Computer Vision and Deep Learning for Automated Surveillance Technology

Teun de Planque
Department of Computer Science
Stanford University
teun@stanford.edu

Abstract

The number of surveillance cameras around the world has increased rapidly over the last couple of years [1]. Currently most of the video material generated by these cameras is examined manually [1]. However, automatic processing and analysis could significantly increase safety, since as a result of this automation security teams can be alerted within a shorter period of time. In this paper, computer vision and deep learning techniques are used to analyze images, and create text descriptions of images that can be send to the police or the owner of the security camera. Specifically, a convolutional neural network is used to extract image features, principal component analysis is used to reduce the dimensionality of the image features, and the resulting image features are then fed into a recurrent neural network or a long short-term memory to generate a text description. Different convolutional neural network architectures are evaluated based on their effectiveness to extract image features for image descriptions. This paper demonstrates that deeper convolutional neural networks (16-18 depth layers) extract more useful image features than less deep convolutional neural networks (8, 10, 12 layers). Our models generate high quality image descriptions, and obtain comparable to state-of-the-art performance on the image description generation task.

1. Introduction

The proliferation of digital image surveillance technology has changed the security sector. Surveillance cameras are used all over the world and they record billions of hours of security footage each year [1]. This increased number of security video material creates both opportunities and new challenges. On the one hand, the larger number of surveillance cameras can serve as deterrence, and have the potential to give us more information about crimes. On the other hand, it is challenging or even impossible for humans to effectively analyze such a large amount of video material, and as a result new methods have to be developed to analyze surveillance camera video material.

Figure 1: Computer vision techniques can improve security by automatically warning the police when detecting suspicious behavior.

Currently, most surveillance camera video analysis is done manually, and the video analysis is generally done after the occurrence of a crime. Analysis of the video material in retrospect can help the police identify criminals, learn more about the details of the crime, and prevent similar crimes from happening in the future. Nonetheless, with software that can analyze the video material in real-time (i.e. while the crime is happening), police can take immediate action. This can significantly improve safety, increasing the overall usefulness of surveillance cameras.

Computers vision is essential for the effective automation of surveillance camera video material analysis. A major theme in the field of computer vision has been to duplicate the abilities of human vision by electronically perceiving and understanding an image. In the case of surveillance camera material it is essential to understand the content of the surveillance camera image input and turn this image input into a simplified representation that can be used to take action if necessary. In this paper, computer vision techniques are used to extract image features that can then be turned into a description of the image (Figure 1). This image description can then be sent to the owner of the surveillance camera, a security organization, or the police. The owner of the surveillance camera, a security organization, or the police can then decide to take action based on this image description. Being able to automatically describe the content of an image using properly formed English sentences is a very challenging task. This task is significantly harder, for example, than the well-studied image classification or object recognition tasks, which have been a main focus in the computer vision community. Indeed, a description must capture not only the objects contained in an image, but it also must express how these objects relate to each other as well as their attributes and the activities they are involved in. Moreover, the above semantic knowledge has to be expressed in a natural language like English, which means that a language model is needed in addition to visual understanding.

In this paper, a convolutional neural network (CNN) is used to extract features from the security camera generated images. CNNs have been shown to be powerful models for image classification and object detection tasks. Different convolutional neural network architectures are evaluated based on their ability to extract image features for image descriptions. To extract image features the CNNs are first trained for the image classification task. After this training phase, the CNNs are used to extract image features. The activations of the second-to-last fully connected layer are used as image features. Principal component analysis is then used to reduce the dimensionality of the feature vector, while still retaining as much of the variance in the dataset as possible. The resulting features vector is then fed into a recurrent neural network (RNN) or a long short-term memory (LSTM). The RNN and LSTM generate an image description based on the image features. The recurrent structures allow the RNN and LSTM to exhibit dynamic temporal behavior, which is essential for the generation of correct English sentences. Especially, the LSTM is well suited to the generation of long sentences since it can remember values for very long durations. Our models generate high quality image descriptions, and obtain comparable to state-of-the-art performance on the image description generation task.

