commentaries that are as close as possible to the ones created by the experts.

This paper is organised as follows. The second section describes the state of the art in caption generation, the third section describes the learning model. The experimental part is presented in the fourth section and deals with synthetic and real data. The conclusion summarises the findings and presents the prospects.

## 2 Related Work

Commentary generation is a process where numerical data is encoded and transformed into natural language. This concept is similar in its nature to many problems found in the literature such as caption generation for images and language translation. Many of these methods rely on Recurrent Neural Networks that were applied to sequence to sequence models in language translation [1,2]. We are motivated by this encoder-decoder structure as it translates data between different representations through a context vector linking both ends of the architecture. In our case, we encode financial data into vector representation then we translate it to language form by the decoder.

This same concept has been applied to create captions for images [3]. In this context, retrieval-based methods use neural networks to map image and text into a vector representation [4] or use similarity metrics applied on predefined image features [5]. Mao et al. [6] use a language model that relies on a recurrent neural network instead of a feed-forward based model. Vinyals et al. [7] use Long Short Term Memory (LSTM) networks to generate captions for an image from a pre-trained Convolutional Neural Network (CNN). Donahue et al. [8] apply a similar structure based on LSTM to caption a video sequence. Karpathy and Li [9] follow a different approach in captioning an image that consists of learning through a multimodal embedding space in a model that uses a bidirectional RNN over sentences and a R-CNN over image regions. There is a comparative study that applies different captioning methods namely Visual-Back [11], Guiding-LSTM [12], ShowTell [7] and DeepSemantic [9] to describe car images [10].

## 3 Learning Model

An overview of the method is illustrated in Fig. 2. For each month, data is aggregated by brand and customer resulting in the pivot table  $V$  shown on

Fig. 1. In this matrix denoted  $V$ , an element  $v(i, j)$  is the variance of Brand  $i$  for Customer  $j$  for the given month. More specifically, an element  $v(i, j)$  is the difference between the sum of the forecast and the sum of the actual of all products belonging to Brand  $i$  and delivered to Customer  $j$ . The row vector  $V^{(i)}$  contains the values of the variance of all customers for Brand  $i$ . As shown in Fig. 1, the expert may write a comment like the one created for brand 2.

During training, matrix  $V$  is parsed row by row and each row vector  $V^{(i)}$  along with the corresponding analyst commentary  $C^{(i)}$  is fed into the learning model. The model learns to predict the next word of the commentary jointly from the current word and the variance vector of the Brand  $i$ .

	Customer 1	Customer 2	...	Customer j	...	
Brand 1	+0.10	+0.11	...	-0.01	...	
Brand 2	-0.13	-0.09	...	+0.27	...	
...						
Brand i	+0.02	+0.03	...	+0.14	...	← $V^{(i)}$
...						

Commentaries
No comment
Promotion did well for Customer j
...
No comment
...

**Fig. 1.** Matrix  $V$  generated for one month. An element of  $V$  denoted  $v(i, j)$  - highlighted on the matrix - is the variance of Brand  $i$  for Customer  $j$ . The row vector  $V^{(i)}$  is the variance of all customers for brand  $i$ . (Color figure online)

We propose to use an encoder-decoder architecture similar to [10] but with both encoder and decoder defined as LSTM [13] or GRU [3]. The LSTM network is an RNN that is designed to handle vanishing and exploding gradient issues through the use of forget, input and output gates. The GRU network is a simplified version of the LSTM that uses only two gates to solve the vanishing gradient problem called update gate and reset gate. We also propose to use a simple neural network as encoder.

