Show and Tell: Reimplementing the Google NIC image captioning model

Ruochong Chen
University of Michigan - Ann Arbor
Ann Arbor, MI
ruochong@umich.edu

1. Introduction

Image captioning, the task of generating descriptive and coherent textual descriptions for images, has been a compelling area of research at the intersection of computer vision(CV) and natural language processing (NLP).

In this paper, we contribute by re-implementing and investigating the techniques used in the Google NIC model[16]. This model is mainly consists of a pretrained CNN(Convolutional Neural Network)[12] and LSTM(Long Short Term Memory) model[13]. On top of that, we also experimented different strategies attempting to improve the results of the model without making significant changes to the model's architecture.

2. Related Work

Before the NIC model, there were also similar works that attempts to generate text captions for visual data. The paper "Understanding Videos, Constructing Plots Learning a Visually Grounded Storyline Model from Annotated Videos"[7] attempted to model the plot/storyline of videos as an AND-OR graph consisted of AND and OR nodes and generates the description of the video based on the graph made by chaining each event node. The paper "Exploring Nearest Neighbor Approaches for Image Captioning"[5] exploited the fact that many generated captions are identical to captions found in the training dataset, and proposed to find the k nearest neighbor images given an image feature and generate the caption based on a consensus mechanism similar to RANSAC[6].The paper "Every Picture Tells a Story: Generating Sentences from Images"[3] tried to map both the image space and sentence space to a triplet of (object,action,scene) as an intermediate representation, with each node having a set of candidate words. The paper "Composing Simple Image Descriptions using Web-scale N-grams"[10] proposed a method that uses 21 image attributes and generates captions through phrase selection and n-gram phrase fusion.

Compared to the NIC model, all those previous methods tried to predict captions for an image using previously seen captions or with handcrafted visual features, while the NIC model directly uses features learned by CNN.

3. Method

3.1. Model

Based on the original paper, this model follows an encoder-decoder structure, where an image is encoded as the initial hidden state of the lstm, and the resulting caption is decoded from the starting word and encoded hidden state. The model uses a pretrained CNN(GoogLeNet/inceptionv1[15] according to the original paper) to extract rich features from the image, and then encode the feature vector with an LSTM. Figure 1 illustrates the architecture of this model.

Mathematically speaking, this model aims to maximize the probability of predicting the corresponding descriptions S given an image I with model parameters θ through this formulation:

$$\theta^* = arg\max_{\theta} \sum_{(I,S)} \log p(S|I;\theta)$$

The log probability of predicting the entire sentence S, which consists of N tokens, can be unrolled into a joint probability of predicting the next token S_t at time step t given the image I and tokens from previous time steps $S_0, ..., S_{t-1}$, for all the time steps from 0 to N. This gives us the following formulation:

$$\log p(S|I) = \sum_{t=0}^{N} \log p(S_t|I, S_0, ..., S_{t-1})$$

We can model this formulation with an RNN(Recurrent Neural Network), where the information we captured from time step 0 to timestep t-1 is stored in the memory m_t. In the case of an LSTM, the memory is consists of 2 components: hidden state h_t and cell state(forget) c_t, which allows the model to better retain the long term dependencies in the text captions.

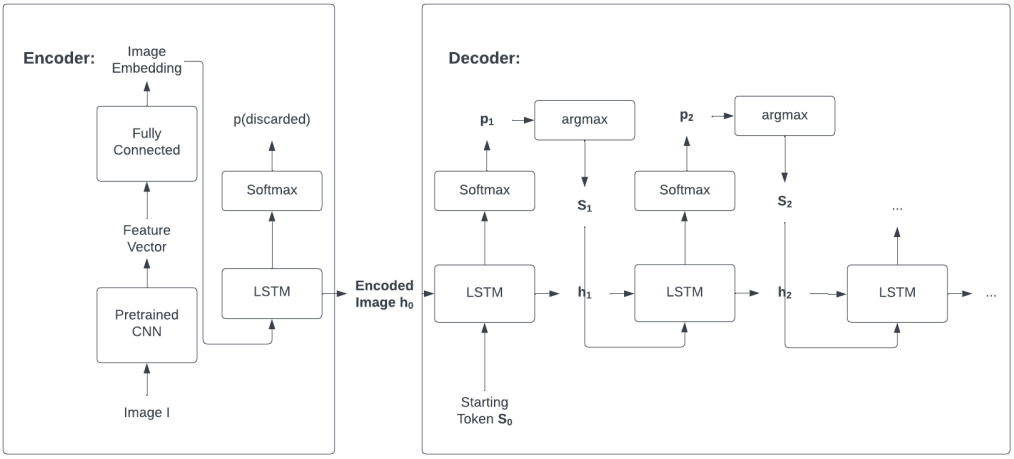

Figure 1: Model Architecture

3.2. Data

Due to limited computing resources and time, this project only trained the model on the Flickr8k dataset[8], which contains 8000 images and 40000 text captions, with 5 captions describing each image.