2. Review of Previous Work

2.1. Overview

Most work in visual recognition has originally focused on image classification, i.e. assigning labels corresponding to a fixed number of categories to images. Great progress in image classification has been made over the last couple of years, especially with the use of deep learning techniques [2, 3]. Nevertheless, a category label still provides limited information about an image. Some initial attempts at generating more detailed image descriptions have been made, for instance by Farhadi et al. and Kulkarni et al. [4, 5], but these models are generally dependent on hard-coded sentences and visual concepts. In addition, the goal of most of these works is to accurately describe the content of an image in a single sentence. However, this one sentence requirement unnecessarily limits the quality of the descriptions generated by the model. Several works, for example by Li et al., Gould et al., and Fidler et al., focused on obtaining a holistic understanding of scenes and objects depicted on images [6, 7, 8, 9]. Nonetheless, the goal of these works was to correctly assign labels corresponding to a fixed number of categories to the scene type of an image, instead of generating higher-level explanations of the scenes and objects depicted on an image.

Generating sentences that describe the content of images has already been explored. Several works attempt to solve this task by finding the image in the training set that is most similar to the test image and then returning the caption associated with the test image [4, 10, 11, 12, 13]. Jia et al., Kuznetsova et al., and Li et al. find multiple similar images, and combine their captions to generate the resulting caption [14, 15, 16]. Kuznetsova et al., and Gupta et al. tried using a fixed sentence template in combination with object detection and feature learning [5, 17, 18]. They tried to identify objects and features contained in the image, and based on the identified objects contained in the image they used their sentence template to create sentences describing the image. Nevertheless, this approach greatly limits the output variety of the model.

Recently there has been a resurgence of interest in image caption generation, as a result of the latest developments in deep learning [2, 19, 20, 21, 22]. Several deep learning approaches have been developed for generating higher level word descriptions of images [21, 22]. Convolutional Neural Networks have been shown to be powerful models for image classification and object detection tasks. In addition, new models to obtain low-dimensional vector representations of words such as word2vec, and GloVe (Global Vectors for Word Representation) and Recurrent Neural Networks can together create models that combine image features with language modeling to generate image descriptions [21, 22]. Karpathy et al. developed a Multimodal Recurrent Neural Network architecture that uses inferred alignments to learn to generate novel descriptions of image regions [21]. Similarly, Kiros et al. used a log-bilinear model that generates full sentence descriptions for images [22]. However, their model uses a fixed window context [22].

2.2. Contributions

Concretely, our contribution is threefold:

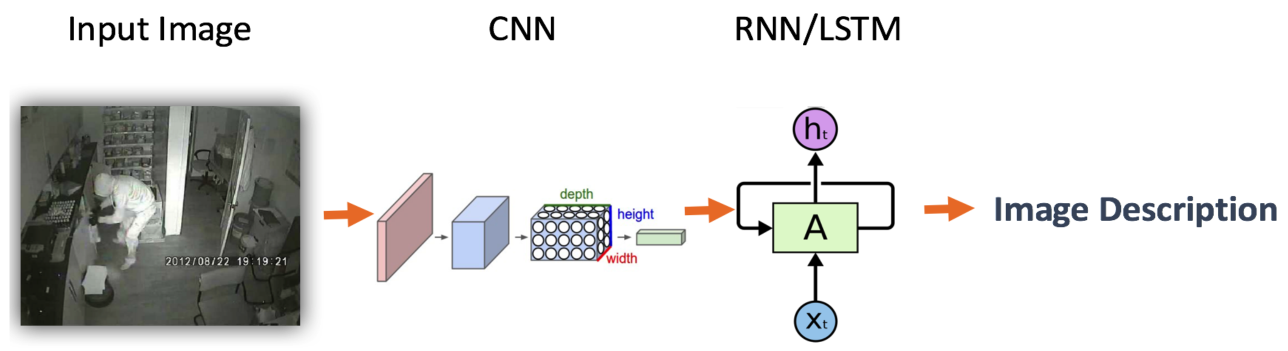


Figure 2: Overview of technical approach.

1. Different CNN layer patterns (i.e. different network depths) are evaluated on the image description generation task.
2. The performance of the RNN and LSTM for the image description generation task are compared for several CNN implementations.
3. Close to state-of-the-art results are achieved with the presented models.

In addition, the full image description generation deep learning pipeline is implemented without relying on external deep learning packages such as Tensorflow, and instead using only fundamental python packages for scientific computing (e.g. numpy, scipy).

3. Technical Approach

Overview. We implemented a deep recurrent architecture that automatically produces short descriptions of images (Figure 2). Our models use CNNs to obtain image features. We then feed these features into either an RNN or a LSTM network (Figure 3) to generate a description of the image in valid English.