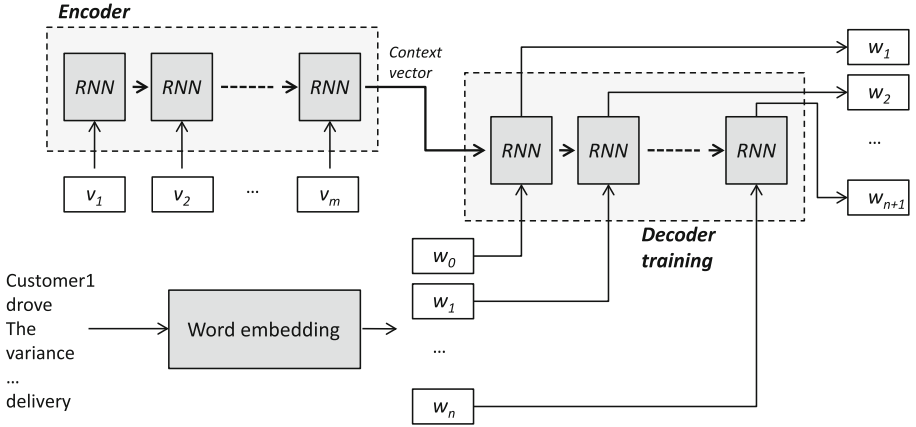
The commentary for brand  $i$  noted  $C^{(i)}$  is a sequence of  $n$  words such that:

$$C^{(i)} = \{w_0^{(i)}, w_1^{(i)}, \dots, w_{n+1}^{(i)}\} \quad (1)$$

$w_0^{(i)}$  and  $w_{n+1}^{(i)}$  are special tokens that mark respectively the START and the END of the commentary. The ideal commentary is the one that maximises the probability  $p(C^{(i)}|V^{(i)})$  for a given variance vector  $V^{(i)}$ . This probability is defined with a joint probability as follows:

$$\log p(C^{(i)}|V^{(i)}) = \sum_{k=2}^n \log p(w_k^{(i)}|V^{(i)}, w_1^{(i)}, \dots, w_{k-1}^{(i)}) \quad (2)$$

The encoder RNN converts the variance vector  $V^{(i)}$  into a fixed-size context vector serving as an initial value for the decoder. The chain nature of Eq. 2 is modelled by the decoder RNN. This RNN maps the current word and the



**Fig. 2.** Learning model architecture: the model consists in an encoder and decoder both implemented by a Recurrent Neural Network depicted unrolled

previous cell state that includes all past information learned up to the current step to the output vector. This vector is applied to a softmax classifier [14] that calculates the probability of each word in the dictionary. The objective function is defined as:

$$\theta = \arg \max_{\theta} \sum_{(C^{(i)}, V^{(i)})} \log p(C^{(i)} | V^{(i)}; \theta) \quad (3)$$

where  $\theta$  is the model parameter that contains all RNN parameters and word embedding. The objective function can be reformulated as a loss function  $L(C, V^{(i)} | \theta)$  to be used in neural network training:

$$L(C^{(i)}, V^{(i)} | \theta) = - \sum_{(C^{(i)}, V^{(i)})} \sum_{k=1}^n \log p_k(w_k^{(i)} | V^{(i)}; \theta) \quad (4)$$

During training, the pair variance vector for brand  $i$  and the corresponding commentary  $(V^{(i)}, C^{(i)})$  is fed to the model. The words of the commentary  $C^{(i)}$  are applied to the decoder RNN input at each time step. We use the START token as first input to predict the first word, the commentary words are then applied sequentially to the decoder until the END token is reached. We use the RMSprop optimizer [15] to accelerate the learning. This method divides the learning rate for each weight in the model by a running average of the magnitudes of recent gradient for that weight thus allowing a fast convergence. This optimizer is usually a good choice for recurrent neural networks [16].

## 4 Experiment

The experimental part includes model testing on synthetic and real data. The real dataset is proprietary and is provided by our research partner. In both

experiments, we use the following evaluation metrics: BLEU (BiLingual Evaluation Understudy) [17], ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [18] and METEOR (Metric for Evaluation of Translation with Explicit Ordering) [19]. BLEU score is widely used to assess the quality of machine translation models. It evaluates the quality of text which has been translated from a natural language to another by comparing the machine’s output with that of a human. BLEU’s output is a number between 0 and 1 that indicates the similarity between the reference and generated text, with values closer to one representing more similar texts. Clarity of grammatical correctness are not taken into account. For each commentary, we compute four scores BLEU-1, BLEU-2, BLEU-3 and BLEU-4 that compare the generated commentary with the human created one. BLEU-1 score considers only the precision at an unigram level, while BLEU-4 considers all four-grams.