The image feature vectors were extracted by replacing the last layer of CNNs pretrained on classification tasks(GoogLeNet) with identity matrix, and all captions were tokenized with the BertTokenizer[4] provided by Huggingface's Transformers library[17] and padded to a length of 45. The resulting image/text feature vectors were saved to disk for future reuse.

For training an LSTM, we feed in the target sequence, and ask it to predict the right shifted target sequence. Suppose the original sentence S is consists of $S_0, ..., S_N$, we feed the input sequence $S_0, ..., S_{N-1}$ to the lstm and compare the prediction with target sequence $S_1, ..., S_N$ so that the lstm can learn to predict the next token for each token in one caption.

After preprocessing, the custom dataset class we built will yield a tuple of (image feature vector, input sequence vector, target sequence vector), where all the text sequence vectors are padded to a length of 45.

3.3. Training

The baseline model for this project is trained on Flick8k's training set, which contains 6000 images and 30000 text captions.

The model aims to minimize the negative log likelihood of predicted captions, so we used CrossEntropyLoss for training/validation as it combines softmax and NLLLoss.

The baseline model is trained with the following hyperparameters:

The model doesn't seem to converge even after 500 epochs(see Figure 2), and the training process has been extremely time consuming so I trained a new baseline model for 500 epochs using the Adam optimizer with lr=1e-4. The training/validation plot is shown in Figure 3, where we can observe the same overfitting issue encountered by the original paper, where the validation loss started to rise dramatically after around 60-80 epochs.

CNN	Batch size	Embed Dim	Hidden Dim	Optimizer	Learning Rate
GoogLeNet	256	512	512	SGD	1e-2

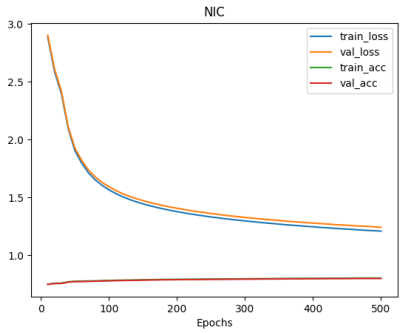

Figure 2: Baseline with SGD

To combat the overfitting problem, we used the early stopping with a patience of 5, which has saved us a lot of time for training the models and prevented the overfitting issue.

3.4. Inference

This project implemented the two appraoches for inference mentioned in the original paper: Greedy Sampling and Beam Search.

Greedy Sampling: Starting with the first start-of-sequence token, sample each word S_t with the highest softmax probability score, and then pass S_t and the previous

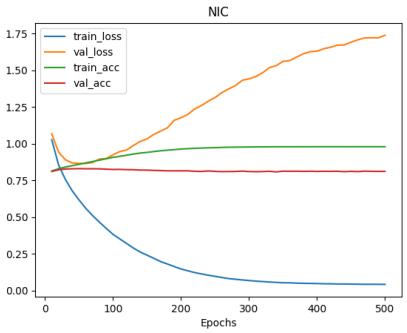

Figure 3: Baseline with Adam

memory m_t into the model's decoder to generate new softmax probability scores for next sampling step. Run this algorithm until encounter end-of-sequence token or reached max sequence length.

Beam Search: Starting with the start-of-sequence token and score 0 in the priority queue, select k predicted words with the top k probability score, add the log probability score to the score of the previous sequence, and push each resulting sequence to the priority queue with their priority being their normalized cumulative score. When end-of-sequence token is encountered in a sequence, put that into the result set, and terminate the loop until the result set has the desired size N_C or max iteration N is reached, and return the first sequence in the priority queue. The sequences are ordered by normalized cumulative scores to prevent choosing solely from shorter sequences.

The implementation of priority queue based beam search in this project referenced the "Best First Beam Search" paper from MIT[11].