3.1. CNN-based Image Feature Extractor

For feature extraction, we use a CNN. CNNs have been widely used and studied for images tasks, and are currently state-of-the-art methods for object recognition and detection [20]. CNNs consist of a sequence of layers; every layer of a CNN transforms one volume of activations to another through a differentiable function [21]. There are four main types of layers to build CNN architectures: convolutional layers, pooling layers, RELU layers, and fully-connected layers. Convolutional layers compute the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region that they are connected to in the input volume [21]. Pool layers will perform a down-sampling operation along the spatial dimensions (i.e. width and height) [21]. RELU layer will apply an element-wise activation function, such as the $max(0, x)$ thresholding at zero. Finally, fully-connected layers compute the class scores. As with ordinary neural networks, each neuron in this layer will be connected to all the numbers in the previous volume.

We pre-train the CNN on the dataset for the image classification task. We then keep the weights of the network, and for each image extract features from the second-to-last fully connected layer of the network as described by Karpathy et al [21]. This gives us a 4096-dimensional image feature vector that we reduce using principal component analysis (PCA) to a 512-Dimensional image feature vector due to computational constraints. We feed these features into the first layer of our RNN or LSTM at the first iteration [24].

For the pre-training phase we use the image category labels contained in our dataset. During the pre-training phase our goal is to obtain a high accuracy on the image classification task, so that our CNN is able to correctly assign labels for a fixed number of categories to images. We use back-propagation to update the weights in the network, so that the network learns to detect features in the images that can be used for image classification or in our case image description generation.

After the pre-training phase we can use the pre-trained CNN to detect image features for the image description generation task. As described by Karpathy et al. image features obtained by a convolutional neural network pre-trained on the image classification task, can also be used for the image description generation task [21]. Similar to Donahue et al. we extract image features from the second-to-last layer from the CNN, and then perform PCA to reduce the dimension of the image features because of computational constraints [36]. We designed our CNNs so that the image

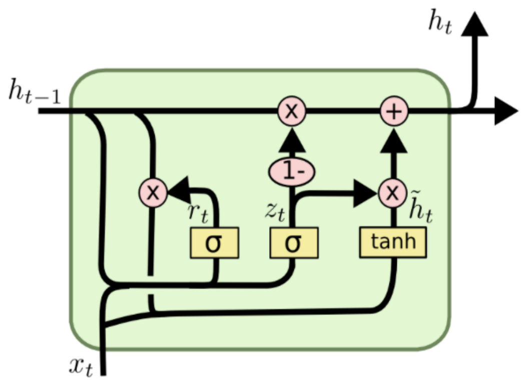


Figure 3: Example of an LSTM unit and its gates

features of the second-to-last fully connected layer are of dimension 4096. Using PCA we replace the high-dimensional (4096-dimensional) data by its projection onto its 512 most important axes, so that the resulting feature vector is 512-dimensional. These axes are the axes corresponding to the largest eigenvalues of the covariance matrix.

We experiment with several CNN network depths, i.e. CNNs of depth 6, 8, 10, 12, 14, and 16. With depth we refer to the total number of convolutional and fully connected layers. For each of the network depths we create one model using the RNN to generate sentences, and one model using the LSTM to generate sentences.

3.2. RNN-based Sentence Generator

We first experiment with vanilla RNNs as they have been shown to be powerful models for processing sequential data [25, 26]. Vanilla RNNs can learn complex temporal dynamics by mapping input sequences to a sequence of hidden states, and hidden states to outputs via the following recurrent equations.

$$h_t = f(W_{hh}h_{t-1} + W_{xh}x_t) \tag{1}$$

$$y_t = W_{hy}h_t \tag{2}$$

where f is an element-wise non-linearity, $h_t \in \mathbb{R}^N$ is the hidden state with N hidden units, and y_t is the output at time t. In our implementation, we use a hyperbolic tangent as our element-wise non-linearity. For a length T input sequence $x_1, x_2, ..., x_T$, the updates above are computed sequentially as h_1 (letting $h_0 = 0$), y_1, h_2, y_2,... h_T, y_T.

3.3. LSTM-based Sentence Generator

Although RNNs have proven successful on tasks such as text generation and speech recognition [25, 26], it is difficult to train them to learn long-term dynamics. This problem is likely due to the vanishing and exploding gradients problem that can result from propagating the gradients down through the many layers of the recurrent networks. LSTM networks (Figure 3) provide a solution by incorporating memory units that allow the networks to learn when to forget previous hidden states and when to update hidden states when given new information [24].