While BLEU is a precision-based score used in the machine translation community, ROUGE is a recall-based from the summarization community. It compares the overlapping n-grams, words sequences and word pairs. In our evaluation, we use the ROUGE-L version. ROUGE metric penalises short sentences as it relies on recall [20]. METEOR is used in machine translation and calculates the harmonic mean of precision and recall of unigram matches between the reference and the generated sentences. It also applies synonyms and paraphrase matching for implicit word matching. We use the package nlgeval in [21] to calculate ROUGE and METEOR scores.

We also assess the model in terms of predicting when a commentary needs to be generated. In this context, we want to evaluate whether the model has learned to detect specific patterns on the variance vector  $V^{(i)}$  that triggers the commentary generation. This is a binary classification problem where a positive output stands for a nonempty commentary and a negative output for an empty commentary. We use accuracy and F1-score [22] as evaluation metrics. The accuracy is the number of correct predictions to the total number of input samples. In our case, it refers to the ratio of the generated commentaries to the total number of non empty commentaries or the commentary generation probability. F1-score is the harmonic mean of the precision and recall [22] defined by Eq. 5. Precision and Recall are respectively defined by  $TP/(TP + FP)$  and  $TP/(TP + FN)$ , where  $TP$  is the number of True Positives and  $FN$  the number of False Negatives.

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \quad (5)$$

In both experiments, we compare different networks for both encoder and decoder, namely GRU and LSTM. We also test a one layer neural network as an encoder denoted NN that receives  $V^{(i)}$  and encodes it into an initial value for the encoder.

### 4.1 Results with Synthetic Data

In the synthetic case, we consider that the analyst generates a commentary whenever one or many values in the variance vector  $V^{(i)}$  are over a threshold that we define here as  $\pm 0.5$ . Therefore, we initialise the values of  $V^{(i)}$  according to a uniform distribution between  $-0.5$  and  $+0.5$ , then we choose randomly one to three customers and bring their variance outside the range  $[-0.5, +0.5]$ . A graphic visualisation of  $V^{(i)}$  is shown on Fig. 3. We consider three types of commentaries depending on the number of customers with a high variance. Some samples do not includes any high variance, in such case, en empty commentary is generated.

We train the model by applying at each time step the synthetic variance vector  $V^{(i)}$  for brand  $i$  to the encoder and the corresponding synthetic commentary  $C^{(i)}$  to the decoder. Commentaries are tokenized and applied to a word embedding neural network. We generate 23562 commentaries and we run the simulations for 100 epochs with a batch size of 20.

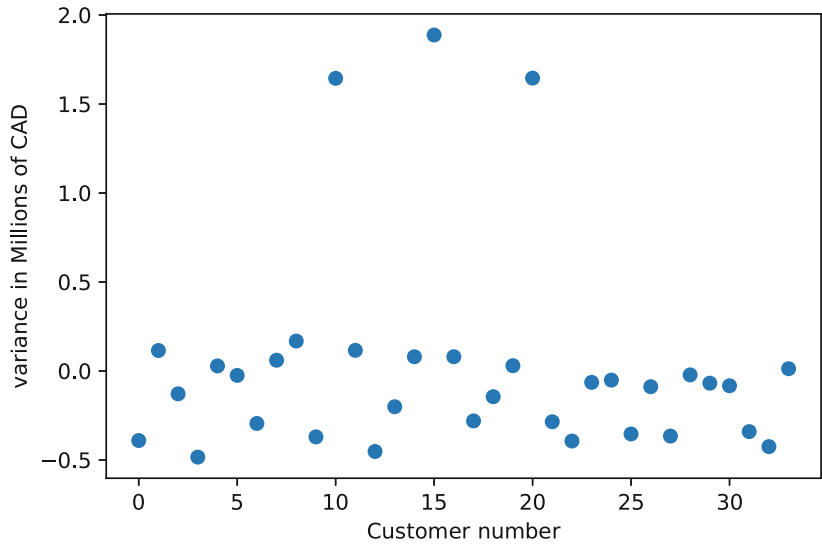
**Table 3.** BLEU, ROUGE, METEOR scores with synthetic data