We chose a beam size(k) of 10 instead of 20 for the sake of time. While we didn't observe too much of difference in performance between greedy search and beam search, we choose beam search as a default in later evaluations according to the original paper.

3.5. Metrics

We chose to evaluate our models based on BLEU(BiLingual Evaluation Understudy) 1-4[14] and METEOR(Metric for Evaluation of Translation with Explicit Ordering)[9] scores. Our baseline model achieved the following scores:

	BLEU				METEOR
	1	2	3	4	
Greedy Search	43	21	5	2	29
Beam Search	43	14	5	2	23
Original Paper[18]	66	42	27	18	-

This baseline model's performance is, apparently, unsatisfying due to its failure to converge(in a short time) with the SGD optimizer. However, in the following section, I will discuss how we managed to achieve much better performance with various experiments.

4. Experiments

4.1. Batch normalization

Adding batch norm for image embedding increased the BLEU score by several points, but the model still wasn't able to converge in 500 epochs. We will be using batch normalization in the subsequent experiments.

	BLEU				METEOR
	1	2	3	4	
Baseline	43	14	5	2	23
Batch norm	42	21	8	3	24

4.2. SGD vs Adam

Replacing SGD with Adam optimizer with a learning rate of 1e-4 resulted in a much faster convergence when training the model. On top of the much faster training speed, the performance of the model has increased drastically with Adam optimizer due to the convergence, which was close to the original paper.

	BLEU				METEOR
	1	2	3	4	
SGD_{BN}	42	21	8	3	24
Adam_{BN}	57	36	20	11	37

4.3. GoogLeNet(Inceptionv1) vs Inceptionv3

We believe that Inceptionv3 will give better representation of image features as the successor of GoogLeNet, and it did resulted in a slightly better performance, so we chose to use Inceptionv3 for subsequent experiments.

4.4. Embedding & Hidden size

We also experimented several sets of combination of text embedding size and LSTM hidden layer size, and we found that embed size = 768 with hidden size = 512 yielded the best performance, and increasing hidden layer size caused a much slower training speed.

Embed	Hidden	BLEU				METEOR
		1	2	3	4	
512	512	57	36	20	11	37
768	512	59	38	23	12	39
1024	512	58	37	21	11	39
1024	768	59	38	22	11	39

4.5. Pretrained text embeddings

Surprisingly, initializing our model's word embedding layer with BERT's pretrained embedding matrix(since we

are using Bert's tokenizer and vocabulary) resulted in a worse performance then without pretrained embeddings. This might be caused by a mismatch of target task.

	BLEU				METEOR
	1	**2**	**3**	**4**	
No pretrained	59	38	23	12	39
Pretrained	56	37	21	11	37

4.6. Image feature extractors

We also tried different models for image feature extraction. Pretrained Inceptionv3, Resnet101, VGG19, ViT are provided by the torchvision package, and pretrained CLIP was provided by the Huggingface Transformers library.

Model	BLEU				METEOR
	1	**2**	**3**	**4**	
Inceptionv3	59	38	23	12	39
Resnet101	58	37	21	11	38
VGG19_BN	56	35	20	10	36
ViT	53	30	16	8	33
CLIP	64	44	28	16	44

embed size = 768, hidden size = 512

While the configuration of embed size = 768, hidden size = 512 yielded the best performance for the Inceptionv3 baseline model, we also tried a configuration of embed size = 1024, hidden size = 512 out of curiosity.

Model	BLEU				METEOR
	1	**2**	**3**	**4**	
Inceptionv3	58	37	21	11	39
Resnet101	59	38	22	12	39
VGG19_BN	56	36	20	11	38
ViT	53	31	17	8	34
CLIP	67	48	30	17	46

embed size = 1024, hidden size = 512

Not surprisingly, using CLIP to extract image features yielded an exceptional performance that far exceeds that of the original paper, while the image vectors it generated had the smallest dimensionality(512). This can be caused by the fact that CLIP is specifically trained on learning joint embeddings of image and text, and can be transferred to suit the task of image captioning very easily.

We use the model trained on image features extracted by CLIP with embed size = 1024 and hidden size = 512 and call it our **"Final Model"**.

4.7. Qualitative results

Figure 4 also shows some qualitative results. Images were randomly chosen from the testing set and their captions are generated with the Final Model. We can see that the generated captions seemed pretty reasonable.