At each time-step, we receive an input $x_t \in \mathbb{R}^D$ and the previous hidden state $h_{t-1} \in \mathbb{R}^H$, the LSTM also maintains an H-dimensional cell state, so we also get the previous cell state $c_{t-1} \in \mathbb{R}^H$. The learnable parameters of the LSTM are an input-to-hidden matrix $W_x \in \mathbb{R}^{4HxD}$, a hidden-to-hidden matrix $W_h \in \mathbb{R}^{4HxH}$, and a bias vector $b \in \mathbb{R}^{4H}$.

At each time step, we compute an activation vector $a \in \mathbb{R}^{4H}$ as

$$a = W_x x_t + W_h h_{t-1} + b \tag{3}$$

We then divide a into 4 vectors a_i, a_f, a_o, $a_g \in \mathbb{R}^H$ where a_i consists of the first H elements of a, a_f is the next H elements of a, etc.. We then compute four gates which control whether to forget the current cell value $f \in \mathbb{R}^H$, if it should read its input $i \in \mathbb{R}^H$, and whether to output the new cell value $o \in \mathbb{R}^H$, and the block input $g \in \mathbb{R}^H$.

$$i = \sigma(a_i) \tag{4}$$

$$f = \sigma(a_f) \tag{5}$$

$$o = \sigma(a_o) \tag{6}$$

$$g = tanh(a_g) \tag{7}$$

where σ is the sigmoid function and $tanh$ is the hyperbolic tangent; both are applied element-wise.

Finally, we compute the next cell state c_t which encodes knowledge at every time step of what inputs have been observed up to this step, and the next hidden state h_t as

$$c_t = f \circ c_{t-1} + i \circ g \tag{8}$$

$$h_t = o \circ tanh(c_t) \tag{9}$$

where $\circ$ represents the Hadamard product. The inclusion of these multiplicative gates permits the regulation of information flow through the computational unit, allowing for more stable gradients and long-term sequence dependencies [24]. Such multiplicative gates make it possible to train the LSTM robustly as these gates deal well with exploding and vanishing gradients. The non-linearities are sigmoid $\sigma()$ and hyperbolic tangent $tanh()$.

Our LSTM model takes the image I and a sequence of input vectors $(x_1, ..., x_T)$. It then computes a sequence of hidden states $(h_1, ..., h_t)$ and a sequence of outputs $(y_1, ..., y_t)$ by following the recurrence relation for $t = 1$ to T:

$$b_v = W_{hi}[CNN(I)] \tag{10}$$

$$h_t = f(W_{hx}x_t + W_{hh}h_{t-1} + b_h + 1(t = 1) \circ b_v) \tag{11}$$

$$y_t = Softmax(W_{oh}h_t + b_o) \tag{12}$$

where W_{hi}, W_{hx}, W_{hh}, W_{oh}, x_i, b_h, and b_o are learnable parameters and $CNN(I)$ represents the image features extracted by the CNN.

3.4. Training.

We train our LSTM model to correctly predict the next word (y_t) based on the current word (x_t), and the previous context (h_{t-1}). We do this as follows: we set $h_0 = 0$, x_1 to the $START$ vector, and the desired label y_1 as the first word in the sequence. We then set x_2 to the word vector corresponding to the first word generated by the network. Based on this first word vector and the previous context the network then predicts the second word, etc. The word vectors are generated using the word2vec embedding model as described by Mikolov et. al [27]. During the last step, x_T represent the last word, and y_T is set to an END token.

3.5. Testing.

To predict a sentence, we obtain the image features b_v, set $h_0 = 0$, set x_1 to the $START$ vector, and compute the distribution over the first word y_1. Accordingly, we pick the argmax from the distribution, set its embedding vector as x_2, and repeat the procedure until the END token is generated.

3.6. Softmax Loss.

At every time-step, we generate a score for each word in the vocabulary. We then use the ground truth words in combination with the softmax function to compute the losses and gradients. We sum the losses over time and average them over the minibatch. Since we operate over minibatches and because different generated sentences may have different lengths, we append $NULL$ tokens to the end of each caption so that they all have the same lengths. In addition, our loss function accepts a mask array that informs it on which elements of the scores counts towards the loss in order to prevent the $NULL$ tokens to count towards the loss or gradient.

3.7. Optimization.

We use Stochastic Gradient Descent (SGD) with minibatches of 25 image-sentence pairs and momentum of 0.95. We cross-validate the learning rate and the weight decay. We achieved our best results using Adam, which is a method for efficient stochastic optimization that only requires first-order gradients and computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients [28]. Adam's main advantages are that the magnitudes of parameter updates are invariant to rescaling of the gradients, its step-size is approximately bounded by the step-size hyperparameter, and it automatically performs a form of step-size annealing [28].