Enc./Dec.	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	METEOR
GRU/GRU	0.769	0.626	0.563	0.438	0.814	0.367
NN/GRU	0.546	0.405	0.370	0.273	0.552	0.255
NN/LSTM	0.526	0.390	0.360	0.269	0.558	0.249

Table 3 shows that the highest BLEU, ROUGE and METEOR scores are obtained with the architecture GRU/GRU. The measured BLEU-1 score for this architecture is 0.769 and shows that almost 77% of the words generated in the commentaries were identical to the synthetic ones, while 43.8% of the four-grams were identical. The ROUGE score shows 81.4% of similarity by measuring the recall over the longest common subsequences between reference and predicted sentences. METEOR is also the highest for the GRU/GRU architecture but it is lower than BLEU and ROUGE scores as it takes into account the order of the matched unigrams of the predicted commentary with respect to the human generated one.

Table 4 shows the architecture NN/GRU (NN as encoder and GRU as decoder) provides the best prediction regarding generating a commentary according to the rules above mentioned. Indeed, in predicting that a commentary is needed for a given variance vector, the accuracy of this architecture was as high as 0.863 with a F1-score of 0.906. This score shows a higher rate of true positives compared to the two other architectures.

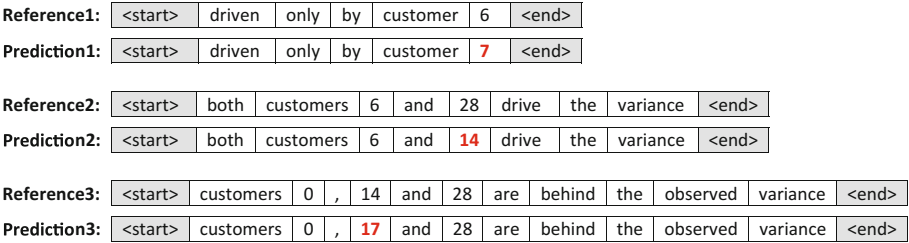
Figure 4 shows samples of reference and predicted commentaries using the GRU/GRU architecture. In general, the model is able to detect when the variance is associated with one, two or three customers although sometimes the customer numbers are not always correctly predicted.



**Fig. 3.** Sample of a variance vector  $V^{(i)}$  for brand  $i$  where three customers have a high variance between actual and forecast

**Table 4.** Commentary generation metrics with synthetic data

Enc./Dec.	Accuracy	F1-score
GRU/GRU	0.769	0.438
NN/GRU	0.863	0.906
NN/LSTM	0.838	0.889



**Fig. 4.** Examples of reference and predicted commentaries in the experiment with synthetic data

4.2 Results with Real Data

We test the model with real data where patterns are more complex than in the synthetic case and the content of a commentary depends on the analyst experience. The dataset consists of 1330 commentaries. We apply stemming to reduce



**Table 5.** BLEU, ROUGE, METEOR scores with real data

Enc./Dec.	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	METEOR
GRU/GRU	0.172	0.076	0.067	0.041	0.470	0.122
NN/GRU	0.367	0.279	0.283	0.234	0.594	0.237
NN/LSTM	0.399	0.306	0.310	0.258	0.615	0.252

**Table 6.** Commentary generation metrics with real data

Enc./Dec.	Accuracy	F1-score
GRU/GRU	0.685	0.605
NN/GRU	0.828	0.816
NN/LSTM	0.844	0.830

the dictionary size before tokenizing the comments. As in the synthetic case, we compare different architecture and measure their performance in terms of BLEU, ROUGE and METEOR scores and commentary generation probability.