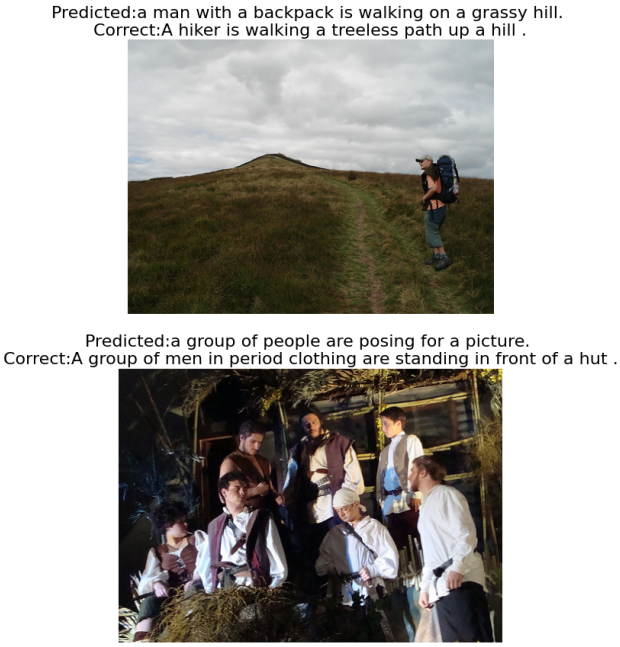

Figure 4: Qualitative Results

4.8. Transfer Learning

We also finetuned our trained Final Model on Shinkai Makoto dataset[1] and Pokemon dataset[2] from Huggingface to see the model's generalizability, achieving decent results on these smaller datasets(around 1000 images).

Model	BLEU				METEOR
	1	**2**	**3**	**4**	
Shinkai	49	41	31	21	50
Pokemon	58	51	44	39	56

5. Conclusion

We have reimplemented NIC, a Neural Image Captioning model, published by Google in 2015, and built a complete pipeline including data preprocessing, training, testing, and inference. As a result, we successfully reproduced a result close to what the original paper presented with our converged models.

Throughout the experiments conducted in this project, we were able to produce a final model that drastically outperforms the original paper's result, and we also confirmed the difficulties the original paper encountered.

Due to the time and resource constraint, we were not able to replicate a lot of things the original paper presented like evaluating ranking results, CIDER metric, and training on larger datasets. In the future, we can try more modern techniques such as adding in attention mechanism, replacing LSTM decoder with a transformer-based decoder like BERT or GPT, training on much larger datasets.

References

[1] Fung804/makoto-shinkai-picture. https://huggingface.co/datasets/Fung804/makoto-shinkai-picture/viewer/default/train. Accessed: December 1, 2023. 4

[2] lambdalabs/pokemon-blip-captions. https://huggingface.co/datasets/lambdalabs/pokemon-blip-captions. Accessed: December 1, 2023. 4

[3] Mohammad Amin Sadeghi Peter Young Cyrus Rashtchian1 Julia Hockenmaier David Forsyth Ali Farhadi, Mohsen Hejrati. Every picture tells a story:generating sentences from images, 2010. 1

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018. 2

[5] Jacob Devlin, Saurabh Gupta, Ross Girshick, Margaret Mitchell, and C. Lawrence Zitnick. Exploring nearest neighbor approaches for image captioning, 2015. 1

[6] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981. 1

[7] Abhinav Gupta, Praveen Srinivasan, Jianbo Shi, and Larry S. Davis. Understanding videos, constructing plots learning a visually grounded storyline model from annotated videos. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2012–2019, 2009. 1

[8] Micah Hodosh, Peter Young, and Julia Hockenmaier. Flickr8k dataset. 2

[9] Alon Lavie and Abhaya Agarwal. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, page 228–231, USA, 2007. Association for Computational Linguistics. 3

[10] Siming Li, Girish Kulkarni, Tamara L Berg, Alexander C Berg, and Yejin Choi. Composing simple image descriptions using web-scale n-grams. In Sharon Goldwater and Christopher Manning, editors, *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 220–228, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. 1

[11] Clara Meister, Tim Vieira, and Ryan Cotterell. Best-first beam search, 2020. 3

[12] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. 1

[13] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. 1

[14] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, page 311–318, USA, 2002. Association for Computational Linguistics. 3

[15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. 1

[16] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator, 2014. 1

[17] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2019. 2

[18] Qi Wu, Chunhua Shen, Anton van den Hengel, Peng Wang, and Anthony Dick. Image captioning and visual question answering based on attributes and external knowledge, 2016. 3