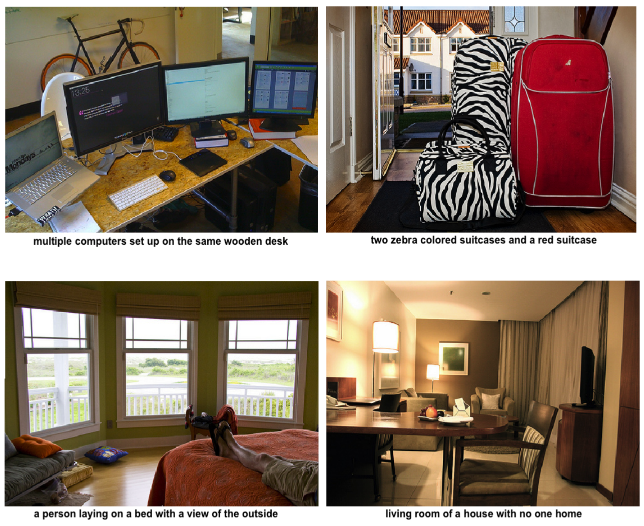

multiple computers set up on the same wooden desk
two zebra colored suitcases and a red suitcase
a person laying on a bed with a view of the outside
living room of a house with no one home

Figure 4: Example images with associated captions contained in the dataset.

4. Experiments and Results

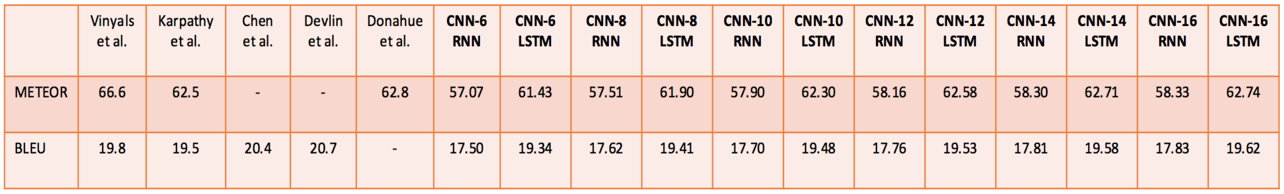
4.1. Dataset

We will use the 2014 release of the Microsoft COCO dataset which has become the standard testbed for image captioning [29]. The dataset consists of 80,000 training images and 40,000 validation images, each annotated with 5 captions written by workers on Amazon Mechanical Turk and associated categories. Four example images with captions can be seen in Figure 4. We convert all sentences to lower-case, and discard non-alphanumeric characters.

4.2. Evaluation Metric

For each image we expect a caption that provides a correct but brief explanation in valid English of the images. The closer the generated caption is to the captions written by workers on Amazon mechanical Turk the better.

The effectiveness of our model is tested on 40,000 images contained in the Microsoft COCO dataset. We evaluate the generated captions using the following metrics: BLEU (Bilingual Evaluation Understudy) [30], and METEOR (Metric for Evaluation of Translation with Explicit Ordering) [31][32]. 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 computed by counting the number of matches between the n-grams of the candidate caption and the n-grams of the reference caption. METEOR was designed to fix some of the problems found in the more popular BLEU metric, and also produce good correlation with human judgement at the sentence or segment level [30]. METEOR differs from the BLEU metric in that BLEU seeks correlation at



	Vinyals et al.	Karpathy et al.	Chen et al.	Devlin et al.	Donahue et al.	CNN-6 RNN	CNN-6 LSTM	CNN-8 RNN	CNN-8 LSTM	CNN-10 RNN	CNN-10 LSTM	CNN-12 RNN	CNN-12 LSTM	CNN-14 RNN	CNN-14 LSTM	CNN-16 RNN	CNN-16 LSTM
METEOR	66.6	62.5	-	-	62.8	57.07	61.43	57.51	61.90	57.90	62.30	58.16	62.58	58.30	62.71	58.33	62.74
BLEU	19.8	19.5	20.4	20.7	-	17.50	19.34	17.62	19.41	17.70	19.48	17.76	19.53	17.81	19.58	17.83	19.62

Figure 5: Comparison of METEOR and BLEU scores obtained by our models and state-of-the-art models [21, 33, 34, 35, 36]. CNN-x indicates that the CNN model has a total of x convolutional plus fully connected layers.