The results on Tables 5 and 6 show that the NN/LSTM architecture performs best. Indeed, the BLEU-1 score with this architecture is the higher reaching 0.399. This means that almost 40% of the predicted words are identical to the original commentaries. The BLEU-4 score is also the highest and reaches 0.258. Thus, 25.8% of the four-grams in the generated commentaries are identical to the original commentaries. ROUGE and METEOR scores are also the highest. This performance may be explained by the greater dependency of the LSTM on long term memory which is more needed in real commentaries than on synthetic ones.

Regarding the commentaries generation, the NN/LSTM was 84.4% accurate in predicting that a commentary needs to be generated. An F1-score of 83% indicates a higher rate of true positives compared to the other architectures.

Figure 5 shows samples of the predicted commentaries compared with the real ones. The first predicted sample is different from the source, while in the second one, the model was able to predict the customer name but unable to completely predict the commentary. In the last sample, the prediction and reference correspond exactly.

By comparing synthetic and real results, we notice the higher performance of the GRU decoder over the LSTM decoder with synthetic data. This can be interpreted by the similarity between commentaries generated synthetically. Indeed, we created four different classes of commentaries where the only difference within the same class was the customers number. On the other hand, in the real case, the similarity is low between commentaries and learning relies more on long term relation between words in the same commentary. This may explain the higher performance of the LSTM over the GRU network. In fact, the LSTM uses a more complex structure with four gates which allows a better dependency with the past compared to the GRU network.

Reference1:	<start>	brand	growth	<end>			
Prediction1:	<start>	overlay	did	not	materialize	<end>	
Reference2:	<start>	customer1	strong	baseline	on	region1	<end>
Prediction2:	<start>	customer1	overseas	order	<end>		
Reference3:	<start>	anticipated	volume	did	not	materialize	<end>
Prediction3:	<start>	anticipated	volume	did	not	materialize	<end>

**Fig. 5.** Examples of reference and predicted commentaries in the experiment with real data. As the dataset is the property of our research partner, some words may not be identical to the real dataset and real customer names were replaced by customer 1, 2 and 3.

## 5 Threats to Validity

In this paper, we note the following threats to validity:

- In the real dataset, in a high number of cases, the analyst decides not to generate any commentary. This results in an imbalanced dataset that would create a wrongly high accuracy. To mitigate the impact of this fact, we combined undersampling of empty commentaries and over sampling of non empty commentaries. In addition, we used F1-score as a second evaluation metric.
- Over-fitting is a problem that was noted during training on both synthetic and real data. We mitigated this problem by using random validation sets and by changing batch sizes.
- A highly diversified vocabulary used in the commentaries impacts negatively the accuracy and convergence of the algorithm. We reduced the impact of such a problem by stemming words and replacing all number by a unique token. The prediction of numbers is out of the scope of this work.

## 6 Conclusion

In this paper, we propose an encoder-decoder architecture that aims at generating commentaries from a dataset reporting the variance between forecast and actual for a consumer good company. The commentaries are originally created by financial analysts depending on their observation of the variance dataset. We train the architecture on synthetic and real commentaries and we compare different variants of the architecture. We assess the performance of the model using BLEU, ROUGE and METEOR scores to compare the accuracy of the generated commentaries. We also assess the probability of generating a commentary by learning from the analyst. We show that an LSTM decoder associated with a one layer neural network encoder performs better than a GRU network in the

real case. The GRU, on the other hand is more accurate in the synthetic case. This difference may be explained by the high variance among real commentaries that needs a long term dependency in the learning process. The more complex four-gate structure of the LSTM allows a better long-term dependency, and provides a better accuracy in commentary generation. The prospects of this work include comparing the encoder-decoder model with an attention mechanism. In such model, the interaction between both entities does not rely only on the context vector but extends the interaction to learning the dependency between each word and all financial data. Such model may allow to learn more complex patterns between brands and between variances over time.