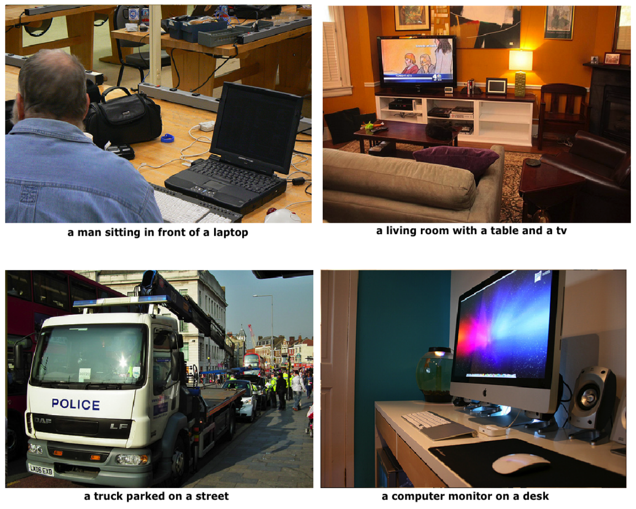

a man sitting in front of a laptop
a living room with a table and a tv
a truck parked on a street
a computer monitor on a desk

Figure 6: Examples of captions generated by our models.

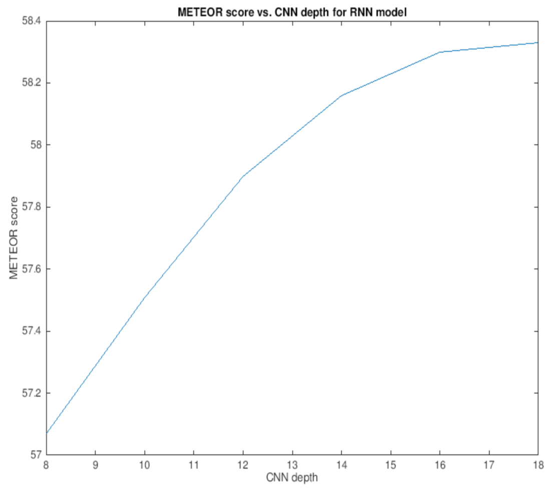


Figure 7: CNN architectures with more layers extract image features that can lead to better RNN generated image descriptions.

the corpus level [31]. For both metrics (BLEU, and METEOR) the higher the score, the better the candidate caption is [30][31][32].

4.3. Results

Our models generate high quality descriptions of images in valid English (Figure 5 and 6). As can be seen from example sentences in Figure 6, the model discovers interpretable visual-semantic correspondences. It even discovers relationships between the objects in the images such as that the man is sitting ”in front” of the computer, and that the truck is parked ”on the street.” The generated descriptions are accurate enough to be helpful for automatic surveillance technology. In general, we find that a relatively large portion of generated sentences (27%) can be found in the training data.

We report the METEOR, and BLEU scores in Figure 5 and compare them to the results obtained in the literature. Our LSTM models achieve close to state-of-the-art performance. They perform slightly better than our RNN models, i.e. they achieve higher BLEU, and METEOR scores. This is most likely because LSTMs are better able to better able to classify, process and predict time series when

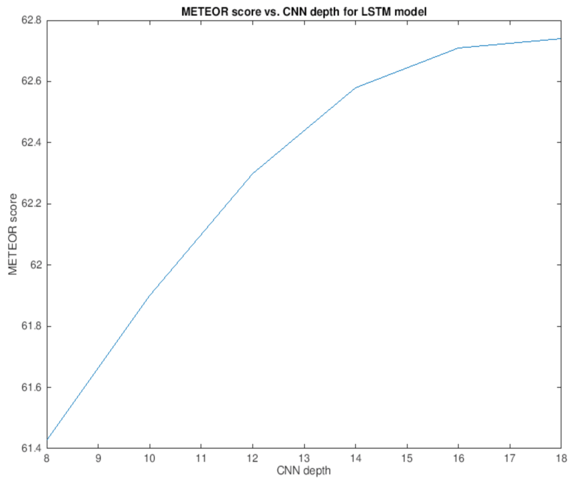


Figure 8: CNN architectures with more layers extract image features that can lead to better LSTM generated image descriptions.

there are very long time lags of unknown size between important events such as with the sentences generated by our model. The vanilla RNN suffers from the vanishing gradient problem, meaning that The beginning of a generated sentence often contains important information about the end of the generated sentence, and as a result models such as the LSTM are relatively well suited for the sentence generation task.

As can be seen in Figure 7 and 8 increasing the depth of the CNN leads to better image features for the image description generation task. Both the LSTM and the RNN models generate better captions using image features obtained by deeper CNN models. In future work, it might be interesting to investigate the cause of the better results obtained using deeper nets. We expect that it is related to the fact that deeper nets can learn more complex functions so that they can better capture the regularities in the training data. Our best model is the CNN of depth 16 in combination with the LSTM. It achieves close to state-of-the-art performance, i.e. METEOR score 62.74 and BLEU score 19.62.

5. Conclusion

We have presented a deep learning model that automatically generates image captions with the goal of increasing the effectiveness of security cameras. Our described model is based on a CNN that encodes an image into a compact representation, followed by a RNN or LSTM that generates corresponding sentences based on the learned image features. We showed that this model achieves comparable to state-of-the-art performance, and that the generated captions are highly descriptive of the objects and scenes depicted on the images. Because of the high quality of the generated image descriptions, camera surveillance based security can greatly benefit. The police or the owner of the camera can decide to take actions based on the image captions generated by our model. Future work can explore more different types of CNN architectures, such as for example CNNs with even more layers. It might also be interesting to explore different CNN training techniques such as other hyperparameter tuning optimization strategies. In addition, one could try using transfer learning to optimize the weights of the CNN based on the quality of the image descriptions it generates.

Acknowledgments

We would like to thank the CS231A course staff for their ongoing support.

References

[1] Tarun Wadhwa. ”The Next Privacy Move: Cameras that Judge your Every Move.” *Forbes*. (2012). Web. 3 Jun. 2016

[2] Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ”ImageNet Large Scale Visual Recognition Challenge.” *International Journal of Computer Vision Int J Comput Vis 115.3* (2015): 211-52. Web. 2 Jun. 2016

[3] Everingham, Mark, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. ”The Pascal Visual Object Classes (VOC) Challenge.” *International Journal of Computer Vision Int J Comput Vis 88.2* (2009): 303-38. Web. 22 May 2016

[4] Farhadi, Ali, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. ”Every Picture Tells a Story: Generating Sentences from Images.” *Computer Vision ECCV 2010 Lecture Notes in Computer Science* (2010): 15-29. Web. 5 Apr. 2016

[5] Kulkarni, Girish, Visruth Premraj, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C. Berg, and Tamara L. Berg. ”Baby Talk: Understanding and Generating Simple Image Descriptions.” *Cvpr 2011* (2011). Web. 27 May 2016

[6] Li, Li-Jia, R. Socher, and Li Fei-Fei. ”Towards Total Scene Understanding: Classification, Annotation and Segmentation in an Automatic Framework.” *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009). Web. 21 Apr. 2016

[7] Gould, Stephen, Richard Fulton, and Daphne Koller. ”Decomposing a Scene into Geometric and Semantically Consistent Regions.” *2009 IEEE 12th International Conference on Computer Vision* (2009). Web. 6 May 2016

[8] Fidler, Sanja, Abhishek Sharma, and Raquel Urtasun. ”A Sentence Is Worth a Thousand Pixels.” *2013 IEEE Conference on Computer Vision and Pattern Recognition* (2013). Web. 18 May 2016

[9] Li, Li-Jia, and Li Fei-Fei. ”What, Where and Who? Classifying Events by Scene and Object Recognition.” *2007 IEEE 11th International Conference on Computer Vision* (2007). Web. 10 Apr. 2016

[10] Lazaridou, Angeliki, Nghia The Pham, and Marco Baroni. ”Combining Language and Vision with a Multimodal Skip-gram Model.” *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (2015). Web. 23 May 2016

[11] Hodosh, Young, and Hockenmaier. ”Framing image description as a ranking task: data, models and evaluation metrics.” *Journal of Artificial Intelligence Research* (2013). Web. 3 Apr. 2016

[12] Socher, Richard, Andrej Karpathy, Quoc V. Le, Christopher Manning, and Andrew Y. Ng. ”Grounded compositional semantics for finding and describing images with sentences.” *Transactions of the Association for Computational Linguistics (TACL)* (2014). Web. 24 May 2016

[13] Ordonez, Vicente, Girish Kulkarni, and Tamara L. Berg. ”Im2text: Describing images using 1 million captioned photographs.” *NIPS: 1143-1151* (2011). Web. 29 May. 2016

[14] Jia, Yangqing, Mathieu Salzmann, and Trevor Darrell.