**Acknowledgement.** This work is supported by a grant from Smart Computing For Innovation (SOSCIPI) consortium, Toronto, Canada.

## References

1. Sutskever, I., Vinyals, O., Le, Q.V.V.: Sequence to sequence learning with neural networks. In: NIPS, pp. 3104–3112 (2014)
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. [arXiv:1409.0473](https://arxiv.org/abs/1409.0473) (2014)
3. Cho, K., et al.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1724–1734 (2014)
4. Farhadi, A., et al.: Every picture tells a story: generating sentences from images. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 15–29. Springer, Heidelberg (2010). [https://doi.org/10.1007/978-3-642-15561-1\\_2](https://doi.org/10.1007/978-3-642-15561-1_2)
5. Hodosh, M., Young, P., Hockenmaier, J.: Framing image description as a ranking task: data, models and evaluation metrics. JAIR **47**, 853–899 (2013)
6. Mao, J., Xu, W., Yang, Y., Wang, J., Yuille, A.: Deep captioning with multimodal recurrent neural networks (M-RNN). [arXiv:1412.6632](https://arxiv.org/abs/1412.6632) (2014)
7. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell, a neural image caption generator. [arXiv:1411.4555](https://arxiv.org/abs/1411.4555) (2014)
8. Donahue, J., et al.: Long-term recurrent convolutional networks for visual recognition and description. [arXiv:1411.4389v2](https://arxiv.org/abs/1411.4389v2) (2014)
9. Karpathy, A., Li, F.-F.: Deep visual-semantic alignments for generating image descriptions. [arXiv:1412.2306](https://arxiv.org/abs/1412.2306) (2014)
10. Chen, L., He, Y., Fan, L.: Let the robot tell: describe car image with natural language via LSTM. Pattern Recogn. Lett. **98**, 75–82 (2017)
11. Fang, H., et al.: From captions to visual concepts and back. In: Proceedings of the IEEE Conference on Computer Vision Pattern Recognition, pp. 1473–1482 (2015)
12. Jia, X., Gavves, E., Fernando, B., Tuytelaars, T.: Guiding long-short term memory for image caption generation. In: Proceedings of the ICCV (2015)
13. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
14. Bridle, J.S.: Probabilistic interpretation of feedforward classification network outputs with relationships to statistical pattern recognition. In: Soulié, F.F., Héroult, J. (eds.) Neurocomputing: Algorithms, Architectures and Applications, pp. 227–236. Springer, Heidelberg (1990). [https://doi.org/10.1007/978-3-642-76153-9\\_28](https://doi.org/10.1007/978-3-642-76153-9_28)

15. Tieleman, T., Hinton, G.: Lecture 6.5-RMSprop: divide the gradient by a running average of its recent magnitude. COURSERA 4(2), 26–31 (2012)
16. Keras Documentation. <https://keras.io/optimizers/>. Accessed 28 Jan 2019
17. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: ACL (2002)
18. Lin, C.Y.: Rouge: a package for automatic evaluation of summaries. In: Text Summarization Branches Out: Proceedings of the ACL 2004 Workshop, pp. 74–81 (2004)
19. Elliott, D., Keller, F.: Image description using visual dependency representations. In: EMNLP, pp. 1292–1302. ACL (2013)
20. Kilickaya, M., Erdem, A., Ikizler-Cinbis, N., Erdem, E.: Re-evaluating automatic metrics for image captioning. In: Proceedings of EACL 2017, pp. 199–209 (2017)
21. Sharma, S., El Asri, L., Schulz, H., Zumer, J.: Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. CoRR, vol. abs/1706.09799 (2017). <http://arxiv.org/abs/1706.09799>
22. Dangeti, P.: Statistics for Machine Learning, 1st edn. Packt Publishing Ltd., Birmingham (2017)