"Learning Cross-modality Similarity for Multinomial Data." *2011 International Conference on Computer Vision* (2011). Web. 28 May 2016

[15] Kuznetsova, Polina, Vicente Ordonez, Alexander C. Berg, Tamara Berg, and Yejin Choi. "Collective generation of natural image descriptions." *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics 1* (2012): 359:368. Web. 30 Apr. 2016

[16] Li, Siming and Kulkarni, Girish and Berg, Tamara L. and Berg, Alexander C. and Choi, Yejin. "Composing simple image descriptions using web-scale n-grams." *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: 220-228* (2011). Web. 27 Apr. 2016

[17] Kuznetsova, Polina, Vicente Ordonez, Tamara Berg, Yejin Choi. "TREETALK: Composition and Compression of Trees for Image Descriptions." *Transactions of the Association for Computational Linguistics 2* (2014): 351-362. Web. 1 Apr. 2016

[18] Gupta and Mannem. "From image annotation to image description. In Neural information processing." *Springer* (2012). Web. 7 Apr. 2015

[19] LeCun, Bottou, Bengio, and Haffner. "Gradient- based learning applied to document recognition." *Proceedings of the IEEE* (1998): 86(11):22782324. Web. 27 May 2016

[20] Krizhevsky, Sutskever, and Hinton. "Imagenet classification with deep convolutional neural networks." *NIPS* (2012). Web. 28 Apr. 2016

[21] Karpathy, Andrej, and Li Fei-Fei. "Deep Visual-semantic Alignments for Generating Image Descriptions." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). Web. 29 May 2016

[22] Kiros Ryan, Rich Zemel, and Ruslan Salakhutdinov. "Multimodal neural language models." *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*: 595-603 (2014). Web. 21 May 2016

[23] Simonyan, Karen and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *CoRR* (2014). Web. 28 May 2016

[24] Hochreiter, Sepp, and Jrgen Schmidhuber. "Long Short-Term Memory." *Neural Computation 9.8* (1997): 1735-780. Web. 23 Apr. 2016

[25] Graves, Alex. "Generating sequences with recurrent neural networks." *CoRR* (2013). Web. 30 May 2016

[26] Graves, Alex and Navdeep Jaitly. "Towards end-to-end speech recognition with recurrent neural networks." *Proceedings of the 31st International Converence on Machine Learning (ICML-14): 1764-1772* (2014). Web. 28 May 2016

[27] Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. "Distributed representations of words and phrases and their compositionality." *Advances in Neural Information Processing Systems (NIPS) 26: 3111-3119* (2013). Web. 29 Apr. 2016

[28] Kingma, Diederik and Jimmy Ba. "Adam: A method for stochastic optimization." *CoRR* (2015). Web. 19 May 2016

[29] Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollr, and C. Lawrence Zitnick. "Microsoft COCO: Common Objects in Context." Computer Vision ECCV 2014 Lecture Notes in Computer Science (2014): 740-55. Web. 27 May 2016

[30] Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu, Bleu: a method for automatic evaluation of machine translation." *Proceedings of the 40th Annual Meeting on Association for Computation Linguistics (ACL): 311-318* (2002). Web. 24 May 2016

[31] Denkowski, Michael, and Alon Lavie. "Meteor Universal: Language Specific Translation Evaluation for Any Target Language." *Proceedings of the Ninth Workshop on Statistical Machine Translation* (2014). Web. 22 Apr. 2016

[32] Vedantam, Ramakrishna, C. Lawrence Zitnick, and Devi Parikh. "CIDEr: Consensus-based Image Description Evaluation." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). Web. 24 May 2016

[33] Vinyals, Oriol, Alexander Toshev, Samy Bengio, and Dumitru Erhan. "Show and Tell: A Neural Image Caption Generator." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). Web. 25 May 2016

[34] Chen, Xinlei and C. Lawrence Zitnick. Learning a Recurrent Visual Representation for Image Caption Generation. *CoRR abs/1411.5654* (2014). Web. 19 May 2016

[35] Fang, Hao, Saurabh Gupta, Forrest Iandola, Rupesh K. Srivastava, Li Deng, Piotr Dollar, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Geoffrey Zweig. "From Captions to Visual Concepts and Back." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). Web. 27 Apr. 2016

[36] Donahue, Jeff, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Trevor Darrell, and Kate Saenko. "Long-term Recurrent Convolutional Networks for Visual Recognition and Description." *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2015). Web. 1 Jun. 2016